On approximating non-convex value functions in stochastic dual dynamic programming and related decomposition methods

Zur Erlangung des akademischen Grades eines

Doktors der Wirtschaftswissenschaften
(Dr. rer. pol.)

von der KIT-Fakultät für Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

Christian Füllner, M.Sc.

Tag der mündlichen Prüfung:	27. Juni 2024
Referent:	Prof. Dr. Steffen Rebennack
Korreferent:	Prof. Dr. Andy Sun

Karlsruhe (2024)

Science is not only a disciple of reason but, also, one of romance and passion

Stephen Hawking

Acknowledgments

First and foremost, I would like to thank my advisor Steffen Rebennack for his great patience and his excellent guidance, advice and support during my academic career, and especially during the work on this thesis. He has continuously opened up new opportunities and challenges for me, which helped me to grow as a researcher, teacher, and as a person. I greatly appreciate the freedom that he gave me to develop and shape my own research projects, but also to reconcile my work at KIT with several side jobs over the years.

I would also like to thank Andy Sun, who is the co-referee for this thesis, and Wolf Fichtner as members of the thesis defense committee.

Oliver Stein and Stefan Nickel deserve my utmost thanks for their phenomenal teaching, presenting mathematical ideas with both rigor and intuition, which aroused my interest in Operations Research early on in my studies, and eventually made me pursue an academic career in this field. After finishing my Master's thesis at his research group, Oliver also supported me in publishing my first ever scientific paper and visiting my first conference, for which I am very grateful.

Moreover, I also would like to express my deepest gratitude to Peter Kirst, who is a good friend without whom I would not be where I am today. From the first day when I became a tutor in OR under his guidance, we developed a close mentor-mentee relationship, which allowed me to conduct ambitious scientific research even before starting my doctoral studies. Since then Peter has been one of my main collaborators, advisors and advocates in research and teaching.

I am really thankful to all the other colleagues that I shared my time with at KIT for the joint work, the helpful discussions and the many shared breaks or evenings. In particular, I want to mention Marcel Sinske, Christoph Neumann, Robert Mohr, Maren Beck, Stefan Schwarze, Alasdair Warwicker, Anil Kaya, Hendrik Otto, Leo Schleier, Jakob Schöffer and Emil Kraft. I also have to thank our secretaries, Erika Simon and Michaela Fahrner who quickly and reliably took care of all the administrative processes behind my work, and made my life much easier.

In late 2021, I had the great opportunity to visit Andy Sun and his research group at Georgia Institute of Technology in Atlanta in the United States. I am beyond thankful to Andy for not only hosting me for three months at Georgia Tech, but closely collaborating on scientific projects with me for over two years now. I also would like to thank Andy's former students Shixuan Zhang and Filipe Cabral for many fruitful discussions, Prof. Steffen Rebennack for encouraging me in this endeavor and the Karlsruhe House of Young Scientist (KHYS) for its generous financial support of my research visit. Many thanks also to the Nettuno family for hosting me during my visit to Atlanta and letting me experience the American way of life first-hand.

Moreover, I want to thank my former employer I.C.I.S., especially my team leader Viktor Wolf and my colleague Karsten Walke, for the opportunity to work with them for three and a half years during my doctoral studies and to combine my theoretical work at KIT with practical experience in the energy sector.

I am also grateful to all my research peers for their scientific input that contributed to this work. In particular, I want to mention Nils Löhndorf and Bikey Seranilla who provided a lot of data for my computational experiments, and Oscar Dowson for his outstanding Julia implementation of SDDP which forms the backbone of many of my own implementations.

In addition, I have to express how delighted I am with all the great friends that stuck with me over the last few years through the good, but also through the hard and stressful times. A big thank you to Alexander Jaus, Anastasia Evsyukova, David Wolfer, Marco Gehring, Yingkai Yang, Ngoc Anh Tran, Timurcin Kizilnar, Patrick Tu, Hannes Weinreuter, André Fuchs and Marlene Sonntag. I would also like to mention my student group fuks e.V. at KIT and my sports club FC Südstern Karlsruhe. I really appreciate all of your support.

At last, I would like to thank my beloved family for their continuous and unconditional support, and for always putting faith in me, no matter which challenges I face. I am deeply indebted to my parents Michael and Uta, to my brother Rüdiger and his wife Lisanne. Most importantly, words cannot express my gratitude to my twin brother Christoph who I have not only shared most of my life with, but also this journey as a doctoral student. He is a constant inspiration for me, and one of the main drivers behind my academic career. His daily support, encouragement and valuable advice were absolutely essential to the success of this project.

Karlsruhe, April 2024
Christian Füllner

Abstract

Today, stochastic dual dynamic programming (SDDP) is one of the state-of-the-art algorithms to solve multistage stochastic optimization problems. One of its key ideas, borrowed from Benders decomposition, is to decompose a multistage problem into several subproblems which are coupled by so-called value functions. As these functions are not known explicitly, they are iteratively approximated using cutting-planes (linear cuts). However, in order for this to work, several crucial assumptions have to be satisfied, among them linearity (or at least convexity) of all occurring functions as well as stagewise independence of the uncertain data. In many applications, this is not guaranteed. For this reason, various enhancements of SDDP have been proposed which allow to relax some of these assumptions.

The research in this thesis addresses an open challenge in this regard, which is to extend SDDP to problem classes for which non-convexities arise in the value functions, and thus linear cuts are not sufficient to guarantee (almost sure) convergence to an optimal solution. The focus is on three specific types of problems: a) including integrality constraints, b) including non-convex functions, c) with the uncertain data modeled by a non-convex autoregressive process. It is shown that in all three cases a tight approximation of the value functions can be achieved using special non-convex cuts. By careful design, based on these results solution methods with proven convergence are developed, the one for case c) being the first of its kind. This extends the toolbox of SDDP-like algorithms substantially.

In addition, a novel framework is presented to generate linear cuts with favorable properties, which may help to improve the computational performance of the existing SDDP-derivative stochastic dual dynamic integer programming (SDDiP). Finally, as many of the proposed linear and non-convex cuts rely on special Lagrangian relaxations, a detailed theoretical study of these relaxations and their properties with respect to the value functions is conducted.

All algorithmic contributions are tested in case studies on real-world applications, such as unit commitment, hydrothermal scheduling or lot-sizing, confirming their effectiveness and their potential. However, the studies also reveal that due to a considerable computational overhead in generating and incorporating the proposed (non-convex) cuts, efficiency and computational tractability still remain major challenges for SDDP-like algorithms given non-convex problems.

Contents

Part I

Research summary

Chapter 1

Introduction

1.1 Motivation and research outline

In many decision-making situations in practice, multiple subsequent decisions have to be taken that are coupled by their effects on a common system state. Often at least some of the decision-relevant data are subject to uncertainty. In order to hedge against the risk of bad outcomes, this uncertainty should be taken into account in the decision-making. To compute an optimal policy of decisions, this situation can be mathematically modeled as a multistage stochastic optimization problem and can then be solved by a suitable solution method.

In order to keep this kind of problem tractable, it is usually assumed that the uncertainty is modeled by discrete and finite random variables, and thus can be represented by a finite scenario tree. However, even with this assumption, solving a multistage stochastic optimization problem as a standalone problem is usually computationally intractable. Therefore, a common idea, borrowed from Benders decomposition (Benders, 1962), is to decompose it into several subproblems which are coupled by so-called value functions. As these value functions are not known explicitly, they are iteratively approximated using cutting-planes (also called *linear cuts* or *Benders cuts*). If all subproblems are linear (or at least convex) without integer constraints, then the value functions are convex, and therefore these approximations are sufficient to guarantee finite convergence to an optimal solution. This constitutes the algorithm *nested Benders decomposition* (NBD) (Birge, 1980).

In general, the size of finite scenario trees grows exponentially in the number of stages of the multistage problem. Therefore, despite being preferrable to solving a single problem, applying NBD becomes too expensive for all but moderately sized scenario trees. As long as the uncertainty in the problem data is stagewise independent, however, the scenario tree collapses to a recombining tree. This means that the number of subproblems to be solved grows only linearly in the number of stages, even if the total number of considered scenarios is still exponential in the number of stages. This gap in the number of subproblems and scenarios can be exploited by considering only a sample of scenarios in each iteration, while still approximating all value functions. In this case, still (almost sure) finite convergence can be proven. This is the key principle of *stochastic dual dynamic programming* (SDDP) (Pereira and Pinto, 1991). To this date, it is one of the state-of-the-art algorithms to solve multistage stochastic linear (or convex) programs.

This thesis makes several contributions to the research on SDDP and related decomposition algorithms. It is composed of five scientific papers collectively.

Since its invention, SDDP has been applied in numerous case studies, most prominently, but not exclusively in power system optimization. Also various attempts have been made to modify the algorithm, either to improve its computational performance or to relax some of its crucial assumptions and to extend it to more general problem classes. Therefore, SDDP has developed into a broad research field. This motivates the first major contribution of this thesis.

(1) We provide a comprehensive tutorial-type review of SDDP as a research field. [Füllner and Rebennack (2023), see Paper A]

One of the open challenges that is identified is to develop effective and efficient extensions to cases in which the subproblems and the associated value functions become non-convex. In such cases, linear cuts are not sufficient to guarantee (almost sure) convergence of NBD or SDDP to an optimal solution because in general, either validity or tightness of these cuts is compromised. The main research objective of this thesis is to contribute to closing this research gap. The focus is on three specific types of problems:

a) problems including integrality constraints,

b) problems including non-convex functions,

c) problems for which the uncertain data is modeled by a non-convex *autoregressive* (AR) process.

All these cases are relevant in practice, for instance they may occur in unit commitment (Zou et al., 2019a; Hjelmeland et al., 2019) or hydrothermal scheduling (Löhndorf and Shapiro, 2019) problems.

For case a), a first pivotal approach exists in *stochastic dual dynamic integer programming* (SDDiP), an SDDP-variant where instead of classical Benders cuts linear cuts are generated based on Lagrangian duality (Zou et al., 2019b). If all state variables coupling the stages are binary, it can be shown that these Lagrangian cuts are tight, and thus sufficient to ensure (almost sure) finite convergence of NBD or SDDP. However, to exploit this approach in the case of general mixed-integer state variables, the state variables have to be approximated using binary variables, which is computationally costly. Moreover, this requires a decision on the number of binary variables in advance, usually without knowledge which choice will be sufficient to guarantee a certain approximation quality.

Additionally, for cases a) and b) the generation of *non-convex cuts* in extensions of SDDP based on augmented Lagrangian duality has been proposed during our work on this thesis (Ahmed et al., 2022; Zhang and Sun, 2022). We still proceeded with our work, but with taking these new research findings into account.

Overall, in this thesis, we show that for all three cases, a), b) and c) a tight approximation of the non-convex value functions can be achieved by using carefully designed non-convex cuts. These cuts can be incorporated into NBD-like or SDDP-like algorithms to develop solution methods with proven convergence. More precisely, we make the following contributions.

(2) With respect to case a), we investigate how SDDiP can be "dynamized" in the sense that the binary approximation of the state variables is refined throughout the solution process if necessary instead of being static. This can be done without compromising the validity of previously generated cuts. Interestingly, as we show, following this approach leads to a lift-and-project cut generation process, which results in tight non-convex cuts approximating the non-convex value functions in the original state space. We call the resulting algorithm *Dynamic SDDiP* and prove its convergence. We also show in detail how our non-convex cuts and their generation process differ from

the proposals in Ahmed et al. (2022); Zhang and Sun (2022), and explain the strengths and weaknesses of each approach.
[Füllner and Rebennack (2022); Füllner et al. (2024b), see Papers B and C]

(3) With respect to case b), we propose a new solution framework that can be seen as an extension of NBD to general non-convex multistage stochastic problems, and may be generalized to include sampling in the future. Essentially, this framework incorporates Dynamic SDDiP and dynamically refined piecewise linear relaxations of nonlinear functions as key components. This method is called *non-convex nested Benders decomposition* (NC-NBD). Again, we provide a convergence proof. Apart from the method in Zhang and Sun (2022) NC-NBD is the first exact solution method for this general problem class. In contrast to the former, nonlinear problems only have to be solved occasionally instead of each iteration.
[Füllner and Rebennack (2022), see Paper B]

(4) With respect to case c), we propose an extension of SDDP that can handle non-convex log-linear AR processes describing the uncertainty in the right-hand side of a multistage stochastic linear problem. A major component of this extension is the generation of cuts that are non-convex in the history of the AR process, but still linear in the original state variables. As we show, this allows for incorporation of these cuts into SDDP. For this type of problem, our work is the first one that does not require an approximation of the AR process. Therefore, it allows for more flexibility and accuracy in modeling uncertain data in SDDP.
[Füllner and Rebennack (2024), see Paper E]

In addition to studying non-convex approximations, we also make contributions to the generation of linear cuts, which are for instance used in SDDiP or heuristic approaches to solve non-convex problems.

(5) Based on recent advances for Benders decomposition (Fischetti et al., 2010; Brandenberg and Stursberg, 2021; Hosseini and Turner, 2021) and two-stage stochastic programming (Chen and Luedtke, 2022) we present a novel framework to generate Lagrangian cuts with favorable properties, such as maximum cut depth, facet-defining behavior or Pareto-optimality. These cuts

may prove helpful in improving the computational performance of SDDiP.
[Füllner et al. (2024a), see Paper D]

Finally, as many of our proposed solution methods and (linear or non-convex) cuts rely on the concepts of Lipschitz regularization and Lagrangian duality, we shed light on both concepts from a theoretical perspective.

(6) We provide a thorough theoretical study on Lipschitz regularization, Lagrangian duality and their relations to the value functions as well as each other. In particular, we analyze in detail the effects that so-called copy constraints and Lipschitz regularization have on Lagrangian duals, the generation of Lagrangian cuts and the properties of those cuts. By doing that, we are able to generalize the tightness result from SDDiP (Zou et al., 2019b) to the Lipschitz regularized case.
[Füllner et al. (2024b), see Paper C]

In order to evaluate the performance of our theoretical and algorithmic contributions, we also study them computationally.

(7) We conduct several computational experiments and case studies on real-world applications, such as unit commitment, hydrothermal scheduling or lot-sizing problems, to test the proposed solution methods. The results confirm their effectiveness and their potential. However, the studies also reveal a considerable computational overhead in generating and incorporating the proposed (non-convex) cuts in many cases. This shows that efficiency and computational tractability still remain major challenges when applying NBD-like and SDDP-like algorithms to non-convex problems.

1.2 Structure of the thesis

This thesis is organized in two parts. Part I provides a general overview on the research conducted in this thesis and puts the individual scientific papers into context. In Chapter 2, some theoretical background is introduced. In Chapter 3, then the research objectives of this work are motivated. The conducted studies and their main contributions are summarized in Chapter 4. Finally, Chapter 5 concludes with a summary, discusses limitations of the presented results and provides an outlook on potential future research directions. Part II of this thesis contains the full manuscripts of all five scientific papers.

Chapter 2

Theoretical background

In this section, we introduce some notation and give a basic overview on the theoretical background that is required to understand the motivation for as well as the main contributions of the research presented in this thesis. For details, we refer to the papers in Part II.

2.1 Mathematical definitions

First, we present some important mathematical concepts that are used throughout this thesis. We define $\overline{\mathbb{R}} := \mathbb{R} \cup \{\pm\infty\}$.

Definition 2.1.1 (Convex set). *A set $M \subseteq \mathbb{R}^n$ is called* convex *if for all $\lambda \in (0,1)$ and all $x, y \in M$ also $\lambda x + (1-\lambda)y \in M$.*

Definition 2.1.2 (Epigraph). *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a function. Then the set*

$$epi(f) := \Big\{(x,\alpha) \; : \; x \in \mathbb{R}^n, \alpha \in \mathbb{R}, \alpha \geq f(x)\Big\}$$

is called the epigraph *of f.*

Definition 2.1.3 (Convex function). *A function $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ is* convex *if its epigraph $epi(f)$ is a convex subset of $\mathbb{R}^{n+1}$.*

The last definition is taken from Rockafellar (1970). Note that it deviates from the most common definition of convex functions, but is more convenient in our context, as it allows $f(\cdot)$ to take infinite values. Moreover, any function $f(\cdot)$

satisfying the standard definition can be extended to one satisfying the definition presented above.

Definition 2.1.4 (Lipschitz continuous function)**.** *Let $M \subseteq \mathbb{R}^n$ and let $\|\cdot\|$ be some norm. A function $f : M \to \overline{\mathbb{R}}$ is* Lipschitz continuous *on M (with respect to $\|\cdot\|$) if there exists some constant $L > 0$ such that for all $x, y \in M$*

$$|f(x) - f(y)| \leq L\|x - y\|.$$

2.2 Multistage decision processes

One of the main premises in multistage stochastic programming is that in some given system, several subsequent decisions have to be taken over a known and finite time horizon $[T] := \{1, \dots, T\}$, with $T \in \mathbb{N}$. More precisely, on each stage $t \in T$ decisions $x_t \in \mathbb{R}^{d_t}$ have to be taken, where $d_t \in \mathbb{N}$ denotes the dimension.

The challenge is that these subsequent decisions are not independent of each other, but coupled. This means that taking a specific decision at one point of time may restrict the feasible decisions that can be taken at later points. We should note that in multistage stochastic programming x_t takes both the role of a decision variable (or *action*) that is actively taken by the decision-maker and an underlying system *state variable* that these decisions rely on. Sometimes, purely local decisions are modeled by additional vectors $y_t \in R^{\bar{d}_t}, \bar{d}_t \in \mathbb{N}$, but we abstain from this in this introduction. In dynamic programming, on the contrary, actions and states are clearly distinguished in notation.

The second main premise is that some of the data in the decision-making problem are uncertain and only revealed over time. Usually, the first stage data are assumed deterministic, but we include it in our formal description.

For each stage $t \in [T]$, let $(\Omega_t, \mathscr{F}_t, \mathbb{P}_t)$ denote a probability space, and let the sigma algebras define a filtration $\mathscr{F}_1 \subseteq \mathscr{F}_2 \cdots \subseteq \mathscr{F}_T$. Then, for each t, we can define an $\mathscr{F}_t$-measurable random vector $\boldsymbol{\xi}_t : \Omega_t \to \mathbb{R}^{\kappa_t}, \kappa_t \in \mathbb{N}$, on this probability space to model the uncertain data of this stage. To distinguish the random vectors $\boldsymbol{\xi}_t$ from their realizations ξ_t, we use bold font. In general, bold font is used to signify random vectors. The support of $\boldsymbol{\xi}_t$ is denoted by Ξ_t, with Ξ_1 a singleton. We can combine all random vectors to a stochastic process $(\boldsymbol{\xi}_t)_{t \in [T]}$ with realizations $\xi := (\xi_1, \dots, \xi_T)$.

Reconsidering the decision process given this uncertainty, the paradigm is that at the first stage, decisions x_1 are taken which hedge against the uncertainty over the whole time horizon $[T]$ (so-called *here-and-now* decisions). On all following stages $t = 2, \dots, T$, in contrast, decisions x_t can be taken *after* the uncertain data for that specific stage has realized, that is, new information can be taken into account (so-called *wait-and-see* decisions). In other words, the decision x_t can be taken dependent on how the uncertain data up to stage t unfolds. This makes $\boldsymbol{x}_t(\cdot)$ a *function* of ξ_t, and by coupling a function of the whole history $\xi_{[t-1]} := (\xi_1, \dots, \xi_{t-1})$. Importantly, $\boldsymbol{x}_t(\cdot)$ is also $\mathscr{F}_t$-measurable, and thus *non-anticipative*, meaning that it does only depend on historic realizations, but does not anticipate future events.

The multistage decision process with uncertainty is visualized in Figure 2.1.

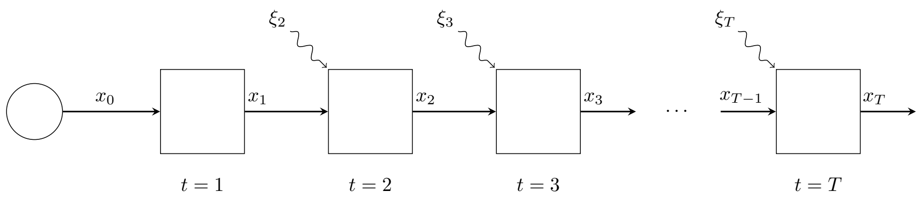


Figure 2.1: Multistage decision process with uncertainty (ξ_1 is deterministic).

A common example to illustrate this is a hydrothermal power system in which a thermal generator and a hydro power plant are available to meet the electricity demand over time at minimum cost. Whereas thermal generation is cost-intensive due to the necessity to purchase fuel, draining water from a hydro reservoir through a turbine to generate electricity has almost no operational cost. Therefore, given an electricity demand at some stage t, the naive strategy is to use as much hydro power as possible to satisfy the demand. However, this kind of strategy completely neglects the future consequences of this decision. Using too much water at stage t may lead to a shortage of water later on, especially in dry periods without considerable inflows. This may directly translate to higher costs or a shortage of power at later stages. Moreover, future inflows are uncertain. For this reason, the potential value of storing water for later stages should already be taken into account in the decision-making at stage t. In other words, the ability to store water in reservoirs leads to a temporal coupling of the decisions over the stages.

The considered decisions x_t may have to satisfy certain constraints. For instance, in our previous power system example, the thermal generation and hydro generation together have to meet the demand at each stage t. Mathematically, the constraints can be modeled using some $\mathscr{F}_t$-measurable set-valued mapping $\boldsymbol{\mathcal{X}}_t(x_{t-1}, \xi_t)$, which is defined by several equations and inequalities. Moreover, we may enforce that x_t (or some of its components) are only allowed to take integer values. In addition to constraint satisfication, an objective function $f_t(\cdot, \xi_t)$ is used to evaluate the quality of different decisions x_t given a realization ξ_t. In our previous example, the objective function measures the costs.

The aim in the decision-making process is then to come up with a sequence of decision functions $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$, also called *policy*, which provides feasible decisions for each stage $t \in [T]$ and almost every realization of the uncertain data, while minimizing (or maximizing) the objective function *on average*. Therefore, we consider (conditional) expected values in the objective function. This decision-making problem can be modeled as an optimization problem. We formalize this in Sect. 2.3 and refer to it as a *multistage stochastic program* (MSP).

If $\boldsymbol{\xi}_t$ are continuous random vectors, then for all but very special cases, this optimization problem becomes computationally intractable (Rebennack, 2016). For this reason, in practice usually the true distributions are approximated by discrete ones, for example using a *sample average approximation* (SAA) (Shapiro et al., 2014). For the remainder of this thesis, we simply assume that $\boldsymbol{\xi}_t$ is a discrete and finite random variable for all $t \in [T]$ with a known conditional distribution $F_{|\xi_{[t-1]}}$ given a history $\xi_{[t-1]}$. This implies that over the whole time horizon, the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ can only take finitely many different realizations ξ^s, called *scenarios*, which we index by $s \in \mathcal{S}$, where $\mathcal{S}$ is a discrete index set.

In this setting, the uncertainty in the problem can be represented by a finite scenario tree, as illustrated in Figure 2.2.

2.3 A single-problem formulation

Using the previously described ingredients, we can now formalize MSP. There are different approaches to do this. We start with formulating it as a single optimization problem.

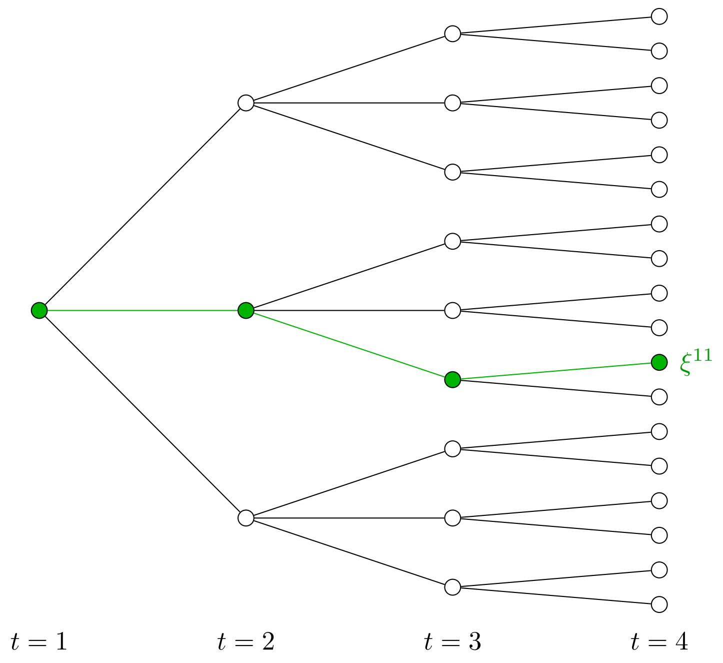


Figure 2.2: Finite scenario tree for an example with $T=4$. It models 18 scenarios ξ^s, with ξ^{11} highlighted.

Scenario representation. Given our assumption of discrete and finite random vectors, we can formulate MSP as a finite-dimensional optimization problem by replacing each decision function $\boldsymbol{x}_t(\xi_{[t]})$ with a finite number of variables $x_{ts}, s \in \mathcal{S}$. As becomes clear from Figure 2.2, to ensure non-anticipativiy, we have to make sure that $x_{ts} = x_{ts'}$ for all $t \in [T]$ and all $s' \in \mathcal{S}_{st}$, where the latter is the set of scenarios that share the same node of the tree with scenario s at stage t. This yields the problem

$$v^* = \begin{cases} \min\limits_{x_1, x_{2s}, \ldots, x_{Ts}} & f_1(x_1) + \sum\limits_{s \in \mathcal{S}} p_s \Bigg(\sum\limits_{t=2,\ldots,T} f_t(x_{ts}, \xi_t^s) \Bigg) \\ \text{s.t.} & x_1 \in \mathcal{X}_1 \\ & x_{ts} \in \mathcal{X}_t(x_{t-1,s}, \xi_t^s) \quad \forall s \in \mathcal{S}\ \forall t = 2, \ldots, T \\ & x_{ts} = x_{ts'} \quad \forall s \in \mathcal{S}, s' \in \mathcal{S}_{st},\ \forall t = 2, \ldots, T. \end{cases} \tag{2.1}$$

This deterministic problem is often referred to as the *extensive form* of MSP. In the case that $f_t(\cdot)$ and all functions describing $\mathcal{X}_t(\cdot, \xi_t^s)$ are linear, it is a large-scale *linear program* (LP).

Nodal representation. A second way to formulate the extensive form of MSP avoids using stage indices t and scenario indices s, but represents the scenario tree using a set of nodes $\mathcal{N}$. The root node is denoted by r. We define $\overline{\mathcal{N}} := \mathcal{N} \setminus \{r\}$ to address the set of nodes without the root node, $\widetilde{\mathcal{N}}$ to address the set of nodes without leaf nodes and denote by $\mathcal{N}(t)$ the nodes at stage t. For each node $n \in \overline{\mathcal{N}}$, the unique ancestor node is denoted by $a(n)$, and for each $n \in \mathcal{N}$, and the set of child nodes is denoted by $\mathcal{C}(n)$. The probability for some node n is $p_n > 0$. The transition probabilities between adjacent nodes $n, m \in \mathcal{N}$ can then be determined as $p_{nm} := \frac{p_m}{p_n}$. Using these definitions, problem (2.1) can be expressed as

$$v^* = \begin{cases} \min\limits_{x_n, n \in \mathcal{N}} & f_r(x_r) + \sum\limits_{n \in \overline{\mathcal{N}}} p_n f_n(x_n, \xi_n) \\ \text{s.t.} & x_r \in \mathcal{X}_r \\ & x_n \in \mathcal{X}_n(x_{a(n)}, \xi_n) \quad \forall n \in \overline{\mathcal{N}}. \end{cases} \tag{2.2}$$

In particular, this formulation does not require explicit non-anticipativity constraints, as they are implicitly considered by the nodal connections. Therefore, when we consider the full scenario tree, we mostly revert to this notation for the remainder of this chapter. However occasionally, we introduce deviating notation as well if this proves beneficial for later chapters.

The main issue with formulation (2.2) is that due to introducing a separate variable x_n for each node $n \in \mathcal{N}$, and the number of nodes growing exponentially in the number of stages T (see Figure 2.2), the problem size grows exponentially in T as well. Therefore, for large scenario trees, the problem becomes too large to be solved by monolith solution methods. For this reason, a lot of research in multistage stochastic programming is devoted to solution methods that decompose problems of type (2.1). In the next section, we introduce one natural way to achieve such a decomposition.

For the remainder of Part I, we take the following two assumptions with respect to problems (2.2) (and (2.1) analogously). First, we assume that the sets $\mathcal{X}_n(x_{a(n)}, \xi_n)$ are compact sets for all $x_{a(n)}$ and ξ_n. Second, we assume that $\mathcal{X}_1$ is non-empty and that we have *relatively complete recourse*. This means that for any node n and any feasible $x_{a(n)}$ there also exists some $x_n \in \mathcal{X}_n(x_{a(n)}, \xi_n)$. Finally, we assume that the functions $f_n(\cdot)$ are at least lower semicontinuous (l.sc.) for all

$n \in \mathcal{N}$. These assumptions combined guarantee that problem (2.2) is feasible and takes a finite optimal value v^*. This allows us to focus on the main contributions of our research without having to consider cases of infeasibility or unboundedness. Note, however, that in some of the research papers in Part II of this thesis, slightly different assumptions are taken, depending on the specific context.

2.4 Dynamic programming equations

As discussed before, solving MSP in its extensive form is often computationally prohibitive. Therefore, usually decomposition methods are applied in the solution process. An intuitive way to decompose the problem is readily available from the scenario tree depicted in Figure 2.2: The problem can be decomposed by stages and scenarios, or in other words, by nodes of the scenario tree. In order to still solve the original problem, the nodal subproblems have to be coupled. This yields the so-called *dynamic programming equations* (DPE), which exploit the well-known optimality principle by Richard E. Bellman (Bellman, 1957).

For any node $n \in \overline{\mathcal{N}}$ the DPE are given by

$$Q_n(x_{a(n)}) := \begin{cases} \min\limits_{x_n} & f_n(x_n) + \mathcal{Q}_{\mathcal{C}(n)}(x_n) \\ \text{s.t.} & x_n \in \mathcal{X}_n(x_{a(n)}), \end{cases} \tag{2.3}$$

where

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) := \sum_{m \in \mathcal{C}(n)} p_{nm} Q_m(x_n)$$

For leaf nodes $n \in \mathcal{N}$, we set $\mathcal{Q}_{\mathcal{C}(n)}(x_n) \equiv 0$. For the root node, we obtain

$$v^* = \begin{cases} \min\limits_{x_r} & f_r(x_r) + \mathcal{Q}_{\mathcal{C}(1)}(x_1) \\ \text{s.t.} & x_r \in \mathcal{X}_1. \end{cases}$$

The functions $Q_n(\cdot)$ are called *value functions* and the functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ are called *expected value functions* (or *cost-to-go functions*). These functions are the focal point of the research presented in this thesis.

Note that we define all (expected) value functions as functions mapping from $\mathbb{R}^{d_n}, d_n \in \mathbb{N}$, to $\overline{\mathbb{R}}$, meaning that they take the value $+\infty$ for values of $x_{a(n)}$ for which the subproblems are infeasible.

2.5 Approximating value functions

Whereas the DPE provide a way to decompose MSP into smaller subproblems, applying them in a solution method is challenging. The main challenge is that the (expected) value functions are not known explicitly. More precisely, they can be evaluated for any sequence $(x_n)_{n\in\mathcal{N}}$, but they are not known in a *functional form.* For this reason, a common approach is to iteratively approximate them, an idea that traces back to Benders decomposition (Benders, 1962) and Kelley's cutting-plane method (Kelley, 1960).

We denote the approximations of the expected value functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ by $\mathfrak{Q}_{\mathcal{C}(n)}(\cdot)$ and refer to them as *cut approximations.* By replacing $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ in problem formulation (2.3) with $\mathfrak{Q}_{\mathcal{C}(n)}(\cdot)$, we obtain the *approximate value function* and subproblems

$$\underline{Q}_n^i(x_{a(n)}^i) := \begin{cases} \min\limits_{x_n} & f_n(x_n) + \mathfrak{Q}_{\mathcal{C}(n)}^i(x_n) \\ \text{s.t.} & x_n \in \mathcal{X}_n(x_{a(n)}). \end{cases} \tag{2.4}$$

The iteration index i highlights that the approximations are iteratively updated within the solution method.

This approximation approach is often referred to as *single-cut* because the expected value functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ are each approximated by a single batch of cuts. Alternatively, each value function $Q_m(\cdot), m \in \mathcal{C}(n)$, can be approximated separately (*multi-cut* approach). For MSPs usually a single-cut approach is preferred for computational reasons, as much less cuts are added.

In any case, the cut approximations should satisfy some important properties, which we present in their form for a single-cut approach:

- **Validity.** They should be valid underestimators, *i.e.*, for any i, any $n \in \widetilde{\mathcal{N}}$ and any x_n they should satisfy

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \mathfrak{Q}_{\mathcal{C}(n)}^i(x_n).$$

This is important to consistently obtain lower bounds for v^*.

- **Tightness.** They should be tight approximators in the sense that for any i and any $n \in \widetilde{\mathcal{N}}$, whenever the cut approximation is updated from $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$ to $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ at some point x^i_n, it should satisfy
$$\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(x^i_n) = \underline{\mathcal{Q}}^{i+1}_{\mathcal{C}(n)}(x^i_n) := \sum_{m \in \mathcal{C}(n)} p_{nm} \underline{Q}^{i+1}_m(x^i_n).$$
This is important in order to achieve exact approximations and to solve the original MSP eventually.

- **Lipschitz continuity.** If the state variables x_n (or at least some of their components) are continuous, the cut approximations should be Lipschitz continuous. This is required to ensure (finite) convergence of the solution method to the true solution of MSP, as it ensures that the tightness property also leads to a sufficient improvement of the approximation quality in a neighborhood of x^i_n. It also prevents the cut approximations from becoming arbitrarily steep, which is important to prevent numerical issues.

How the approximations $\mathfrak{Q}_{\mathcal{C}(n)}(\cdot)$ can be determined for a specific instance of MSP is highly dependent on the properties of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, which in turn depend on the specific properties of the objective functions $f_n(\cdot)$ and the constraint sets $\mathcal{X}_n(\cdot)$.

2.6 Multistage stochastic linear programs

A special class of MSP that is well-explored in the literature is the class of *multistage stochastic linear program* (MSLP). Here, the objective function $f_n(\cdot)$ is a linear function $f_n(x_n) = c_n^\top x_n$ for all $n \in \mathcal{N}$ with some coefficient vector $c_n \in \mathbb{R}^{d_n}$. Moreover, for any $n \in \mathcal{N}$ and $x_{a(n)}$, the set $\mathcal{X}_n(x_{a(n)})$ is defined by finitely many affine functions, thus a convex polyhedron. In particular, all variables in x_n are continuous, so no integer requirements exist. More specifically, the constraint sets can be defined as

$$\mathcal{X}_n(x_{a(n)}) := \Big\{ x_n \in X_n \subset \mathbb{R}^{d_n} \ : \ T_n x_{a(n)} + W_n x_n = h_n \Big\} \tag{2.5}$$

with a non-empty polyhedron X_n, *e.g.*, modeling non-negativity constraints, and coefficient matrices T_n, W_n and vectors h_n of appropriate dimension. It is the coefficients in T_n, W_n, h_n and c_n that may be subject to uncertainty.

As indicated before, for MSLPs, the (expected) value functions $Q_n(\cdot)$ and $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ have some favorable properties (see Birge and Louveaux (2011), Chapter 3 for an idea of the proof).

Lemma 2.6.1. *For any node $n \in \overline{\mathcal{N}}$, the functions $Q_n(\cdot)$ defined in* (2.3)*, and for any node $n \in \widetilde{\mathcal{N}}$, the functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ defined in* (2.4) *are piecewise linear and convex functions in $x_{a(n)}$ on $\mathbb{R}^{d_{a(n)}}$ and x_n on $\mathbb{R}^{d_n}$, respectively.*

The convexity implies that these functions can be approximated from below by linear cuts (cutting-planes), while the piecewise linearity implies that in fact finitely many such functions suffice to achieve an exact representation of the true function. The main reason is that the elements T_n, h_n and $x_{a(n)}$ only appear in the *right-hand side* (RHS) of the LP (2.3). Therefore, the feasible set of its LP dual is independent of those elements. As it possesses finitely many extreme points, and by strong duality for LPs, the value function $Q_n(\cdot)$ can be expressed as the pointwise maximum of finitely many affine functions, which is piecewise linear and convex. From its definition, the same follows for $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$.

By approximating $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ with linear cuts and taking the pointwise maximum of these cuts, the cut approximations $\mathfrak{Q}_{\mathcal{C}(n)}(\cdot)$ are piecewise linear and convex as well. This also allows to still express the subproblems as LPs. Moreover, by backward recursion from stage T, the above properties do also hold for the approximate value functions.

Lemma 2.6.2. *For any stage $n \in \overline{\mathcal{N}}$, the function $\underline{Q}_n(\cdot)$ defined in* (2.4) *and its expectation $\underline{\mathcal{Q}}_{\mathcal{C}(n)}(\cdot)$ are piecewise linear and convex functions in $x_{a(n)}$ on $\mathbb{R}^{d_{a(n)}}$.*

2.7 Nested Benders decomposition

We present *nested Benders decomposition* (NBD) as a first solution method for MSLPs. It was first introduced in (Birge, 1980) and basically extends Benders decomposition (Benders, 1962) and the L-shaped method (van Slyke and Wets, 1969) to MSLPs.

The main idea of NBD is to traverse the scenario tree in forward and backward direction in each iteration to improve the cut approximations $\mathfrak{Q}_{\mathcal{C}(n)}(\cdot)$ of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ until convergence is achieved and an optimal solution for MSLP, *i.e.*, an optimal policy and the optimal value v^*, are computed.

We explain this in more detail now. In each iteration i, first a forward pass through the tree is conducted. Starting with stage 1, the approximate subproblem (2.4) is solved for each node, and the obtained solution x_n^i is handed forward to all successor nodes where it enters the subproblems as a parameter. We call x_n^i a *trial* solution because it is the current candidate for an optimal solution to the true subproblem. Following this approach, trial solutions are computed for all nodes $n \in \mathcal{N}$, defining a trial policy.

Evaluating this feasible trial policy in the original objective function, we obtain an upper bound to v^*:

$$\overline{v}^i := f_1(x_1^i) + \sum_{n \in \overline{\mathcal{N}}} p_n c_n^\top x_n^i.$$

The forward pass is illustrated in Figure 2.3.

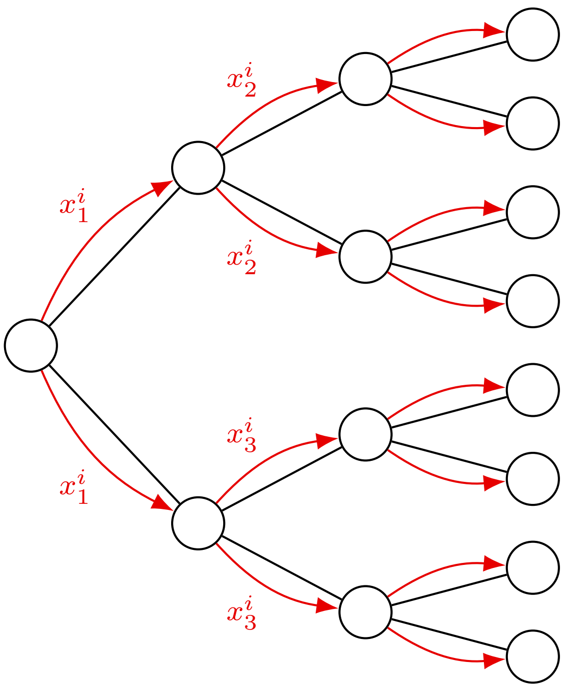


Figure 2.3: Forward pass illustration for NBD.

In the following backward pass through the scenario tree, starting with nodes $n \in \mathcal{N}_{T-1}$, the approximate subproblems (2.4) are updated by improving the cut approximations $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$. To this end, the LP duals of the (already updated) approximate subproblems (2.4) of each successor node $m \in \mathcal{C}(n)$ are solved. Then, the optimal dual multipliers π^i_m to the coupling constraints $T_m x_n + W_m x_m = h_m$ and the optimal value $\underline{Q}^{i+1}_m(x^i_n)$ are handed back to node n. There, a new linear cut can be generated using formula

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \sum_{m \in \mathcal{C}(n)} p_{nm} \underline{Q}^{i+1}_m(x^i_n) + \sum_{m \in \mathcal{C}(n)} (\beta^i_m)^\top (x_n - x^i_n),$$

where $\beta^i_m := -(\pi^i_m)^\top T_m$. This type of cut is often called *Benders cut* because it is constructed in the same way as in Benders decomposition (Benders, 1962). By defining

$$\beta^i_{\mathcal{C}(n)} := \sum_{m \in \mathcal{C}(n)} (\beta^i_m)$$

and

$$\alpha^i_{\mathcal{C}(n)} := \sum_{m \in \mathcal{C}(n)} p_{nm} \underline{Q}^{i+1}_m(x^i_n) - \left(\beta^i_{\mathcal{C}(n)}\right)^\top x^i_n$$

it can also be expressed as

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \alpha^i_{\mathcal{C}(n)} + \left(\beta^i_{\mathcal{C}(n)}\right)^\top x_n.$$

Quantity $\alpha^i_{\mathcal{C}(n)}$ is called the *cut intercept* and $\beta^i_{\mathcal{C}(n)}$ is called the *cut gradient*.

With this cut, the cut approximation $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$ is updated to $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$:

$$\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(x_n) := \max\left\{\mathfrak{Q}^i_{\mathcal{C}(n)}(x_n),\ \alpha^i_{\mathcal{C}(n)} + \left(\beta^i_{\mathcal{C}(n)}\right)^\top x_n\right\}. \tag{2.6}$$

At the end of the backward pass, by solving the first-stage approximate subproblem (2.4) a lower bound $\underline{v}^i$ for the optimal value v^* of MSLP is obtained. If $\overline{v}^i - \underline{v}^i < \varepsilon$ for some predefined tolerance $\varepsilon \geq 0$, then NBD terminates with an (approximately) optimal solution to MSLP. Otherwise, a new iteration is started.

The backward pass is illustrated in Figure 2.4.

An exemplary piecewise linear and convex expected value function $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ and some related cuts are depicted in Figure 2.5.

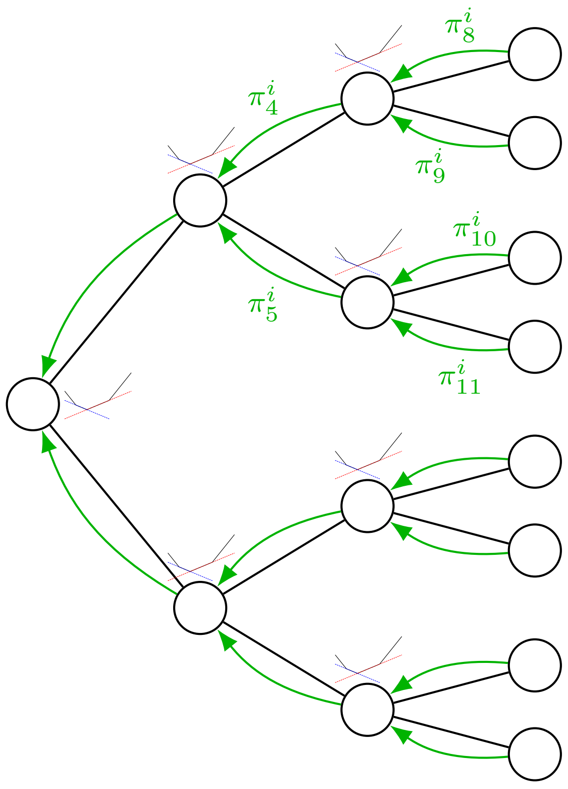


Figure 2.4: Backward pass illustration for NBD.

Recall that the dual feasible set for each node does not change during the algorithm and only possesses finitely many extreme points. Therefore, by restricting to dual extreme points in the cut generation process, only finitely many different Benders cuts can be generated. As these cuts are also valid and tight, finite convergence of NBD can be established (Birge, 1980).

Despite these merits, NBD comes with a significant computational bottleneck. Compared to solving the extensive form of MSLP, there is no need to solve a single problem that grows exponentially in size in T. Nevertheless, solving the DPE may still be computationally intractable for large scenario trees because the number of subproblems to be solved in each iteration grows exponentially in T. For this reason, NBD is only a reasonable solution method for problems with a moderate number of stages. In the next sections, we present SDDP, which can be interpreted as a derivative of NBD that avoids this computational hurdle to a certain degree.

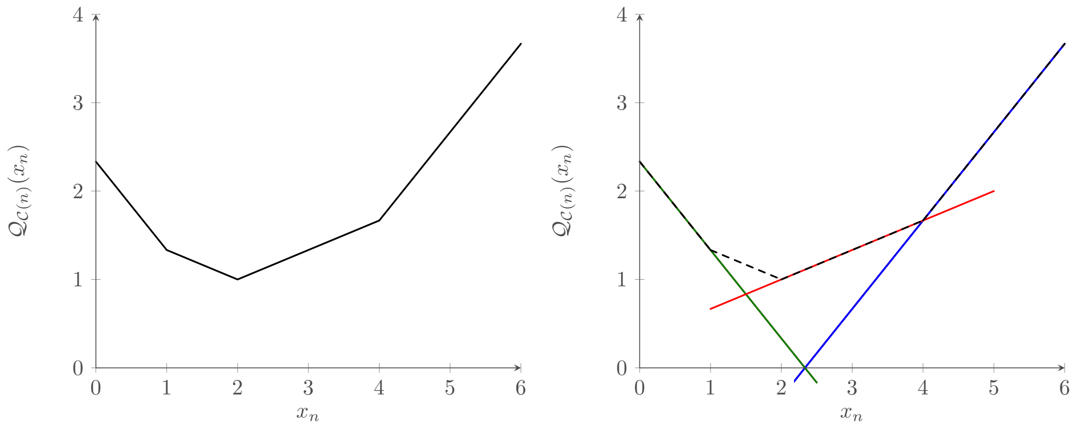


Figure 2.5: Piecewise linear and convex expected value function $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ with three exemplary cuts.

Still, many of our research papers in Part II of this thesis make contributions related to NBD. The reason is that deriving results in a form that suffices NBD is more general, as these results automatically translate to SDDP as well, and thus can be used for both solution methods.

2.8 Stagewise independent uncertainty

We now focus on *stochastic dual dynamic programming* (SDDP) as an alternative solution method for MSLPs. In many practical applications it is the state-of-the-art approach for these kind of problems. A crucial difference between NBD, as presented in the last section, and SDDP is that the latter, at least in its standard form, requires the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ to be *stagewise independent*. This means that for all $t \in [T]$, the random vector $\boldsymbol{\xi}_t$ has to be independent of the history $\xi_{[t-1]}$ of the process. In particular, if modeled using a scenario tree, each node has an equivalent set of successor nodes. This is illustrated in Figure 2.6, where the same color indicates equivalent nodes.

A more condensed and adequate representation of a stagewise independent data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ with discrete and finite random vectors $\boldsymbol{\xi}_t$ for all $t \in [T]$ can be achieved using a *recombining tree* or *scenario lattice*. This is illustrated in Figure 2.7 for the same case as before.

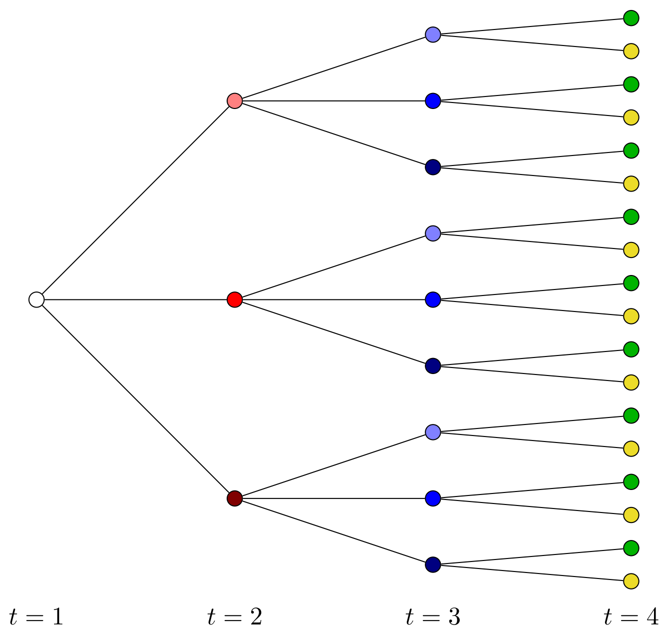


Figure 2.6: Finite scenario tree under stagewise independence. Same colors indicate same realizations and probabilities.

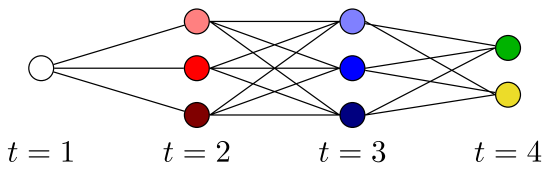


Figure 2.7: Recombining tree under stagewise independence. Same colors indicate same realizations and probabilities.

In the following, when we assume stagewise independence, we denote the realizations of $\boldsymbol{\xi}_t$ by $\xi_{tj}, j = 1, \ldots, q_t$, with $q_t \in \mathbb{N}$, and the associated probabilities by p_{tj}. Importantly, Figures 2.6 and 2.7 show different representations of the same data process, so both trees encode the same number $|\mathcal{S}| = \prod_{t \in [T]} q_t$ of scenarios.

Under stagewise independence, the DPE that we presented in Sect. 2.4 simplify significantly. In particular, there exist only q_t value functions and only one single expected value function for each stage $t = 2, \ldots, T$.

As there exists no nodal dependence, to present the DPE we may revert to using the stage index t again. For any stage $t = 2, \dots, T$ and any realization $\xi_{tj}, j = 1, \dots, q_t$, the DPE read

$$Q_t(x_{t-1}, \xi_{tj}) := \begin{cases} \min\limits_{x_t} & f_t(x_t, \xi_{tj}) + \mathcal{Q}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_{tj}), \end{cases} \tag{2.7}$$

where

$$\mathcal{Q}_{t+1}(x_t) := \sum_{j=1}^{q_{t+1}} p_{t+1,j} Q_{t+1}(x_t, \xi_{t+1,j}).$$

In contrast to before, here the expectation is unconditional. For the first stage, we obtain

$$v^* = \begin{cases} \min\limits_{x_1} & f_1(x_1) + \mathcal{Q}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases}$$

Analogously, also the approximate subproblems simplify. For instance, for $t = 2, \dots, T$ and any realization $\xi_{tj}, j = 1, \dots, q_t$, they become

$$\underline{Q}_t^i(x_{t-1}^i, \xi_{tj}) := \begin{cases} \min\limits_{x_t} & f_t(x_t, \xi_{tj}) + \mathfrak{Q}_t^i(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^i, \xi_{tj}). \end{cases} \tag{2.8}$$

2.9 Stochastic dual dynamic programming

In this section, we present SDDP in its standard form, as proposed by Pereira and Pinto (1991). In general, the functional principle of SDDP is very similar to NBD. As for NBD, each iteration consists of a forward pass and a backward pass through the scenario tree, which due to stagewise independence can be modeled as a recombining tree, though.

The main difference is that in contrast to considering all scenarios $s \in \mathcal{S}$ in each iteration, only a subset $\mathcal{K} \subseteq \mathcal{S}$ is sampled. With respect to Figures 2.3 and 2.4 this means that only a subset of paths through the tree is considered in each iteration. Usually $|\mathcal{K}|$ is chosen much smaller than $|\mathcal{S}|$, in many implementations even $|\mathcal{K}| = 1$ is standard. This sampling does not only reduce the computational effort in the forward pass, but especially in the backward pass where only $1 + |\mathcal{K}| \sum_{t=2}^{T} q_t$ subproblems have to be solved. This number is linear in T, whereas the total number of scenarios $|\mathcal{S}|$ grows exponentially in T (Rebennack, 2016). Note that based on this sampling step, the forward pass is often referred to as a *forward simulation*.

Without stagewise independence, sampling reduces the computational effort *per iteration*, but may as well lead to a higher number of required iterations until convergence because not all nodes of the scenario tree are visited in each iteration. In contrast, under stagewise independence, the uncertainty can be modeled by a recombining tree and there exists only one set of value functions and one expected value function per stage. This means that even if only a sample of scenarios is visited per iteration, still all nodes are visited and all cut approximations are updated in each iteration. This is crucial for the performance of SDDP.

The property that cuts derived for a specific sampled scenario $k \in \mathcal{K}$ are also valid for all other scenarios $s \in \mathcal{S}$ is often referred to as *cut-sharing* in the literature (Infanger and Morton, 1996). This makes sense from the perspective of a classical scenario tree, see Figure 2.6, where cut coefficients are derived for nodes in a specific sample path, but are also valid for all equivalent nodes in other parts of the tree. From the perspective of a recombining tree, cuts are shared among scenarios because these scenarios share the same nodes in the recombining tree, and thus the same value functions.

We present a pseudo-code of SDDP in Algorithm 1.

We should address two more important topics with respect to SDDP. First, due to the sampling, in contrast to NBD no valid upper bound for v^* is computed in each iteration. By evaluating the trial points obtained for the sampled scenarios in the objective function, *i.e.*, computing

$$\overline{v}^i_{\mathcal{K}} := \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \sum_{t=1}^{T} c_t(\xi_t^k)^\top x_t^{ik}, \tag{2.9}$$

we only obtain an unbiased estimator of the true upper bound $\overline{v}^i$. This is also referred to as a *statistical upper bound*. In particular, $\overline{v}^i_{\mathcal{K}}$ is not guaranteed to exceed v^*, it may even fall below $\underline{v}^i$.

A direct consequence of this aspect is that the stopping criterion of NBD cannot be carried over to SDDP. Instead, different stopping criteria have been proposed for SDDP. Pereira and Pinto (1991) initially suggested to stop SDDP as soon as

Algorithm 1 Standard SDDP

```
1: Initialize cut approximations $\mathfrak{Q}_t^1(\cdot)$ for all $t = 2, \dots, T$.
2: Initialize lower bound with $\underline{v}^0 = -\infty$.
3: Set iteration counter to $i \leftarrow 0$.
4: while Stopping criterion not satisfied do
5:     Set $i \leftarrow i + 1$.
6:     Sample a subset $\mathcal{K} \subseteq \mathcal{S}$ of scenarios.        ▷ Forward pass
7:     for stages $t \in [T]$ do
8:         for samples $k \in \mathcal{K}$ do
9:             Solve the stage-$t$ subproblem (2.8) associated with $\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k)$ to
               obtain trial point $x_t^{ik}$.
10:        end for
11:    end for
12:    If required, compute an upper bound estimate $\overline{v}_{\mathcal{K}}^i$ according to (2.9).
13:    for stages $t = T, \dots, 2$ do        ▷ Backward pass
14:        for samples $k \in \mathcal{K}$ do
15:            for realizations $j = 1, \dots, q_t$ do
16:                Solve the stage-$t$ subproblem (2.8) associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_t^k)$.
                   Store the optimal value and dual vector $\pi_t^{ikj}$.
17:            end for
18:            Use relation (2.7) to create a new cut for $\mathcal{Q}_t(\cdot)$.
19:            Update the cut approximation $\mathfrak{Q}_t^i(\cdot)$ to $\mathfrak{Q}_t^{i+1}(\cdot)$ using relation (2.6).
20:        end for
21:    end for
22:    Solve the first-stage subproblem to obtain a lower bound $\underline{v}^i$.
23: end while
```

the lower bound $\underline{v}^i$ is contained in the confidence interval that can be derived for $\overline{v}_{\mathcal{K}}^i$. However, this approach has several flaws, such as incentivizing premature stopping (Shapiro, 2011). Therefore, in practice, usually more pragmatic criteria are used, *e.g.*, stopping after a finite number of iterations or cuts, or when the lower bounds $\underline{v}^i$ show no significant improvement over several iterations.

Second, for small $|\mathcal{K}|$, the upper bound estimate (2.9) has almost no explanatory power with respect to the true upper bound $\overline{v}^i$, and thus the quality of the identified policy. To evaluate the policy obtained in SDDP, in practice therefore often an additional forward simulation is conducted after SDDP has terminated. For this simulation a much higher number of sample paths through the scenario tree is used, *e.g.* $|\mathcal{K}| \in \{1000, 10000\}$, leading to a reasonable estimator $\overline{v}_{\mathcal{K}}$.

In the presented form, and given that an *independent random sampling* procedure is used in line 6 of Algorithm 1, SDDP can be shown to converge with probability 1 to an optimal policy of MSLP in a finite number of iterations (almost sure finite convergence) (Philpott and Guan, 2008).

Despite its computational advantages compared to NBD and performing well for several MSLPs in practice, SDDP is still computationally expensive in the worst-case, though. Its worst-case complexity is only polynomial in q_t, but exponential in T and the state dimension d_t (Lan, 2022; Zhang and Sun, 2022).

2.10 Expanding the state space

As pointed out before, a crucial requirement for the functioning of standard SDDP is stagewise independence of the uncertainty in the data. In many practical applications, this assumption is not satisfied. For this reason, for cases of stagewise dependent uncertainty, various modifications of SDDP are proposed in the literature. One approach is to extend SDDP to Markov chain uncertainty, which can also be represented by a scenario lattice. The idea is then to approximate the occurring stagewise dependent uncertainty by a Markov chain, and thus make it applicable to SDDP (Löhndorf and Shapiro, 2019). Other approaches are combining SDDP with an underlying Markov chain (Philpott et al., 2013), using conditional cuts (van Ackooij and Warin, 2020), using saddle cuts (Downward et al., 2020) or dual variants of SDDP (Guigues et al., 2023). The last two approaches are specifically suited for stagewise dependent uncertainty occurring in the objective function.

Often, the uncertainty can also be modeled by an AR process. A prominent example is the modeling of hydro inflows into reservoirs in power system applications (de Matos and Finardi, 2012). As long as the AR process is linear, SDDP can be applied in a straightforward way by increasing the dimension of the state variables. This is why this approach is mostly known as *expanding the state space.* We illustrate this using a simple process of form

$$\xi_t = \phi_t \xi_{t-1} + \eta_t. \tag{2.10}$$

Here, ϕ_t is a vector of autoregressive coefficients and η_t are realizations of a stagewise independent random variable $\boldsymbol{\eta}_t$ representing the error term.

The key idea now is to consider (x_{t-1}, ξ_{t-1}) as the system state *i.e.*, interpreting ξ_{t-1} as an additional state variable (Shapiro et al., 2013). By doing this and adding equation (2.10) to the problem formulation (2.7), for any $t \in [T]$ and any realization $j = 1, \dots, q_t$ of $\boldsymbol{\eta}_t$, we obtain value functions

$$Q_t(x_{t-1}, \xi_{t-1}, \eta_{tj}) = \begin{cases} \min\limits_{x_t, \eta_t} & f_t(x_t, \xi_{tj}) + \mathcal{Q}_{t+1}(x_t, \xi_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \\ & \xi_t = \phi_t \xi_{t-1} + \eta_{tj}. \end{cases}$$

This approach manages to reformulate the problem in such a way that the remaining uncertain data exhibits stagewise independence, and thus cuts can be naturally derived as functions in x_{t-1} and ξ_{t-1}, and shared between scenarios. In return, the state and decision space dimensions increase. It is important to note, however, that it is not necessarily required to add the AR process formula (2.10) to the constraints explicitly. It may also be evaluated outside of the subproblems to obtain ξ_t and taken into account in the cut generation process to derive tailor-made cut intercept formulas, which allow cuts to be adapted to a specific history ξ_{t-1} (Infanger and Morton, 1996; Rebennack, 2016).

2.11 Stochastic dual dynamic integer programming

One of the key ingredients of NBD and SDDP as solution methods for MSLPs is the approximation of (expected) value functions with *tight* linear cuts that are constructed using LP duality. As we have seen, after finitely many steps, an exact representation of the value functions is possible. When some or all components of the state variable x_t have to satisfy integer requirements, *i.e.*, the problem at hand is a *multistage stochastic mixed-integer linear program* (MS-MILP), the previously presented cut generation approach is not sufficient to ensure convergence to an optimal solution.

First, the subproblems in the DPE (2.7) occurring in SDDP are *mixed-integer linear programs* (MILPs) instead of LPs. Therefore, the approach of using LP duals is not applicable. It is possible to derive Benders cuts using the duals of the LP *relaxation* of subproblems (2.7), but these cuts are not guaranteed to be tight for $\mathcal{Q}_t(\cdot)$. Second, even if Lagrangian duality is used as an alternative that applies to MILPs, the strong duality property that is required for tightness is not guaranteed to hold. Third, and related to second, in contrast to the continuous linear case, the value functions $Q_t(\cdot)$ and expected value functions $\mathcal{Q}_t(\cdot)$ are not guaranteed to be piecewise linear and convex. Therefore, approximating them by linear cuts that are both *tight and valid* may not be possible. We discuss this in more detail in Chapter 3.

Still, some extensions of SDDP to MS-MILPs have been put forward that come with convergence guarantees. The most prominent one is SDDiP (Zou et al., 2019b).

Here, to generate cuts, first a copy variable z_t for the current state x_{t-1}^i is introduced together with some constraint set Z_t. It allows to replace x_{t-1}^i by a decision variable in the original constraints. The subproblems become

$$\underline{Q}_t^i(x_{t-1}^i, \xi_{tj}) = \begin{cases} \min\limits_{x_t, z_t} & f_t(x_t, \xi_{tj}) + \mathfrak{Q}_t^i(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(z_t, \xi_{tj}) \\ & z_t = x_{t-1}^i \\ & z_t \in Z_t. \end{cases}$$

Then, a Lagrangian relaxation is considered where the copy constraint is relaxed. This yields the inner problem

$$\mathcal{L}_t^{i+1}(\pi_t) := \begin{cases} \min\limits_{x_t, z_t} & f_t(x_t, \xi_{tj}) + \mathfrak{Q}_t^i(x_t) - \pi_t^\top z_t \\ \text{s.t.} & x_t \in \mathcal{X}_t(z_t, \xi_{tj}) \\ & z_t \in Z_t \end{cases}$$

for some dual multipliers $\pi_t \in \mathbb{R}^{d_{t-1}}$, and the outer (Lagrangian dual) problem

$$\max_{\pi_t} \ \mathcal{L}_t^{i+1}(\pi_t) + \pi_t^\top x_{t-1}^i. \tag{2.11}$$

By solving the outer problem (2.11), and using the optimal dual multipliers π_t^i and the value $\mathcal{L}_t^{i+1}(\pi_t^i)$, a valid cut for $\mathcal{Q}_t(\cdot)$ can be derived as

$$\mathcal{Q}_t(x_{t-1}) \geq \mathcal{L}_t^{i+1}(\pi_t^i) + (\pi_t^i)^\top x_{t-1}.$$

As shown in Zou et al. (2019b), when the state variables are pure binary, *i.e.*, $x_t \in \{0,1\}^{d_t}$, and the set Z_t is chosen as $Z_t = \{0,1\}^{d_{t-1}}$ or $Z_t = [0,1]^{d_{t-1}}$, these cuts are also tight in the sense defined before. Therefore, the same convergence properties hold as for SDDP.

In cases where the state variables are not binary, but satisfy some bound constraints $x_{tj} \in [0, U_j]$, with $U_j \in \mathbb{R}$, for all components $j = 1, \ldots, d_t$, it is proposed to replace them with a finite number of binary state variables and to solve the obtained approximation of the original MS-MILP (Zou et al., 2019b). More precisely, let $\beta_{tj} \in (0,1]$ be some grid precision. Then, using additional binary variables $\lambda_{tkj}, k = 1, \ldots, K_{tj}$, for $K_{tj} = \lfloor \log_2(U_j) \rfloor + 1$ K_{tj}, x_{tj} can be replaced by the quantity $\sum_{k=1}^{K_{tj}} 2^{k-1} \beta_{tj} \lambda_{tkj}$.

If $x_{tj} \in \mathbb{Z}$ for some component j, this representation can be made exact for $\beta_{tj} = 1$, if $x_{tj} \in \mathbb{R}$, then only an approximation is possible, with the quality controlled by the precision β_{tj}.

Chapter 3

Research objectives

In this chapter, we motivate the research objectives and topics that are studied in this thesis.

3.1 The state of research on SDDP

In its standard form SDDP is a rather simple algorithm, only consisting of a forward simulation and a backward pass through a recombining scenario tree, iteratively generating cuts for the expected value functions; see Sect. 2.9. However, since its invention it has gained enormous interest in research from a theoretical perspective, with respect to computational improvements, and in practice where it is applied to numerous case studies and to this date is one of the state-of-the-art algorithms to tackle large-scale MSLPs. Therefore, it has developed into a wide-ranging research area.

From a theoretical perspective, standard SDDP comes with a variety of necessary assumptions, among them linearity of all involved functions, no integer constraints, relatively complete recourse, discrete and finite random variables $\boldsymbol{\xi}_t$, and crucially, stagewise independence of the data process $(\boldsymbol{\xi}_t)_{t \in [T]}$. Many of these assumptions may not be satisfied in practice. For instance, many applications require integer constraints, *e.g.*, to model investment decisions in generation expansion problems or start-up or shut-down constraints in unit commitment problems. As another example, hydro inflows into reservoirs usually show a temporal correlation, and thus may not be modeled appropriately assuming stagewise independence. For these reasons, there has been a legitimate interest in relaxing some of the standard assumptions and making SDDP applicable to a broader class of problems.

Finally, SDDP has been observed to show slow convergence, with lower bounds stalling prematurely for some practical problems (Ávila et al., 2024). As a consequence, various acceleration techniques have been proposed.

Due to the sheer amount and the variety of proposed extensions, it becomes increasingly difficult to keep track of the state of research on SDDP. Therefore, the first research objective of this thesis is to shed light on SDDP as a research field, exploring existing extensions, their strengths and weaknesses as well as giving an outlook on future research on SDDP.

Research objective 1. Provide a comprehensive survey and review of SDDP as a research field.

3.2 Non-convex value functions

In Papers B to E, we focus on one specific class of challenges with respect to SDDP.

Previously, we have pointed out that a crucial property for the functioning of NBD and SDDP is that the (expected) value functions are piecewise linear and convex functions in the state variables. This prerequisite allows a valid, tight and finite approximation of these functions using Benders cuts, meaning that after finitely many of these linear cuts the original functions can be reproduced. This is important to establish the known convergence results for both solution methods.

NBD and SDDP can be enhanced to nonlinear *convex* problems, for which the (expected) value functions remain convex, but are no longer guaranteed to be piecewise linear, without changing the cut generation mechanism (Girardeau et al., 2015; Guigues, 2016). However, for many practical applications this generalization is not sufficient. Instead, many multistage decision-making problems in practice can only be modeled appropriately by MSPs which, if reformulated as DPE (2.7), lead to *non-convex* (expected) value functions. These value functions cannot be tightly approximated from below using linear cuts in general. Either *valid* linear cuts can be derived that approximate them from below, but lack the tightness property, or *tight* affine functions can be derived that may violate validity.

The main research question of this thesis is how SDDP (and related decomposition methods such as NBD) can be enhanced to these kind of problems.

Research objective 2. Develop approaches to deal with non-convex (expected) value functions and achieve tight approximations in SDDP and related decomposition methods.

We specifically deal with three different and prominent cases in which non-convex value functions may occur.

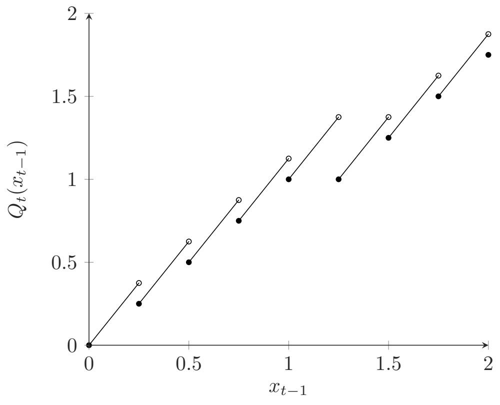
- **Case 1: Integer variables.** If the subproblems contain mixed-integer variables, then even if all occurring functions are linear, the (expected) value functions are not guaranteed to be convex. In fact, they are not even guaranteed to be continuous. This is illustrated in Figure 3.1a.

- **Case 2: Non-convex functional description.** If the subproblems contain non-convex functions in the objective or nonlinear functions in the constraints, such that the feasible set $\mathcal{X}_t(x_{t-1}, \xi_t)$ is non-convex, then even if no integer variables are present, the (expected) value functions become nonlinear non-convex. This is illustrated in Figure 3.1b.

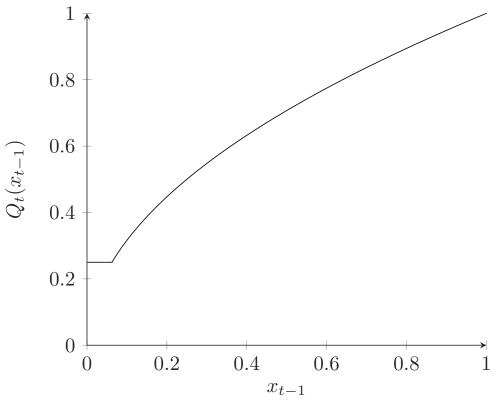
- **Case 3: Non-convex stagewise dependent uncertainty.** Recall the linear AR process (2.10) to model the uncertain data in MSLP. Now assume that this AR process is more generally defined by $\xi_t = b_t(\xi_{[t-1]}, \eta_t)$ for some non-convex function $b_t(\cdot)$. By using the expanding-the-state-space approach and adding this AR model equation to the subproblem constraints, the feasible set becomes non-convex in the new state variable $\xi_{[t-1]}$. Thus, we are back to Case 2. Even if the model equation is only considered implicitly when evaluating $\boldsymbol{\xi}_t$ and when generating cuts, the (expected) value functions can be shown to be non-convex in $\xi_{[t-1]}$ in general.

Clearly, also combinations of these three cases are possible. This is illustrated in a venn diagram in Figure 3.2. It is also highlighted which case or combination of cases is studied in which research paper in Part II of this thesis.

We discuss the motivation for studying these specific cases in more detail below and also put forward more specific research objectives for each of them.



(a) Mixed-integer linear subproblem.



(b) Nonlinear non-convex subproblem.

Figure 3.1: Examples of non-convex value functions.

Before we do this, we take a general perspective on dealing with non-convex value functions in MSP. In principle, three different approaches can be taken.

- **Approach I: Using non-convex cuts.** The idea is to approximate the non-convex (expected) value functions with non-convex functions, which we also refer to as *cuts* for convenience. In contrast to linear cuts, if chosen appropriately, these cuts can be tight without compromising validity.
- **Approach II: Using linear cuts under tightness-ensuring conditions.** Here, linear cuts are generated irrespective of the non-convexity of the value functions. However, to achieve tightness and convergence guarantees, certain conditions have to be met, *e.g.* binary state variables x_{t-1} in SDDiP. Applying this approach may also involve a reformulation of the original problem.
- **Approach III: Using linear cuts without tightness.** Similarly to Approach II valid linear cuts are generated, but without tightness-ensuring conditions. This leads to a heuristic version of SDDP or related decomposition methods without convergence guarantees. However, it may still yield satisfactory policies in practical applications, where decomposition methods are often terminated before convergence is achieved anyway.

We should mention that in two-stage stochastic programming a fourth approach is common where convergence can be guaranteed by branching on the first-stage

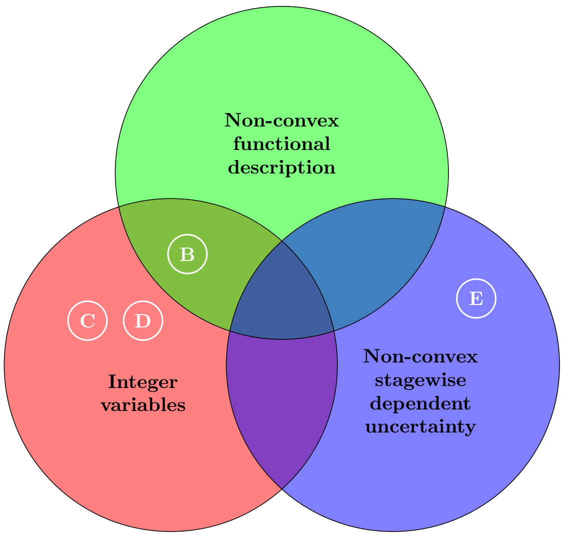

Non-convex functional description
B
C
D
E
Integer variables
Non-convex stagewise dependent uncertainty

Figure 3.2: Venn diagram of properties of MSP which lead to non-convex (expected) value functions. Circles with letters indicate which case is studied in which research paper in Part II of this thesis.

Table 3.1: Classification of research papers in this thesis.

	Studied approximation approach		
Studied case	Approach I	Approach II	Approach III
Case 1	B and C	C and D	C and D
Case 2	B		
Case 3	E		

variables, even if non-tight cuts are used. For a multistage setting this approach is not applicable, though, as it requires branching in all nodes but the leaf nodes, which is computationally too expensive.

In this thesis, we are interested in solution methods with convergence guarantees, so we mainly focus on Approaches I and II. However, all the results presented for Approach II may also be applied heuristically when pursuing Approach III. Table 3.1 classifies our research papers (see Part II of this thesis) with respect to the cases of non-convexity and the chosen approach to address this uncertainty.

3.2.1 Using non-convex cuts for MS-MILPs

As described in Sect. 2.11, if all state variables x_{t-1} in an MS-MILP are binary, SDDiP is guaranteed to converge to an optimal policy in finitely many steps almost surely. It only differs from SDDP in the aspect that special Lagrangian cuts are generated instead of classical Benders cuts. If the state variables are not binary, it is proposed to approximate them with binary variables. Whereas this approximation can be made exact for bounded integer state variables, this is not the case for generally continuous state variables. Nonetheless, Zou et al. (2019b) show that given the feasibility assumption of *complete continuous recourse*, which is considerably stronger than relatively complete recourse, but ensures Lipschitz continuity of the value functions, the problem can be solved to arbitrary precision by choosing a sufficiently large number of binary variables.

However, this approach comes with some drawbacks. The assumption of complete continuous recourse is rather strong and not necessarily satisfied in applications. More crucially, identifying a sufficiently high number of binary variables is difficult in practice, as it requires knowledge of problem-specific constants. Therefore, a suitable binarization precision has to be estimated in advance. If it is chosen too coarse, then the solved problem may strongly deviate from the original MS-MILP. If it is chosen very finely, then the state space may become unnecessarily large, negatively affecting the computational performance.

These observations create the demand for a dynamic version of SDDiP where the binary approximation is refined dynamically if required throughout the solution process. Importantly, previously generated cuts should remain valid in order to avoid that the approximations of $\mathcal{Q}_t(\cdot)$ have to be started from scratch.

Research objective 2.1. Develop an extension of SDDiP where the state binarization is dynamically refined if required without destroying the validity of previously generated cuts.

A second objective that is naturally arising in the context of non-convex value functions is to explore if some tight non-convex approximations of the value functions could be derived.

Research objective 2.2. Develop a method to generate tight non-convex cuts for the value functions that are practically applicable.

Interestingly, as we shall see in Sect. 4.3, pursuing research objective 2.1 naturally leads to non-convex cuts, thus also addresses research objective 2.2.

During the work on this project, some first solution methods that make use of non-convex cuts in MS-MILPs have already been proposed (Ahmed et al., 2022; Zhang and Sun, 2022). Basically, both methods generate cuts based on solving *augmented Lagrangian dual* problems. Even with these methods already published, working on the above research objectives was continued, but taking the new insight into account. In Sect. 4.3 and 4.4, we explain in detail the relations and differences between our research results and the two prior proposals.

3.2.2 Using non-convex cuts for general MS-MINLPs

With SDDiP a decomposition method exists that extends SDDP to the broader problem class of MS-MILPs. A natural follow-up research question is how to further extend it to a general non-convex *multistage stochastic mixed-integer nonlinear program* (MS-MINLP), which also contains non-convex functions in the objective and constraints. Being able to handle this type of problem is practically relevant, as some effects in applications can only be modeled appropriately by nonlinear functions, *e.g.*, the valve-point effect of thermal generators (Pedroso et al., 2014) or the water head effect in hydro reservoirs (Cerisola et al., 2012).

However, these types of problems pose the additional challenge that all subproblems are nonlinear problems, and thus notoriously harder to solve than LPs or MILPs. Therefore, previous attempts at generalizations of SDDP in this direction either used convexifications and linear cuts (Cerisola et al., 2012; Steeger and Rebennack, 2017) or required specific assumptions to be satisfied, for instance monotonicity of the value functions (Philpott et al., 2020). We try to come up with a more general method.

Research objective 2.3. Develop a method that extends SDDP or NBD to non-convex *mixed-integer nonlinear programs* (MINLPs).

Again, we have to mention the work by Zhang and Sun (2022), which was published during this research project and qualifies for research objective 2.3. We contrast it with our work and highlight our unique contributions in Sect. 4.4.

3.2.3 Using linear cuts for MS-MILPs

In addition to our work on tight non-convex approximations, we also try to contribute to improving SDDiP and the generation of linear Lagrangian cuts. These types of cuts are still relevant in solving MS-MILPs because the generation and incorporation of non-convex cuts may require excessive computational resources.

A main computational challenge for SDDiP itself is that solving Lagrangian dual problems in each iteration requires significant time, which is even aggravated if a state binarization is applied. This is already pointed out in the original SDDiP work (Zou et al., 2019b), and as a remedy Lagrangian cuts are combined with cheaper cuts, such as (strengthened) Benders cuts.

Moreover, despite their favorable, convergence-ensuring tightness properties, Lagrangian cuts may not always yield the best possible approximations of the (expected) value functions in practice. Sometimes there may exist alternative cuts, possibly not tight at x_{t-1}^i, that may significantly speed-up the convergence process due to improving the approximation outside of x_{t-1}^i.

Finally, especially when a state binarization is applied and cuts are only generated at extreme points of the state space, the dual problems are often degenerate, so that infinitely many different Lagrangian cuts may be generated, which may drastically differ in their approximation quality outside of x_{t-1}^i.

Given these challenges, our amibition is to improve the computational performance of SDDiP, either by generating the standard Lagrangian cuts faster or by generating different Lagrangian cuts with computationally preferable properties.

Research objective 2.4. Study possible modifications of the generation process for Lagrangian cuts in MS-MILPs in order to improve the computational performance of SDDiP or related methods relying on these cuts.

3.2.4 Dealing with non-convex stagewise dependent uncertainty

As mentioned before, if we allow the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ to be stagewise dependent and to be described by a non-convex AR process, the (expected) value functions occurring in SDDP are no longer guaranteed to be convex in the history $\xi_{[t-1]}$, which serves as a new state variable.

Due to this challenge, so far extensions of SDDP to AR processes have been mostly limited to different types of linear processes (Guigues, 2014; Infanger and Morton, 1996; Lohmann et al., 2016; Queiroz and Morton, 2013). An exception is the work by Guigues (2014) where convex AR processes are considered. In this case the (expected) value functions remain convex, though, so linear cuts can be derived to tightly approximate them. Another exception are special nonlinear processes studied by Infanger and Morton (1996) which still satisfy some linearity or additivity properties.

However, non-convex AR processes are appropriate models in many practical applications. For instance, log-linear AR processes (also called *geometric* AR processes) are prominently used to model hydro inflows in hydrothermal scheduling problems (de Matos and Finardi, 2012; Lohmann et al., 2016; Shapiro et al., 2013), where the requirement of non-negativity cannot be satisfied by linear AR processes.

This example shows that an extension of SDDP to more general AR processes is practically relevant. Currently, these processes either have to be linearized, or users have to revert to other modeling techniques, *e.g.* using Markov chain SDDP (Löhndorf and Shapiro, 2019). Our research goal is to adapt SDDP to being able to handle more general AR processes algorithmically. Importantly, the focus is on deriving closed-form cut formulas given these type of processes, so that the cut generation and evaluation remains computationally tractable.

Research objective 2.5. Extend SDDP to more general, especially non-convex classes of AR processes, such that the cut generation and evaluation remains computationally tractable.

Chapter 4

Research studies and contributions

The following sections summarize the main contributions of this thesis. For detailed descriptions of the methodology and the results, the reader is referred to the research papers in Part II.

4.1 A review on SDDP

In Paper A we provide a comprehensive tutorial-type review on SDDP as a research field. The review comprises a detailed introduction to the main concepts behind SDDP, an overview on the variety of existing algorithmic extensions and modifications of SDDP, a discussion of its strengths and weaknesses as well as an outlook on future research directions. The review is addressed to a broad audience: researchers in the stochastic programming community working on SDDP (or related methods) that try to get a clear picture of the current state of the research; practicioners that want to apply SDDP to real-world problems; or novices to SDDP, *e.g.*, scientists from different research communities, that look for an accessible introduction to the research area.

The review is divided into four parts. Part I explains the main mechanism behind SDDP. First, some relevant preliminaries are introduced and then the algorithm is presented in its basic form. In this part, special attention is directed to 9 assumptions that we identified to be crucial for the functioning of SDDP. In addition, its convergence properties and its complexity are discussed in detail.

Also, we point out the differences between SDDP and related solution methods such as NBD or stochastic dynamic programming.

Part II summarizes applications and implementations of SDDP.

Part III then discusses how the previously identified key assumptions can be relaxed when working with SDDP. Among others, this includes dealing with continuous random variables, distributional uncertainty, risk-aversion, stagewise-dependent uncertainty or non-convexities.

Part IV presents a multitude of approaches to improve the computational performance of SDDP. The motivation behind this is that it is observed to converge very slowly for some types of problems in practice (Ávila et al., 2024).

Finally, we present an outlook on future research on SDDP. We identify 7 sub-areas of research on SDDP that should deserve special attention. Particularly noteworthy is the potential of borrowing successful techniques from reinforcement learning, which shares many similarities with multistage stochastic programming, and incorporating them into SDDP.

4.2 The role of copy constraints and Lipschitz regularization

Dealing with SDDP-related decomposition methods in the context of integer variables often makes use of Lagrangian-type relaxations to generate cuts instead of the weaker LP relaxations. This is the case for SDDiP (Zou et al., 2019b), but also for the non-convex cuts in (Ahmed et al., 2022; Zhang and Sun, 2022), which are generated using *augmented* Lagrangian dual problems. Also our own work heavily relies on these ideas.

A second important concept from the literature that we rely on is Lipschitz regularization, which we borrow from (Zhang and Sun, 2022). Lipschitz regularization of subproblems is helpful in allowing to work with Lipschitz continuous value functions even without strong recourse assumptions.

Applying Lagrangian relaxation and Lipschitz regularization in our own work, and extending their usage beyond known results, requires a profound understanding of both techniques. Therefore, in Sect. 3-4 of Paper C, we deeply analyze them and provide new theoretical insight on their main properties and how they relate to each other. In particular, we investigate in detail the effects that copy constraints and Lipschitz regularization have on the generation and the properties of Lagrangian cuts.

With respect to the role of copy constraints $z_t = x_{t-1}^i$, we show that accompanying the new copy variables z_t with additional constraints $z_t \in Z_t$ leads to Lagrangian cuts with different properties depending on the choice of Z_t. In particular, we highlight that the tightness result from SDDiP (Zou et al., 2019b) is actually based on the choice of appropriate sets Z_t and not on the introduction of copy constraints itself.

With respect to Lipschitz regularization, we generalize some existing results on Lagrangian relaxation. It is well-known that solving the Lagrangian dual problem to a subproblem is equivalent to solving a specific convexification of said subproblem, and that the optimal value of both problems coincides with the value of the closed convex envelope of the value function $Q_t(\cdot)$ at x_{t-1}^i. Furthermore, it is well-known that Lipschitz regularizations in the primal space are directly linked to norm bounds in the dual space. We manage to combine these results: We show that given some fixed norm $\|\cdot\|$ and some parameter σ_t for the Lipschitz regularization, solving a norm-bounded Lagrangian dual problem of the stage-t subproblem, where the dual norm to $\|\cdot\|$ is used, is equivalent to solving a specific convexification of the Lipschitz regularized subproblem, and that the optimal value of both problems coincides with the value of the closed convex envelope of the *regularized value function* $Q_t^R(\cdot; \sigma_t\|\cdot\|)$ at x_{t-1}^i.

We use this novel result to show that Lagrangian cuts obtained from norm-bounded dual problems are tight for the closed convex envelope of the regularized value functions $Q_t^R(\cdot; \sigma_t\|\cdot\|)$. As long as all state variables are binary, similar to SDDiP, tightness for the true regularized value functions can be achieved, given that σ_t is sufficiently large or that $\|\cdot\|$ is chosen as the 1-norm. This generalizes the tightness result from Zou et al. (2019b) to the Lipschitz regularized case.

4.3 Dynamic SDDiP

With respect to research objective 2.1, we propose a method called *Dynamic SDDiP*, which enhances SDDiP such that the binary approximation precision can be refined *within* the solution method, while not compromising the validity of previously generated cuts.

Importantly, this contribution is not related to a single paper, but to Papers B and C. In Paper B, a larger decomposition framework is introduced, see Sect. 4.4, and its inner loop can be interpreted as a dynamic version of SDDiP, even though it is only applied to a deterministic problem there. Based on some technical assumptions, also a convergence proof is presented. In Sect. 5 of Paper C, these algorithmic ideas are extended to the stochastic case. Moreover, a more rigorous theoretical analysis is provided. In particular, it is shown how convergence can be achieved when the technical assumptions from Paper B are dropped.

4.3.1 Theoretical results

We present Dynamic SDDiP in more detail now. The first main difference to SDDiP is that no *static* binary approximation of the state space is applied. Instead, the solution method operates in the original state space, which implies that the original MS-MILP is solved, and that all cuts are expressed in the original state variables x_{t-1}.

In order to obtain *tight* cuts, the tightness result from SDDiP is exploited, stating that Lagrangian cuts are tight if the state variables are binary. To this end, a *temporary* lifting to the binary space is conducted (the associated precision is refined if the solutions in the forward pass do not improve for a predefined number of iterations). Then, tight Lagrangian cuts are computed in this binary space. In order to use them in the original subproblems, these cuts are projected back to the original state space. This procedure is crucial to ensure that the cuts remain valid, even if the precision of the binarization is refined later on. This lift-and-project idea is illustrated in Figure 4.1, which is taken from Paper C.

We now address the projection of cuts. In Paper C, we coin the pointwise maximum of this projection the *cut projection closure* (CPC). Importantly, the CPC is a non-convex function in the original state variables x_{t-1}, thus a *non-convex cut* for the non-convex expected value function $\mathcal{Q}_t(\cdot)$. This means that addressing research objective 2.1 implicitly leads to addressing research objective 2.2 as well. To make sure that the subproblems remain MILPs, the CPC can be expressed through linear constraints by introduction of additional (binary) variables.

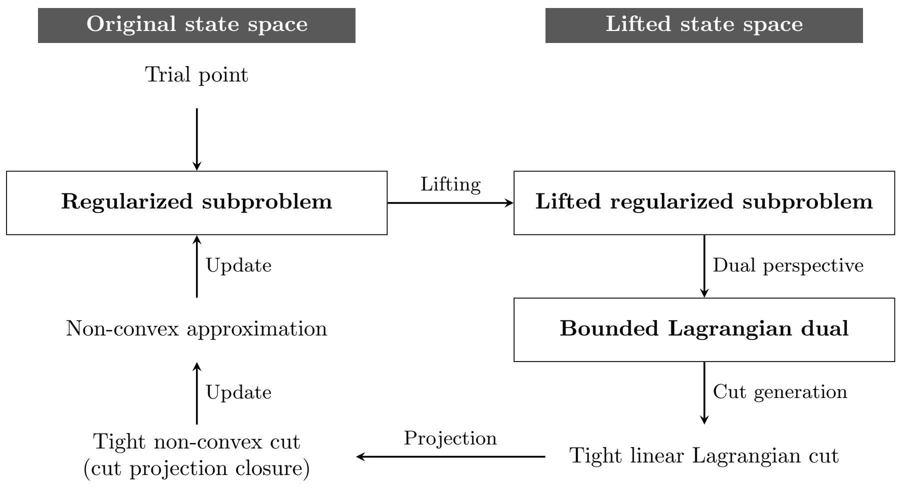


Figure 4.1: The lift-and-project cut generation approach from Dynamic SDDiP.

For the convergence of Dynamic SDDiP, it is required that the CPCs possess certain properties: they should be valid, tight and Lipschitz continuous, cf. Sect. 2.5. We call such CPCs *sufficient*.

Under certain assumptions, the CPC can be shown to be tight at the so-called *anchor point* $x^i_{\mathbb{B};t-1}$. In some sense, this is the point closest to the trial point x^i_{t-1} which can be described exactly using the current state binarization. The distance between $x^i_{\mathbb{B};t-1}$ and x^i_{t-1} can be controlled using binary refinements. The idea is that under Lipschitz continuity of the CPC, then also the error in the cut approximation can be controlled, thus leading to some notion of tightness at x^i_{t-1} as well. However, in order for this to be true, it has to be ruled out that with a binarization refinement also the Lipschitz constant of the CPC increases. In other words, the CPC has to be Lipschitz continuous with a constant independent of the binarization precision. Otherwise, the CPCs may become infinitely steep and convergence is not guaranteed.

To ensure Lipschitz continuity of the CPCs, it is reasonable to consider Lipschitz continuous (expected) value functions within the algorithm. Therefore, we borrow the idea from Zhang and Sun (2022) to apply a Lipschitz regularization with parameter σ_t to the subproblem at stage t. If σ_t is chosen sufficiently large for all $t \in [T]$, still the original MS-MILP is solved (Zhang and Sun, 2022).

In the forward pass of Dynamic SDDiP, Lipschitz regularized subproblems are solved. For the cuts generated in the backward pass, the goals of cut tightness and Lipschitz continuity independent of the binarization precision have to be reconciled. As we show in Paper C, this can be achieved by careful design, that is choosing appropriate weighted norms to bound the Lagrangian dual problem in the lifted binary space.

4.3.2 Comparison with existing approaches

Dynamic SDDiP differs from related methods in the literature. We provide a detailed comparison in the supplementary material to Paper C.

Compared to standard SDDiP there is no need for a static and permanent state binarization nor has the precision of this binarization to be fixed in advance. Instead, the binarization is used only temporarily to ensure tightness of the obtained CPCs, and refined dynamically within the algorithm. Importantly, with these refinements, all previously generated cuts remain valid. This is not possible in the standard SDDiP framework, where refinements shift the value functions of the approximating problem downwards at the risk of existing cuts being violated.

Working in the original state space also comes with the advantage that the original MS-MILP is solved, given that all $\sigma_t, t \in [T]$, are chosen sufficiently large. In contrast, for standard SDDiP with state binarization only an approximation of this problem is solved.

Dynamic SDDiP only requires relatively complete recourse combined with a Lipschitz regularization of the value functions, instead of taking the stronger complete continuous recourse assumption, which may not be satisfied in practice. This approach also has the advantage that all required Lipschitz constants are known at all times.

Compared to SLDP and its generalization in Zhang and Sun (2022), the differences are more subtle. For convergence purposes, all approaches require Lipschitz continuity of the considered value functions to prevent the generated cuts from becoming infinitely steep close to discontinuities. Again, SLDP assumes complete continuous recourse to ensure Lipschitz continuity of the *true* value functions, while Zhang and Sun (2022) suggest to consider Lipschitz regularizations of the value function. We follow the latter approach in combination with relatively complete recourse. In Zhang and Sun (2022) no recourse assumption is taken, but in return the state variables x_{t-1} are only allowed to enter the objective function.

This means that constraints depending on x_{t-1} have to be modeled using indicator functions.

Whereas SLDP solves standard subproblems in the forward pass, in Zhang and Sun (2022) and Dynamic SDDiP Lipschitz regularized subproblems are considered. In the backward pass, SLDP deals with an augmented Lagrangian dual problem without bounds on the dual multipliers. Our proposed method deals with a Lagrangian dual problem with norm bounds in a lifted space. The method from Zhang and Sun (2022) is the most general one, as it includes both an augmenting term in the objective and norm bounds on the dual multipliers.

The proposed Dynamic SDDiP method comes with some drawbacks compared to the other approaches. First, the CPC requires more (binary) variables and constraints to be reformulated using MILP constraints than the cuts from Ahmed et al. (2022). Second, the dual problem has to be solved in a higher-dimensional space due to the state binarization.

On the other hand, Lagrangian dual problems are in general less costly to solve than the augmented dual problems in Ahmed et al. (2022); Zhang and Sun (2022). Moreover, a computational comparison for a simple illustrative example, see the supplementary material to Paper C, indicates that our proposed non-convex cuts may yield better approximations of $\mathcal{Q}_t(\cdot)$ at regions different from the trial point x_{t-1}^i. In particular, the Lipschitz cuts from Ahmed et al. (2022) may be unnecessarily steep, as their slope is pre-determined. Similar observations can be made for augmented Lagrangian cuts, especially if the dual multipliers are not bounded and if the Lipschitz constant estimates are too high. If the estimate is too small, the non-convex cuts are not guaranteed to be valid, though.

Finally, we should note that the generalized conjugacy cuts from Zhang and Sun (2022) have not been tested computationally yet, but were rather introduced in the context of a complexity analysis.

4.3.3 Limitations and outlook

Whereas Dynamic SDDiP yields an interesting way to solve MS-MILPs *in theory*, its practical value to solve these type of problems in real-world applications is yet to be proven. So far, Dynamic SDDiP has only been applied to small illustrative problems or to case studies as part of a larger solution framework, see Sect. 4.4, but not as a standalone method. Therefore, in an ongoing research project it is currently applied to a unit commitment problem. An implementation in Julia is available in the GitHub project `DynamicSDDiP.jl` (Füllner, 2024a).

The main limitation of Dynamic SDDiP in its current form is that it requires a lot of computational effort. First, expressing the CPCs through MILP constraints is complex and may increase the size of the subproblems quickly. Second, computing the CPC is computationally costly, as Lagrangian dual problems have to be solved for each node that is visited in the backward pass. It is yet to be determined if the gain in tightness compared to linear cuts and the gain in dynamic compared to standard SDDiP are worth the additional effort. A possible compromise is to combine the non-convex CPCs with standard linear cuts, so that expensive non-convex cuts are only generated if required in the solution process.

Another research direction that could be worthwhile to explore is to find more efficient ways to represent the CPC using MILP constraints. So far, we used a Big-M approach to reformulate the KKT conditions of the projection problem. However, different approaches are possible, *e.g.*, using SOS-1 conditions or applying novel techniques from bilevel optimization.

4.4 Non-convex nested Benders decomposition

In Paper B, we address research objective 2.3, and propose a new framework to solve multistage (stochastic) non-convex MINLPs. We refer to this method as *non-convex nested Benders decomposition* (NC-NBD). Although the results in the paper are presented for deterministic problems, by traversing the complete scenario tree as in NBD they can be extended to stochastic problems in a straightforward manner. Therefore, for convenience we still refer to the considered problems as multistage stochastic problems in the following. If sampling should be included as in SDDP, some careful modifications are required, as we discuss below.

4.4.1 Theoretical results

The NC-NBD framework assumes (relatively) complete recourse of the MS-MINLP and continuity or Lipschitz continuity of all occurring functions. Other than that, no specific assumptions are taken. In particular, x_{t-1} is allowed to enter the constraint set $\mathcal{X}_t$ at stage t.

It is shown that NC-NBD converges to an approximately optimal solution in finitely many iterations. To our knowledge, apart from the decomposition method in Zhang and Sun (2022), it is the only proven exact solution method for general MS-MINLPs. For a comparison between both methods, see the following section.

The NC-NBD framework combines piecewise linear relaxations, Lipschitz regularization, binary approximation and cut generation in a novel, unique and dynamic fashion. By *dynamic* we mean that all approximations (piecewise linear approximation, state binarization, cut approximation) are refined dynamically where and when it is reasonable during the solution process.

The basic concept is that the considered MS-MINLP is iteratively approximated by MS-MILPs, which are obtained by expressing piecewise linear relaxations of all occurring nonlinear functions as MILP models. These relaxations in turn can be obtained by computing piecewise linear approximations and shifting them down sufficiently (Burlacu et al., 2020; Geißler, 2011). Due to the relaxation property, the MILPs are *outer approximations*, so their optimal values $\widehat{v}^*$ are lower bounds for v^*.

In more detail, NC-NBD consists of two nested loops. In the *outer loop*, with iteration index ℓ, DPE for the true MS-MINLP are considered. The occurring non-convex value functions $Q_t(\cdot)$ are replaced with cut approximations $\mathfrak{Q}_t^\ell(\cdot)$ that are composed of non-convex cuts. The generation of these cuts takes place in the inner loop of the framework. Therefore, the outer loop only contains a forward pass. If the whole scenario tree is traversed, a valid upper bound $\overline{v}^\ell$ can be computed. Moreover, the first-stage problem yields a lower bound $\underline{v}^\ell$. If both bounds are sufficiently close, the outer loop, and by that NC-NBD terminates. Otherwise, for each stage $t \in [T]$, the piecewise linear relaxations are improved in a neighborhood around the current optimal solution x_t^ℓ. Importantly, as we show in the paper, this can be done in such a way that the existing cut approximations remain valid, so the approximation does not have to be started from scratch.

After each outer loop iteration, the current MS-MILP approximation is solved in an *inner loop*. This is done using an NBD-based decomposition method that, apart from not sampling in the forward pass, is equivalent to Dynamic SDDiP, see Sect. 4.3. In particular, this implies that the (expected) value functions occurring in the outer loop are approximated by non-convex cuts, namely CPCs. Again, by careful construction, all existing cuts remain valid with future refinements.

The main steps of the framework are illustrated in Figure 4.2.

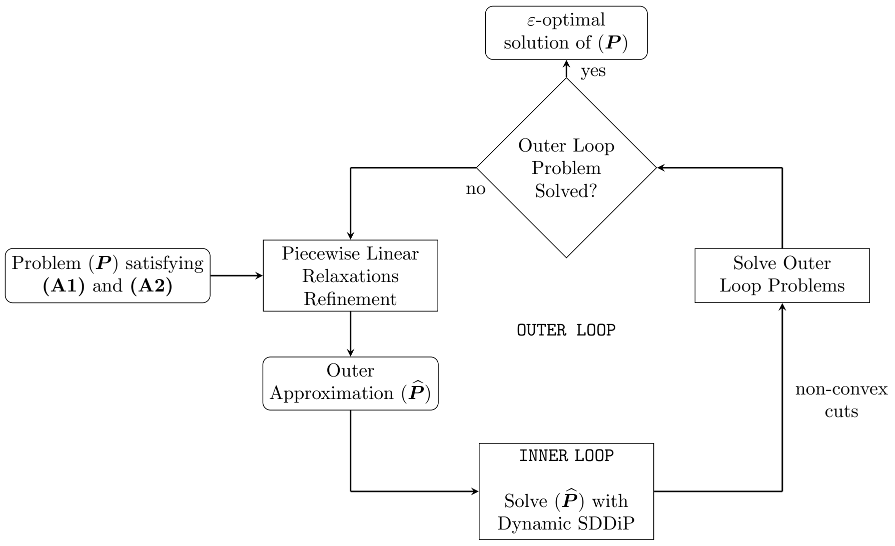

ε-optimal solution of $(\boldsymbol{P})$
yes
Outer Loop Problem Solved?
no
Problem $(\boldsymbol{P})$ satisfying $\mathbf{(A1)}$ and $\mathbf{(A2)}$
Piecewise Linear Relaxations Refinement
Solve Outer Loop Problems
OUTER LOOP
Outer Approximation $(\widehat{\boldsymbol{P}})$
non-convex cuts
INNER LOOP
Solve $(\widehat{\boldsymbol{P}})$ with Dynamic SDDiP

Figure 4.2: Illustration of the main steps of NC-NBD.

Importantly, to solve the MS-MILPs in the inner loop, the regularization parameters σ_t have to be chosen sufficiently large for each $t \in [T]$. Given that the sufficient level may change with the piecewise linear refinements in the outer loop, it may be required to update σ_t iteratively as well. In our computational tests, this was rarely the case, though.

Finally note that the convergence proof in Paper B requires a very technical assumption for the non-convex cuts in the inner loop (Assumption (A4)). However, as discussed in the previous chapter on Dynamic SDDiP, it is shown in Paper C how this assumption can be avoided.

4.4.2 Comparison with existing approaches

For the inner loop and Dynamic SDDiP, most of the relevant differences to existing solution methods have already been discussed in Sect. 4.3. However, we should emphasize that specifically in the NC-NBD setting where an MS-MINLP is solved in a larger framework, and MILPs are only solved in an inner loop, it is crucial that (1) the state binarization precision can be refined dynamically (the required precision may differ for different MILPs constructed in the outer loop), (2) all previously generated cuts remain valid and (3) the cuts generated in the inner loop are expressed in the state variables x_{t-1} that are used in the outer loop problems. Therefore, standard SDDiP cannot be used effectively in this setting.

With regard to the overall framework, as mentioned before, the method proposed in Zhang and Sun (2022) is also an exact solution method for MS-MINLPs. Some main differences to our approach have already been pointed out in Sect. 4.3 on Dynamic SDDiP. Additionally, NC-NBD differs in consisting of two nested loops. This leads to a more complex framework, but it also means that MINLPs only have to be solved occasionally during outer loop iterations, whereas the method in Zhang and Sun (2022) proposes to solve MINLP subproblems in each iteration. This is computationally challenging. Finally, the method in Zhang and Sun (2022) was mostly developed as a means to conduct a complexity analysis. In contrast to NC-NBD, it has not been applied to a case study yet.

4.4.3 Case study and computational results

We test NC-NBD in experiments for moderate-sized instances of a deterministic unit commitment (UC) problem. To this end, it is implemented in Julia in the `NCNBD.jl` project Füllner (2021) (this project is deprectated by now, and currently only the related `DynamicSDDiP.jl` project is maintained).

We run experiments for two different variants of a UC with continuous and binary state variables. In the *base instances*, the objective function is nonlinear concave, as it includes emission costs which are modeled by a quadratic cost curve. In the *valve-point instances*, the valve-point effect for thermal power plants is considered (Pedroso et al., 2014), which leads to a non-convex objective function including a sinusoidal term. The tested instances have between 2 and 36 stages and contain 3 to 10 thermal generators, resulting in 6 to 20 state variables.

For small problems, it can be verified that NC-NBD converges to the exact global solutions. This illustrates the efficacy of NC-NBD to solve MS-MINLPs. However, the observed solution times are very long due to the large computational overhead of solving Lagrangian dual problems in each iteration of the inner loop, as well as due to quickly growing subproblems, caused by modeling the piecewise linear relaxations and the non-convex cuts. Still, NC-NBD manages to outperform some conventional global solvers for problems with 36 stages, but a moderate number of state variables and nonlinearities. Moreover, in retrospect, the implementation of NC-NBD could have been accelerated substantially, as became apparent in later tests of Dynamic SDDiP. Therefore, it should be competitive also for a smaller number of stages.

4.4.4 Limitations and outlook

The main limitation of NC-NBD is the huge computational cost of combining several approximations, and especially solving Lagrangian dual problems in each inner loop iteration and MINLP subproblems in each outer loop iteration. Therefore, in its current form it is rather of theoretical interest.

However, we see some potential to improve NC-NBD in the future. First, the Lagrangian dual problems could be solved more efficiently or with a less strict optimality tolerance. Second, the non-convex cuts could be combined with standard linear cuts to accelerate the solution process. In our experiments, so far we focused on using only the non-convex CPCs to approximate the value functions. Third, as mentioned for Dynamic SDDiP already, more research could be conducted on more efficiently representing the CPCs through MILP constraints. Finally, for a problem at hand, tailor-made piecewise linear relaxations could be used instead of using a general purpose implementation compared to our experiments.

In the future, NC-NBD could also be tested on stochastic instances of unit commitment, however, realistically this requires working on the above performance improvements first. If sampling shall be included, also the stopping criterion for the inner loop has to be adapted accordingly, otherwise this loop may never be left. Additionally, when it comes to proving convergence, it has to be taken into account that sampling may be applied (independently) in both the inner loop and the outer loop.

4.5 A new framework to generate Lagrangian cuts

In Paper D, we present a new framework to generate Lagrangian cuts in decomposition methods for MS-MILPs such as SDDiP. In doing that, we address research objective 2.4.

4.5.1 Theoretical results

The new framework is based on a similar proposal that has been made for standard Benders cuts in Fischetti et al. (2010) and has seen further development and deeper analysis in Brandenberg and Stursberg (2021); Hosseini and Turner (2021).

A key difference to the tranditional generation approach from SDDiP (Zou et al., 2019b) is that the considered Lagrangian relaxation is not derived immediately from the stage-t subproblem. Instead, first a feasibility problem for the epigraph of $\underline{Q}_t^{i+1}(\cdot)$ is formulated. To be precise, given a trial point (x_{t-1}^i, θ_t^i), with θ_t^i the value $\mathfrak{Q}_t^i(x_{t-1}^i)$ of the current cut approximation at stage $t-1$, the problem checks if this point is contained in the epigraph of $\underline{Q}_t^{i+1}(\cdot)$. If this is the case, the optimal value of this feasibility problem is zero, otherwise it is $+\infty$. Now, for this feasibility problem a Lagrangian relaxation can be derived by relaxing the copy constraints $z_t = x_{t-1}^i$ plus the constraint containing θ_t^i.

This type of Lagrangian relaxation has already been presented in Chen and Luedtke (2022) for two-stage stochastic MILPs. However, not only do we extend it to multistage problems, the framework we develop based on this relaxation is also more general and allows for the generation of various different types of Lagrangian cuts, whereas in Chen and Luedtke (2022) only one specific case is considered.

An important observation is that the derived Lagrangian dual problems are unbounded whenever (x_{t-1}^i, θ_t^i) is not contained in the epigraph of the closed convex envelope of $\underline{Q}_t^{i+1}(\cdot)$. To generate a reasonable Lagrangian cut, a bounded problem should be solved, though, as this allows to select the dual optimal solution as the cut coefficients. We present different normalization constraints that are sufficient to achieve this. We show that depending on the chosen normalization, Lagrangian cuts satisfying different quality criteria can be obtained, *e.g.*, deep cuts, facet-defining cuts or Pareto-optimal cuts. In doing that, we draw on similar results that have been presented for Benders decomposition recently and extend them to the stochastic and Lagrangian setting (Brandenberg and Stursberg, 2021; Hosseini and Turner, 2021). In particular, we distinguish linear normalizations (yielding *LN cuts*) and norm-based normalizations (yielding *deep cuts*). Moreover, we investigate in detail the geometrical ideas and relations behind these normalizations.

For LN cuts the coefficients of the normalization constraints have to be chosen carefully to make sure that the normalized Lagrangian dual problem is in fact bounded. Geometrically, we show that this is related to the identification of core points in the epigraphs of $\underline{Q}_t^{i+1}(\cdot)$. For MS-MILPs, identifying such core points can be quite challenging, especially in the presence of integer requirements. We propose five heuristic approaches for the computation of core point candidates.

By incorporating the new cut generation framework we obtain alternative versions of NBD or SDDiP. We prove that under the assumption of binary state variables, NBD still converges to an optimal solution in a finite number of iterations. This proof can be extended to prove almost sure finite convergence for SDDiP using the same arguments as in Zou et al. (2019b).

Our framework allows for a lot of flexibility in cut generation, and thus notably extends the toolbox of SDDiP. The hope is that for a given MS-MILP this can be exploited to identify a type or a combination of different types of Lagrangian cuts that significantly accelerate(s) SDDiP.

4.5.2 Computational results

We perform extensive computational tests for SDDiP to assess the quality of Lagrangian cuts generated using the new framework. To this end, they are added as a feature to the existing GitHub project `DynamicSDDiP.jl` (Füllner, 2024a). For comparison, in our experiments we also generate Benders cuts, strengthened Benders cuts and the standard Lagrangian cuts from SDDiP (Zou et al., 2019b).

All tests are performed on instances of a capacitated lot-sizing problem (CLSP) from the literature. This problem has continuous state variables, so we use a binary approximation of the state space with a discretization precision of 1.0.

We run different batches of tests. First, we test SDDiP using only one type of cut per run. After that, we combine Lagrangian cuts with strengthened Benders cuts to accelerate the solution process. In addition to standard SDDiP with state binarization, we also test it without state binarization. Whereas this method has no convergence guarantees, the dual problems are solved much faster.

Overall, our results show significant improvements of the obtained lower bounds in SDDiP (compared over time, not over iterations) in all cases if using our proposed cuts: with state binarization, without state binarization, combined with strengthened Benders cuts or applying the cuts on their own. With state binarization, especially LN Lagrangian cuts yield strong improvements, without state binarization also deep cuts perform reasonably well.

We also show that SDDiP can be further accelerated by restricting the dual space in the Lagrangian dual problems, even though this again annuls the theoretical convergence guarantees. This approach had been previously suggested by Chen and Luedtke (2022).

However, despite these favorable results, we also observe that better lower bounds do not necessarily translate to better performances of the obtained policies in an in-sample simulation after SDDiP has terminated. In this regard, often policies obtained by using strengthened Benders cuts show the best performance. Moreover, for problems with more than 4 stages none of our test instances manages to close the optimality gap sufficiently, so even after hours of run time the observed gaps are still considerable. This shows that even with carefully modifying the cut generation process, solving large-scale MS-MILPs in reasonable time remains an open challenge.

4.5.3 Limitations and outlook

The main limitation of the new cut generation framework is that, despite improving the lower bounds in SDDiP it does not always lead to better-performing policies because as standard SDDiP, it struggles to close the optimality gap for CLSP.

A major aspect in that regard is that the framework requires a multi-cut approach where all value functions are approximated separately, see Sect. 2.5. This is computationally expensive because q_t cuts are added at stage t per iteration.

We show in the supplementary material of Paper D how our framework can be extended to a single-cut setting. However, first this variant has not been tested in our experiments so far. Second, the dimension of the dual space is increased significantly. Third, this approach only allows for a partial decomposition of the Lagrangian dual problems. Still, we reckon that trying to compute deep Lagrangian cuts or LN Lagrangian cuts in a single-cut framework in a computationally efficient way could be an interesting research direction.

In addition, the computational performance of our proposed cut generation framework could be improved in several ways. The solution of independent Lagrangian duals could be parallelized. Moreover, the dual space restriction suggested by Chen and Luedtke (2022) looks promising to reduce the computational effort while not compromising cut quality by too much. We think that future research could focus more on priorly restricting the dual space to reduce the computational effort for solving Lagrangian dual problems.

Other points of interest with respect to future research are addressing numerical issues, which occasionally occur for LN cuts, identifying core points, or solving Lagrangian dual problems more efficiently in general.

Moreover, although we do not explore and test this in Paper D, we expect that the new cut generation framework may also be applied when Lagrangian duality is used to derive non-convex cuts, *e.g.*, Dynamic SDDiP or NC-NBD. However, these methods do rely on Lipschitz regularizations and solving bounded Lagrangian dual problems, so the framework would have to be adapted to this setting first.

Finally, so far, only tests for CLSP have been conducted and included in Paper D. For the future, further tests are planned on a capacitated facility location problem with pure binary state variables and local integer constraints.

4.6 Non-linear cut-sharing in SDDP

In Paper E we address research objective 2.5, and extend the toolbox of SDDP to stagewise dependent uncertainty occurring in the RHS of MSLPs that is modeled by nonlinear, possibly non-convex AR processes. Our results allow for more flexibility, and therefore potentially more accuracy in modeling uncertainty in the RHS when dealing with MSLPs.

4.6.1 Theoretical results

The theoretical results of the paper consist of two major parts.

First, we deal with general AR processes of type $\xi_t = b_t(\xi_{[t-1]}, \eta_t)$, where $b_t(\cdot)$ is a nonlinear, possibly non-convex function. As explained in Sect. 3.2.4, in this case the common approach to interpret the history $\xi_{[t-1]}$ as an additional state vector leads to non-convex (expected) value functions. Therefore, linear cuts are not sufficient for valid and tight approximations, and thus cannot guarantee convergence of SDDP. We show that instead, cuts can be derived that are linear in the original state variables x_{t-1}, but non-convex in the history $\xi_{[t-1]}$ of the considered AR process, therefore allowing for valid and tight approximations of the non-convex expected value functions.

For these cuts, however, the computational tractability becomes a major challenge for SDDP-type algorithms. In the presented form, the cut formulas at stage t require a recursion over the scenario tree from stages T to $t+1$, which is computationally infeasible in most cases. On the other hand, converting them into closed-form cut formulas that can be evaluated without said recursion is not possible *in general*. Therefore, applying them within SDDP is computationally intractable.

Second, we deal with a special class of nonlinear AR processes, which we refer to as *log-linear* (periodic) AR processes. These processes are widely used to model non-negative stochastic inflows within hydrothermal systems (Shapiro et al., 2013).

By exponentiation, it can be shown that these processes satisfy $\xi_t = b_t(\xi_{[t-1]}, \eta_t)$ with $b_t(\cdot)$ a function of type $e^{\eta_t} \prod_{k \in [t-1]} \xi_k^{\phi_t^{(t-k)}}$ with coefficients $\phi_t^{(t-k)}, k \in [t-1]$. The upper index is put in brackets to distinguish it from exponents.

For this class of AR processes, we are able to develop tractable closed-form cut formulas. The associated cuts are valid and tight. Importantly, they are nonlinear, possibly non-convex in $\xi_{[t-1]}$, but linear in x_{t-1}. Therefore, it is possible to incorporate them into the SDDP subproblems without compromising their linearity (as long as the model equation is not introduced as an explicit constraint). If solvers do not allow for this, the derived formulas can be used to adapt the intercept of a given cut to a scenario at hand, thus to *share* the cut with that particular scenario. To our knowledge, Paper E is the first work proposing *nonlinear* cuts and showing how they can be used within SDDP in this context.

Note that even under special conditions where both the (expected) value functions and our proposed nonlinear cuts become *convex*, our cuts may yield superior results compared to the linear cuts proposed in Guigues (2014). The reason is that both types of cuts are tight, but the nonlinear cuts provide a better approximation outside of the current trial point x_{t-1}^i.

4.6.2 Computational results

To assess the performance of SDDP incorporating our proposed nonlinear cuts, we perform computational tests for a long-term hydrothermal scheduling problem

(LTHS) with stochastic inflows. Our version of SDDP is implemented in Julia and available in the GitHub project `LogLinearSDDP.jl` (Füllner, 2024b).

We use data of the Brazilian power system with 4 energy-equivalent reservoirs, 95 generators and a planning horizon of 60 months (plus additional 60 months to remove end-of-horizon effects) (Shapiro et al., 2013). We run tests using log-linear AR processes with two different lag orders, which have been fitted on historical data. For comparison, we also consider standard SDDP with different linearized AR processes of lag order 1.

First of all, the results of our experiments show that our proposed version of SDDP works as intended. Furthermore, in an out-of-sample simulation performed after termination of SDDP, the policies obtained from our proposed version of SDDP outperform those obtained using standard SDDP with linearized inflow models. More precisely, assuming that the log-linear AR process provides an accurate representation of the inflows, our tailor-made version of SDDP allows for a 7-10% reduction of total costs on average compared to standard SDDP.

These performance gains have to be taken with some caution, though. The main reason is that the average inflow level is consistently lower for the log-linear models than for the linearized models (about 3-4% difference for the largest reservoir). This means that the policies obtained from standard SDDP are trained on an inflow level that is not really comparable to that from the log-linear models, which may explain the worse performance on out-of-sample data from the log-linear models. On the other hand, it is important to clarify that the differences in inflow levels are not deliberately chosen, but arise as a direct consequence of fitting two different types of AR models on the historical data. A simulation analysis indicates that scenarios obtained from the log-linear models better match the statistical properties of the historical data. Similar observations had been made in Löhndorf and Shapiro (2019) before. Hence, based on the available data, the log-linear models should provide a more accurate representation of the inflows. From that perspective, the differences in inflow levels and out-of-performance costs highlight the importance of incorporating nonlinear inflow models directly into SDDP instead of linearizing them.

4.6.3 Limitations and outlook

The main drawback of our approach is that we observe a considerable computational overhead using the proposed non-convex cut formulas in SDDP. For 1000 iterations, our version of SDDP takes about 5-6 hours instead of less than one hour run time for standard SDDP. This overhead is mostly caused by the requirement to iterate over all existing cuts when computing cut intercept factors or adapting the cut intercepts to a given scenario.

A possible remedy is that our methodology can be naturally extended to a hybrid version of SDDP where a log-linear AR process and non-convex cuts are used in early stages, while a linearized process and linear cuts are used in later stages of the MSLP. This extension is explained in the supplementary material to Paper E, but has not been tested computationally yet. Additionally, the efficiency of our implementation could be improved to reduce the computational overhead of using our proposed non-convex cuts.

In the future, research experiments could be conducted for a case study with a convex log-linear AR process. This would allow to get an unbiased comparison of the performance of our proposed non-convex cuts and standard linear cuts, as the same inflows can be used in both cases. In the previous tests, such a comparison is not possible because the results are highly affected by the differences of the AR models themselves.

Finally, with respect to our work on general non-convex AR processes, exploring the usage of approximations for the cut intercepts could be worthwhile in order to avoid the costly recursion over the scenario tree. We provide some first theoretical results pointing in this direction in Paper E, but a more detailed study is left for future research.

Chapter 5

Conclusion

5.1 Summary

This dissertation extends SDDP and related decomposition methods, such as NBD or SDDiP, in several ways.

First, a dynamic version of SDDiP is presented in which the binary approximation of the state variables is dynamically refined and only applied temporarily in order to generate tight non-convex cuts in a lift-and-project scheme.

Second, a generalization of NBD-like and SDDP-like algorithms to general non-convex multistage stochastic MINLPs is proposed. It combines piecewise linear approximations, regularization and the lift-and-project approach from Dynamic SDDiP in a unique fashion.

Third, an extension of SDDP is presented that allows to handle stagewise dependent uncertainty in the right-hand side that is modeled by log-linear autoregressive processes. This contribution allows for more flexibility and accuracy in modeling uncertain data in SDDP, without the need to linearize the stochastic processes.

Fourth, a novel framework is presented to generate Lagrangian cuts in multistage stochastic programming. Depending on the choice of a normalization constraint, Lagrangian cuts satisfying different cut quality criteria can be generated.

As the backbone of the algorithmic ideas, new theoretical results on Lagrangian duality, Lagrangian cuts and Lipschitz regularization are developed.

The proposed solution methods can be applied in several real-world decision-making problems, especially in the energy sector, *e.g.*, unit commitment problems, hydrothermal scheduling problems or generation expansion problems.

5.2 Critical reflection and outlook

In their current form, most of the extensions of SDDP presented in this thesis are rather of theoretical interest, and may serve as a basis for future research, instead of being immediately applicable to large-scale instances of multistage stochastic problems in real-world applications.

As the common denominator among all methods relying on Lagrangian relaxation, *i.e.*, Dynamic SDDiP, NC-NBD and generating Lagrangian cuts, we observe a huge computational bottleneck in solving the Lagrangian dual problems. Moreover, some of these methods involve approximations, such as binary approximation, piecewise linear approximation or non-convex cuts, which require the introduction of several additional (binary) variables and constraints. This lets the considered subproblems grow quickly, and slows down the solution process tremendously over time. Although the proposed methods have convergence guarantees and are shown to improve the approximation quality compared to existing solution methods, it is therefore not clear in general, and probably problem-dependent, if this gain in approximation quality is worth the additional effort. Even if, it has to be considered that an improvement in that regard does not necessarily translate to better-performing policies in in-sample or out-of-sample simulations.

An exception is the proposed SDDP version for log-linear AR processes, which shows considerable computational overhead as well, but within reasonable limits. Also, the increased accuracy of modeling the uncertain data seems to warrant the additional effort, as the obtained policies differ significantly from those obtained using conventional techniques.

For the above reasons, further research, especially on computational improvements, is required for a widespread application of the proposed extensions of SDDP. In this thesis we point out that there exists significant room for improvement in different directions. However, it is important to keep in mind that the tackled problems are multistage/dynamic, non-convex and stochastic, thus very complex by nature and expected to be computationally challenging to solve.

References

Ahmed, S., Cabral, F. G., and da Costa, B. F. P. (2022). Stochastic Lipschitz dynamic programming. *Mathematical Programming*, 191:755–793.

Ávila, D., Papavasiliou, A., and Löhndorf, N. (2024). Batch learning SDDP for long-term hydrothermal planning. *IEEE Transactions on Power Systems*, 39(1):614–627.

Bellman, R. E. (1957). *Dynamic programming.* Princeton University Press, Princeton, New Jersey.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252.

Birge, J. R. (1980). *Solution methods for stochastic dynamic linear programs.* Phd dissertation, Systems Optimization Laboratory, Stanford University.

Birge, J. R. and Louveaux, F. (2011). *Introduction to stochastic programming.* Springer Series in Operations Research and Financial Engineering. Springer Science & Business Media, 2nd edition.

Brandenberg, R. and Stursberg, P. (2021). Refined cut selection for benders decomposition: applied to network capacity expansion problems. *Mathematical Methods of Operations Research*, 94:383–412.

Burlacu, R., Geißler, B., and Schewe, L. (2020). Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes. *Optimization Methods and Software*, 35(1):37–64.

Cerisola, S., Latorre, J. M., and Ramos, A. (2012). Stochastic dual dynamic programming applied to nonconvex hydrothermal models. *European Journal of Operational Research*, 218(3):687–697.

Chen, R. and Luedtke, J. (2022). On generating Lagrangian cuts for two-stage stochastic integer programs. *INFORMS Journal on Computing*, 34(4):2332–2349.

de Matos, V. L. and Finardi, E. C. (2012). A computational study of a stochastic optimization model for long term hydrothermal scheduling. *International Journal of Electrical Power & Energy Systems*, 43(1):1443–1452.

Downward, A., Dowson, O., and Baucke, R. (2020). Stochastic dual dynamic programming with stagewise-dependent objective uncertainty. *Operations Research Letters*, 48:33–39.

Fischetti, M., Salvagnin, D., and Zanette, A. (2010). A note on the selection of Benders' cuts. *Mathematical Programming, Ser. B*, 124:175–182.

Füllner, C. (2021). NCNBD.jl. Code released on GitHub `https://github.com/ChrisFuelOR/NCNBD.jl`.

Füllner, C. (2024a). DynamicSDDiP.jl. Code released on GitHub `https://github.com/ChrisFuelOR/DynamicSDDiP.jl`.

Füllner, C. (2024b). LogLinearSDDP.jl. Code released on GitHub `https://github.com/ChrisFuelOR/LogLinearSDDP.jl`.

Füllner, C. and Rebennack, S. (2022). Non-convex nested Benders decomposition. *Mathematical Programming*, 196:987–1024.

Füllner, C. and Rebennack, S. (2023). Stochastic dual dynamic programming and its variants – A review. Preprint, available at `http://www.optimization-online.org/DB_FILE/2021/01/8217.pdf`.

Füllner, C. and Rebennack, S. (2024). Nonlinear cut-sharing in stochastic dual dynamic programming for log-linear autoregressive uncertainty in the right-hand side. Preprint.

Füllner, C., Sun, X. A., and Rebennack, S. (2024a). A new framework to generate lagrangian cuts in multistage stochastic mixed-integer programming. Preprint.

Füllner, C., Sun, X. A., and Rebennack, S. (2024b). On regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming. Preprint.

Geißler, B. (2011). *Towards globally optimal solutions for MINLPs by discretization techniques with applications in gas network optimization*. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg.

Girardeau, P., Leclère, V., and Philpott, A. B. (2015). On the convergence of decomposition methods for multistage stochastic convex programs. *Mathematics of Operations Research*, 40(1):130–145.

Guigues, V. (2014). SDDP for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57:167–203.

Guigues, V. (2016). Convergence analysis of sampling-based decomposition methods for risk-averse multistage stochastic convex programs. *SIAM Journal on Optimization*, 26(4):2468–2494.

Guigues, V., Shapiro, A., and Cheng, Y. (2023). Duality and sensitivity analysis of multistage linear stochastic programs. *European Journal of Operational Research*, 308(2):752–767.

Hjelmeland, M. N., Zou, J., Helseth, A., and Ahmed, S. (2019). Nonconvex medium-term hydropower scheduling by stochastic dual dynamic integer programming. *IEEE Transactions on Sustainable Energy*, 10(1).

Hosseini, M. and Turner, J. G. (2021). Deepest cuts for Benders decomposition. Preprint, available online at `https://arxiv.org/abs/2110.08448`.

Infanger, G. and Morton, D. P. (1996). Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75:241–256.

Kelley, J. E. (1960). The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712.

Lan, G. (2022). Complexity of stochastic dual dynamic programming. *Mathematical Programming*, 191:717–754.

Lohmann, T., Hering, A. S., and Rebennack, S. (2016). Spatio-temporal hydro forecasting of multireservoir inflows for hydro-thermal scheduling. *European Journal of Operational Research*, 255(1):243–258.

Löhndorf, N. and Shapiro, A. (2019). Modeling time-dependent randomness in stochastic dual dynamic programming. *European Journal of Operational Research*, 273(2):650–661.

Pedroso, J. P., Kubo, M., and Viana, A. (2014). Unit commitment with valve-point loading effect. Technical report, Universidade do Porto.

Pereira, M. V. F. and Pinto, L. M. V. G. (1991). Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375.

Philpott, A. and Guan, Z. (2008). On the convergence of stochastic dual dynamic programming and related methods. *Operations Research Letters*, 36(4):450–455.

Philpott, A. B., de Matos, V. L., and Finardi, E. (2013). On solving multistage stochastic programs with coherent risk measures. *Operations Research*, 61(4):957–970.

Philpott, A. B., Wahid, F., and Bonnans, F. (2020). MIDAS: A mixed integer dynamic approximation scheme. *Mathematical Programming*, 181:19–50.

Queiroz, A. R. and Morton, D. P. (2013). Sharing cuts under aggregated forecast when decomposing multi-stage stochastic programs. *Operations Research Letters*, 41(3):311–316.

Rebennack, S. (2016). Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming. *Mathematical Programming*, 156:343–389.

Rockafellar, R. T. (1970). *Convex analysis*. Princeton university press.

Shapiro, A. (2011). Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, 209(1):63–72.

Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2014). *Lectures on Stochastic Programming: Modeling and Theory, 2nd Ed.* Society for Industrial and Applied Mathematics.

Shapiro, A., Tekaya, W., da Costa, J. P., and Soares, M. P. (2013). Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391.

Steeger, G. and Rebennack, S. (2017). Dynamic convexification within nested Benders decomposition using Lagrangian relaxation: an application to the strategic bidding problem. *European Journal of Operational Research*, 257(2):669–686.

van Ackooij, W. and Warin, X. (2020). On conditional cuts for stochastic dual dynamic programming. *EURO Journal on Computational Optimization*, 8(2):173–199.

van Slyke, R. M. and Wets, R. (1969). L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663.

Zhang, S. and Sun, X. A. (2022). Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 196:935–985.

Zou, J., Ahmed, S., and Sun, X. A. (2019a). Multistage stochastic unit commitment using stochastic dual dynamic integer programming. *IEEE Transactions on Power Systems*, 34(3).

Zou, J., Ahmed, S., and Sun, X. A. (2019b). Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502.

List of abbreviations

AR autoregressive

DPE dynamic programming equations

LP linear program

MILP mixed-integer linear program

MINLP mixed-integer nonlinear program

MSP multistage stochastic program

MSLP multistage stochastic linear program

MS-MILP multistage stochastic mixed-integer linear program

MS-MINLP multistage stochastic mixed-integer nonlinear program

NBD nested Benders decomposition

NC-NBD non-convex nested Benders decomposition

RHS right-hand side

SDDP stochastic dual dynamic programming

SDDiP stochastic dual dynamic integer programming

Part II

Research papers

Paper A

Stochastic dual dynamic programming and its variants – A review

Christian Füllner[a], Steffen Rebennack[a]

[a] *Karlsruhe Institute of Technology (KIT), Institute for Operations Research, Stochastic Optimization, Karlsruhe, Germany*

Submitted to *SIAM Review*, suggested citation:
Füllner C., Rebennack S. (2023), Stochastic dual dynamic programming and its variants – A review, available at `http://www.optimization-online.org/DB_FILE/2021/01/8217.pdf`.

Stochastic Dual Dynamic Programming and its variants – a review

Christian Füllner[1*] and Steffen Rebennack[1]

[1]Institute for Operations Research (IOR), Stochastic Optimization (SOP), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[*]Address correspondence to: christian.fuellner@kit.edu

Abstract

We provide a tutorial-type review on stochastic dual dynamic programming (SDDP), as one of the state-of-the-art solution methods for large-scale multistage stochastic programs. Since introduced about 30 years ago for solving large-scale multistage stochastic linear programming problems in energy planning, SDDP has been applied to practical problems from several fields and is enriched by various improvements and enhancements to broader problem classes. We begin with a detailed introduction to SDDP, with special focus on its motivation, its complexity and required assumptions. Then, we present and discuss in depth the existing enhancements as well as current research trends, allowing for an alleviation of those assumptions.

1 Introduction

In many decision-making situations at least some of the data are uncertain. While this uncertainty is often disregarded, the importance of taking it into account during the decision process was already recognized in 1955 by George Dantzig [44]. In stochastic programming, a common approach to achieve this is to split up this process into two different stages: At the first stage, decisions have to be taken before any uncertain data are revealed and to hedge against the existing uncertainty (so-called *here-and-now* decisions). At the second stage, corrective actions, called *recourse* or *wait-and-see* decisions, can be taken, once the realization of the uncertain data is known [26]. Typically, the aim is to determine an optimal decision rule *in expectation* or with respect to some risk measure.

In many practical applications, not only two, but multiple subsequent decisions have to be taken [7]. If these decisions cannot be taken independently, but are coupled by their effects on a system state, *e.g.*, hydroelectric generation affecting the water level of a reservoir, or orders affecting the size of an inventory stock, this can be modeled as a multistage stochastic problem with several subsequent recourse decisions (this is also referred to as *dynamic programming*, and was recently coined *sequential decision problem* in [170]). In such a problem, trade-offs have to be made between using an existing resource immediately or saving it up for later stages, taking into account the future uncertainty.

Stochastic dual dynamic programming (SDDP) is an algorithm to tackle such multistage stochastic problems in order to compute, or at least approximate, an optimal *policy*, that is, a strategy or decision rule providing the best here-and-now decision as well as the best wait-and-see decisions for any stage and any given realization of the uncertain data. It was first proposed by Pereira and Pinto in 1991 in [153].

Historically, SDDP has its roots in two separate research streams dealing with sequential decision problems. The first one is *stochastic dynamic programming* (SDP), which is closely related to stochastic optimal control and Markov decision processes. Here, a crucial assumption is that the uncertain data on different stages of the decision process are independent of each other (or at least Markovian). In this case, multistage stochastic problems can be expressed by dynamic programming equations (DPE), which decompose the large-scale problem by stages into several smaller subproblems. These DPE exploit the famous optimality principle by Bellman [13], which allows one to express the optimal objective value from some stage t onwards, given some state x_{t-1}, recursively by means of some stage-t objective function and a so-called *expected value function* $\mathcal{Q}_t(\cdot)$, modeling the expected optimal objective value from stage $t+1$ onwards, given the new state x_t. We formally introduce these concepts in Sect. 2.4.

The DPE can be solved exactly by SDP solution methods, such as value iteration [13]. Basically, this method is based on traversing the stages backwards and evaluating the expected value functions $\mathcal{Q}_t(\cdot)$ for all possible states x_{t-1} (concept of a lookup table). Each such evaluation requires solving an optimization problem for all possible realizations of the uncertain data, which, in turn, requires finding an optimal decision over all possible actions. For this evaluation to be possible, it is assumed that the state space, the action space and the scenario space are finite – otherwise they have to be discretized. However, even in the discrete case, enumerating all possible combinations is computationally intractable for all but low dimensions, as the number of evaluations suffers from combinatorial explosion. This phenomenon is known as the *curse of dimensionality* of SDP [169]. In order to circumvent this, *approximate dynamic programming* (ADP) methods have been developed, where expected value functions are approximated instead of being evaluated exactly (or where optimal policies are approximated using different strategies) [169, 170]. SDDP can be regarded as one such method. Due to its close relation with SDP it also heavily relies on the assumption of stagewise independence.

A second perspective on SDDP is one from *stochastic programming*. Traditionally, in this field, multistage uncertain data are often modeled by a scenario tree, which branches at each stage and consists of finitely many possible scenarios. Scenario trees do not require the stochastic data process to be stagewise independent. Using finite scenario trees and assuming linearity, a multistage stochastic program can be reformulated as a large-scale linear programming problem [179]. However, in this extensive form such a problem usually is way too large to be solved by monolithic approaches, since the number of decision variables and constraints grows exponentially in the number of stages. To cope with this challenge, special solution techniques are required which decompose the problem. Based on the L-shaped method for solving two-stage stochastic programs [228] (a special variant of Benders decomposition [17]), one such idea is the extension of Benders-type solution methods to the multistage setting. The *nested Benders decomposition* (NBD) method by Birge [24] is such an extension. It can be interpreted as a nested sequence of solving two-stage stochastic programs while traversing the scenario tree. In contrast to SDP, in NBD the functions $\mathcal{Q}_t(\cdot)$ are not evaluated at all possible states, but iteratively approximated by linear functions called cutting-planes or *cuts*, starting from a rough initial relaxation. Such approximation is possible, since $\mathcal{Q}_t(\cdot)$ can be proven to be convex in x_{t-1} for LPs. It also allows to consider a continuous state space without discretization.

While NBD is a reasonable method to solve multistage stochastic linear programs of moderate time horizons (maximum 4 or 5 time steps), for larger problems, it is still computationally prohibitive, as the scenario tree grows exponentially in the number of stages. As a relief, several methods have been proposed to combine the cutting-plane approximations in NBD with sampling techniques from simulation [38, 53, 102]. The most prominent among these methods is SDDP. From this perspective, SDDP can be considered a sampling-based variant of NBD. In order to use the sampling step in a beneficial way, compared to NBD, SDDP comes with the additional prerequisite that the data process is stagewise independent.

Application-wise, the development of SDDP is closely related to hydrothermal operational planning, which attempts to determine cost-optimal generation decisions for thermal and hydroelectric power plants over several stages, while ensuring system balance and satisfaction of technical constraints. Since future water availability is affected by uncertain inflows into hydro reservoirs, this optimization problem can be considered multistage, stochastic, and thus very complex.

Prior to SDDP, various solution techniques had been proposed to tackle this type of problem. Among those are simulation models, linear programming techniques (either based on assuming inflows as deterministic or based on reformulating stochastic LPs into a deterministic equivalent), special variants of dynamic programming and SDP [230]. However, all of these techniques either do not consider the uncertain nature of inflows, suffer from the aforementioned curses of dimensionality or do not guarantee convergence. For operating a large-scale power system dominated by hydro power these shortcomings are severe, as they prohibit a cost-minimal and reliable, but at the same time computationally efficient operational planning. The development of SDDP by Pereira and Pinto was directly driven by the endeavor to replace SDP with a more efficient optimization technique in operating the Brazilian power system. While it avoids *some* of the computational drawbacks of SDP or NBD (sometimes advertized as “breaking the curse of dimensionality”), SDDP comes with its own shortcomings, as we thoroughly discuss in this paper.

Since its invention in 1991 SDDP has gained enormous interest, both from a theoretical and an application perspective. To this date, it can be considered one of the state-of-the-art solution methods for large-scale multistage stochastic problems. For this reason, it is used in various practical applications to optimize decision processes, for instance hydrothermal operational planning, portfolio optimization or inventory management, see Sect. 9.

Several extensions and improvements of SDDP have been proposed by now, many of them attempting to relax the originally required theoretical assumptions, making SDDP applicable to broader problem classes. Others strive for improving the performance of SDDP because, despite its merits, the algorithm may take too long to converge for large problem instances.

Due to both, the sheer amount and the variety of proposed enhancements, SDDP has developed into a wide-ranging research area with several sub-branches, becoming increasingly difficult to keep track of. In this article, we give a comprehensive tutorial-type review on SDDP-related research, covering its basic principle and assumptions, strengths and weaknesses, existing extensions and current research trends.

Table 1: Table of contents.

1.1 Structure

The structure of this review is summarized in Table 1. The review can be divided into four major parts. In the first part (Sect. 2 to 8), we discuss the basic mechanism of SDDP. This includes formal preliminaries to formulate multistage stochastic decision problems, but also the main algorithmic steps of SDDP and a complexity analysis. In particular, we point out crucial assumptions for standard SDDP to work. In the second part (Sect. 9 and 10), we discuss applications, which underline the practical relevance of SDDP, but also the requirement to relax some of the standard assumptions. In the third part (Sect. 11 to 20), we discuss various extensions of SDDP to cases where the standard assumptions are relaxed. These extensions comprise modifications of SDDP itself as well as modifications or reformulations of the considered decision problems. Finally, in the fourth part (Sect. 21), we discuss approaches to improve the computational performance of SDDP.

1.2 Terminology and Notation

As already mentioned, SDDP is linked to several different research fields and communities, such as stochastic programming, dynamic programming, Markov decision processes, optimal control or reinforcement learning, each using different terminology and notation. This aggravates a presentation of SDDP in a form that is familiar and accessible to all those interested.

Table 2: Abbreviations that are used throughout the text.

(P)AR	(Periodic) Autoregressive process
DPE	Dynamic programming equations
LP	Linear program
MI(N)LP	Mixed-integer (non-)linear program
MSLP	Multistage stochastic linear programming problem
NBD	Nested Benders Decomposition
RHS	Right-hand side
SDP	Stochastic Dynamic Programming
SDDP	Stochastic Dual Dynamic Programming

To our knowledge, the majority of active research on SDDP is conducted by researchers from the stochastic programming community. For this reason, in many sections we resort to stochastic programming language and notation. On the other hand, this review is also dedicated to offer an access to SDDP for practitioners and researchers from fields in which different perspectives and notation are standard. Therefore, we address these differences if required for the understanding of SDDP, and attempt to avoid heavy mathematical programming notation whenever possible, especially in early sections introducing SDDP.

For a general, not SDDP-specific, attempt at unifying different disciplines related to optimization under uncertainty and sequential decision processes into a common framework, we refer to the excellent book [170].

In the following, we denote random variables by bold letters, *e.g.*, $\boldsymbol{\xi}$, and their realizations by letters in normal font, *e.g.*, ξ. To enhance readability, we summarize some recurring abbreviations in Table 2.

2 Preliminaries for SDDP

In order to present SDDP in its standard form, we start by formally introducing the considered decision problem. In particular, we point out assumptions which are crucial for the presented SDDP method to work.

We consider a multistage decision process where decisions x_t have to be taken over some horizon $[T] := \{1, \dots, T\}$ consisting of T stages, with the aim to minimize some objective function subject to constraints. For now, the horizon T is assumed to satisfy the following condition:

Assumption 1 (Finite and deterministic horizon)**.** *The number $T \in \mathbb{N}$ of stages is finite and deterministic.*

We discuss later how SDDP may be applied to cases where this is not satisfied, see Sect. 19 and 20.

2.1 Modeling the Uncertainty

The data in the considered decision process can be subject to uncertainty, which is revealed over time. To this end, we consider a filtered probability space $(\Omega, \mathscr{F}, \mathbb{P})$ with sample space Ω, σ-algebra $\mathscr{F}$ and probability measure $\mathbb{P}$, which models the uncertainty over the horizon $[T]$. Further let $\mathscr{F}_1, \ldots, \mathscr{F}_T$ with $\mathscr{F}_T := \mathscr{F}$ be a sequence of σ-algebras containing the events observable up to time t, thus defining a filtration with $\mathscr{F}_1 \subseteq \mathscr{F}_2 \cdots \subseteq \mathscr{F}_T$, and let Ω_t be the sample space restricted to stage $t \in [T]$. We then define a stochastic process $(\boldsymbol{\xi}_t)_{t\in[T]}$ with random vectors $\boldsymbol{\xi}_t : \Omega_t \to \mathbb{R}^{\kappa_t}, \kappa_t \in \mathbb{N}$, over the probability space. These random vectors are assumed to be $\mathscr{F}_t$-measurable functions. We denote their support by $\Xi_t \subseteq \mathbb{R}^{\kappa_t}$ for all $t \in [T]$. For the first stage, the data are assumed deterministic, *i.e.*, Ξ_1 is a singleton. For each random vector $\boldsymbol{\xi}_t$, we denote a specific realization by ξ_t.

As a crucial ingredient for SDDP to work, we assume that the uncertainty on different stages does not depend on each other.

Assumption 2 (Stagewise independence)**.** *For all $t \in [T]$, the random vector $\boldsymbol{\xi}_t$ is independent of the history $\xi_{[t-1]} := (\xi_1, \ldots, \xi_{t-1})$ of the data process.*

Under Assumption 2, the random vectors $\boldsymbol{\xi}_t$ are often referred to as *noises*. This assumption is common in dynamic programming, but not standard in stochastic programming. In practical applications it may not be satisfied. We address how to apply SDDP to problems with stagewise dependent uncertainty in Sect. 14.

Additionally, we take the following assumptions for the stochastic process.

Assumption 3 (Known distribution)**.** *The probability distribution $F_{\boldsymbol{\xi}}$ of the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is known.*

Assumption 4 (Exogeneity)**.** *The random variables $\boldsymbol{\xi}_t$ are exogeneous, i.e., the distribution $F_{\boldsymbol{\xi}}$ of the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is independent of decisions $(x_t)_{t\in[T]}$.*

Assumption 5 (Finite randomness)**.** *The support Ξ_t of $\boldsymbol{\xi}_t$ is finite for all $t \in [T]$. The number of noise realizations at stage $t \in [T]$ is given by $q_t \in \mathbb{N}$ with $q_1 = 1$.*

We discuss how to apply SDDP if Assumption 3 is not satisfied in Sect. 13. If Assumption 4 is not satisfied, the problem is said to have *decision-dependent* uncertainty [115]. As this case is not covered in the literature on SDDP so far, we do not discuss the relaxation of this assumption.

Assumption 5 is a key assumption for SDDP and standard in dynamic programming and stochastic programming in order to obtain computationally tractable problems. Whereas there exists no direct extension of SDDP to problems that do not satisfy Assumption 5, we discuss possible ways to treat such problems in Sect. 11. As $\boldsymbol{\xi}_t$ is a discrete and finite random variable for all $t \in [T]$, its distribution $F_{\boldsymbol{\xi}}$ is defined by finitely many realizations $\xi_{tj}, j = 1, \ldots, q_t$, and assigned probabilities p_{tj}.

The stagewise independent and finite data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ can be illustrated by a *recombining scenario tree* [179], also called *scenario lattice* [129]. On each stage $t \in [T]$, its nodes represent the possible noise realizations $\xi_{tj}, j = 1, \ldots, q_t$. Due to stagewise independence (Assumption 2) all nodes at the same stage have an identical set of child nodes with the same noise realizations and associated probabilities. We call paths $\xi = (\xi_t)_{t\in[T]}$ through the complete tree (stage-T) *scenarios* and index them by $s \in \mathcal{S}$. Note that for each scenario ξ^s, there exists some $j_s \in \{1, \ldots, q_t\}$ such that $\xi_t^s = \xi_{tj_s}$. The total number of different scenarios modeled by the tree is $|\mathcal{S}| = \prod_{t\in[T]} q_t$. An example of a recombining scenario tree is presented in Figure 1.

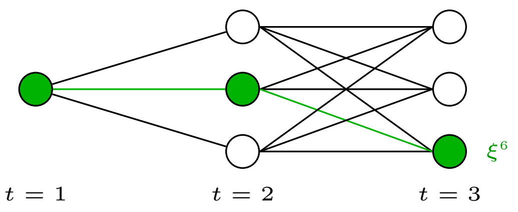


Figure 1: Recombining tree with 3 realizations per stage and highlighted scenario ξ^6.

2.2 The Decision Process

With the stochastic process in mind, we can now turn to the decision process. At stage 1, the *here-and-now* decision x_1 is taken to hedge against the uncertainty in the following stages. At those stages, recourse decisions $\boldsymbol{x}_t \in \mathbb{R}^{n_t}, n_t \in \mathbb{N}$, can be taken under knowledge of the realization of the data process at stage t. This decision process is illustrated in Figure 2.

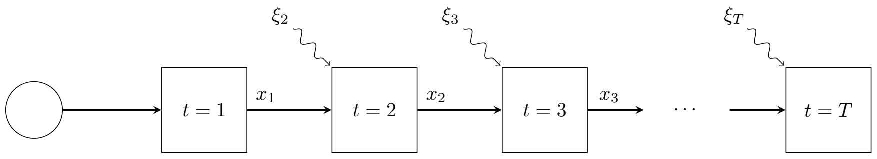


Figure 2: Multistage decision process with uncertainty.

In other words, the paradigm is that decisions can be taken *after* the uncertainty corresponding to stage t has unfolded (so-called *wait-and-see* decisions), making $\boldsymbol{x}_t(\xi_t)$ a function of $\boldsymbol{\xi}_t$, and by that a random variable. We account for that using a bold symbol. Importantly, $\boldsymbol{x}_t(\cdot)$ does only depend on realizations up to stage t, but does not anticipate future events or decisions. Future events are only considered using distributional information. Therefore, $\boldsymbol{x}_t(\cdot)$ is $\mathscr{F}_t$-measurable [201]. As we will see, $\boldsymbol{x}_t(\cdot)$ may also depend on the choice for $\boldsymbol{x}_{t-1}(\cdot)$ and so on, so that despite stagewise independence (Assumption 2), $\boldsymbol{x}_t(\cdot)$ is actually a function of the whole history $\xi_{[t]}$ of the data process.

A sequence of decision functions $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$ is called a *policy* and provides a decision rule for all stages $t \in [T]$ and any realization of the data process. By the previous arguments, such a policy is *non-anticipative*, modeling a sequence of nested conditional decisions. The aim of the decision process is to determine an *optimal* policy with respect to a given objective function and a given set of constraints.

In this context, the following assumptions are standard for SDDP.

Assumption 6 (Linearity)**.** *All functions occurring in the objective and the constraints are linear.*

Assumption 7 (Consecutive coupling)**.** *Only decisions on consecutive stages can be linked by constraints.*

Assumption 8 (Risk-neutral policy)**.** *The aim is to determine an optimal risk-neutral policy.*

As not all of these assumptions are guaranteed to be satisfied for an arbitrary problem in practice, we discuss possible ways to relax them in Sect. 15 and 16 (for Assumption 6), Sect. 18 (for Assumption 7) and Sect. 12 (for Assumption 8).

Under Assumptions 6 and 8, the optimization objective can be expressed as

$$\min_{x_1,\boldsymbol{x_2},\ldots,\boldsymbol{x_T}} \mathbb{E}\left[\sum_{t\in[T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})\right], \tag{1}$$

with data vectors $c_t \in \mathbb{R}^{n_t}$ for all $t \in [T]$ and $\mathbb{E}[\cdot]$ denoting the expected value.

Under Assumptions 6 and 7, for all $t \in [T]$, the constraints on the decisions can be expressed using the $\mathscr{F}_t$-measurable set-valued mappings $\mathcal{X}_t(\cdot)$, which for any x_{t-1} and any $\xi_t \in \Xi_t$ are defined by

$$\mathcal{X}_t(x_{t-1},\xi_t) := \Big\{x_t \in X_t \subset \mathbb{R}^{n_t} \ : \ T_{t-1}(\xi_t)x_{t-1} + W_t(\xi_t)x_t = h_t(\xi_t)\Big\}. \tag{2}$$

Here, $h_t \in \mathbb{R}^{m_t}$ are real data vectors (for $m_t \in \mathbb{N}$), T_t and W_t are real-valued $(m_{t+1} \times n_t)$ and $(m_t \times n_t)$ data matrices and X_t is a non-empty polyhedron, *e.g.*, modeling non-negativity constraints.

As stated before, some (or all) of the problem data can be subject to uncertainty. Hence, for all $t \in [T]$, we consider random variables $\boldsymbol{c}_t(\xi_t), \boldsymbol{T}_{t-1}(\xi_t), \boldsymbol{W}_t(\xi_t)$ and $\boldsymbol{h}_t(\xi_t)$ depending on realizations of $\boldsymbol{\xi}_t$. X_t is considered deterministic. Note again that the first stage is assumed to be deterministic, and that $T_0 \equiv 0$ and $x_0 \equiv 0$. Hence, we define $\mathcal{X}_1 :\equiv \mathcal{X}_1(x_0,\xi_1)$.

Remark 2.1. *For notational simplicity, when we deal with finite random variables $\boldsymbol{\xi}_t$ in this paper, we often index the vectors and matrices c_t, T_{t-1}, W_t and h_t with $j = 1,\ldots,q_t$ if we address specific realizations,* e.g., $c_{tj} := c_t(\xi_{tj})$.

Remark 2.2 (Dynamic programming perspective)**.** *In dynamic programming, Markov decision processes or optimal control, usually a slightly different perspective on sequential decision processes is chosen (see [170] for a comprehensive overview). The main difference is that the occurring variables are differentiated into* state variables *and actual* decisions*. State variables $s_t \in S_t$ model the system state at some stage t. S_t is called the state space. Importantly, state variables may not only comprise the resource state, but also the information or belief state of a system. Local decision variables model decisions on a stage t given a state s_t. In dynamic programming they are usually discrete and called* actions $a_t \in A_t(s_t)$*, in optimal control they are usually continuous and called* controls $u_t \in U_t(s_t)$*. $A_t(s_t)$ and $U_t(s_t)$ are the action space or control space, respectively. The actions or controls are what an agent actually decides on given the current state s_t, whereas the new state s_{t+1} is uniquely determined as $s_{t+1} = \mathcal{T}_t(s_t, u_t, \xi_{t+1})$ using a given transition function $\mathcal{T}_t(\cdot)$ which captures the system dynamic. Therefore, from this perspective, a policy is a sequence of mappings $\boldsymbol{\pi}_t : S_t \to U_t$ from the state space to the control (or action) space. By proper modeling of the state variable, Assumption 7 is naturally satisfied.*

In our above setting, states and actions are intertwined. We can set $s_t = (x_{t-1}, \xi_t)$ and $u_t = x_t$ to switch perspectives [6]. The state space, control space and transition function are then implicitly given by (2) *and the definition of $\boldsymbol{\xi}_t$.*

Whereas our above definitions are prevalent in the literature on SDDP, sometimes also an optimal control perspective is adopted, e.g., *in the French community working on SDDP (see for example [78]). However, in this case usually only the resource state r_t is explicitly considered as a state variable (while not including information on $\boldsymbol{\xi}_t$). Translating our above setting, this implies that $r_t = x_{t-1}$ with state space $R_t = X_t$,*

$u_t = x_t$ and due to $r_{t+1} = u_t$, both the control space $U_t(r_t, \xi_t)$ and the transition function $\mathcal{T}_t(r_t, u_t, \xi_t)$ are given by the equations in (2).

It is worth mentioning that the distinction between state variables and controls (actions) is not only a matter of notation, though, but also relevant computationally because the complexity of SDDP differs in the state and control dimension (see also Remark 2.6 and Sect. 4.2).

Given the constraint sets (2) for all $t \in [T]$, let $\mathcal{X}_0 := \{x_0\}$ and recursively define

$$\mathcal{X}_t := \bigcup_{x_{t-1} \in \mathcal{X}_{t-1}} \bigcup_{\xi_t \in \Xi_t} \mathcal{X}_t(x_{t-1}, \xi_t)$$

for all $t \in [T]$ [71]. Using these definitions, we are able to state assumptions which we require for the feasibility of our decision problem:

Assumption 9. *(Feasibility and Compactness)*

(a) For all $t \in [T]$, all $x_{t-1} \in \mathcal{X}_{t-1}$ and almost all $\xi_t \in \Xi_t$, the set $\mathcal{X}_t(x_{t-1}, \xi_t)$ is a non-empty compact subset of $\mathbb{R}^{n_t}$ (relatively complete recourse).

(b) The set $\mathcal{X}_t$ is bounded for all $t \in [T]$.

Remark 2.3. *Note that the linearity assumption (see Assumption 6), immediately implies that Assumption 9 (a) is not only satisfied for all $x_{t-1} \in \mathcal{X}_{t-1}$, but for all $x_{t-1} \in conv(\mathcal{X}_{t-1})$, where $conv(S)$ denotes the convex hull of a set S.*

The set $\mathcal{X}_t \in \mathbb{R}^{n_t}$ is called *reachable set* in [71] and *effective feasible region* in [117]. It may as well be referred to as the state space sometimes, because in our setting x_t also takes the role of a state variable. However, in other cases the larger polyhedral set X_t may be called state space.

The boundedness of $\mathcal{X}_t$ in (b) is required for some of the convergence results on SDDP presented in Sect. 4. It follows naturally if X_t is bounded, since $\mathcal{X}_t \subseteq X_t$. Property (a) is convenient, but not necessarily required. We discuss possible ways to relax it in Sect. 17.

With all the ingredients defined, we can now model the decision problem in a form that can be tackled by SDDP. Based on its properties, in the following we refer to this problem as a *multistage stochastic linear programming problem* (MSLP). If not specified otherwise, throughout this paper, we assume that (MSLP) satisfies Assumptions 1 to 9. We first discuss two different modeling approaches which are common in the literature.

2.3 Single-problem Formulation

One way to model the decision problem (MSLP) is to formulate it as a single optimization problem. This modeling approach is common in the stochastic programming community. The optimization problem can be obtained by combining (1) with the constraints in (2) for all $t \in [T]$.

Then, under Assumptions 1 to 9, (MSLP) can be written as

$$v^* := \begin{cases} \min\limits_{x_1, \boldsymbol{x_2}, \dots, \boldsymbol{x_T}} & \mathbb{E}\left[\sum\limits_{t \in [T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})\right] \\ \text{s.t.} & x_1 \in \mathcal{X}_1 \\ & \boldsymbol{x}_t(\xi_{[t]}) \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_{[t]} \ \forall t = 2, \dots, T \\ & \boldsymbol{x}_t(\cdot) \ \mathscr{F}_t\text{-measurable} \quad \forall t = 2, \dots, T. \end{cases} \tag{3}$$

Importantly, the decision variables $x_t \in \mathbb{R}^{n_t}$ depend on $\boldsymbol{\xi_t}$ (and on x_{t-1}), so in this representation we optimize over policies. A policy $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ is called *feasible* (or *admissible*) if it satisfies the constraints in (MSLP) for almost every realization of the random data [201].

Assumption 9 (a) implies that the feasible set of (MSLP) is compact and non-empty, and by linearity of the objective (Assumption 6) it follows that v^* is finite.

Due to optimizing over policies, without Assumption 5, (MSLP) is an infinite-dimensional optimization problem. With Assumption 5, however, it can be reformulated to a more accessible form. More precisely, it can be reformulated to a large-scale deterministic problem, the so-called *deterministic equivalent* of (MSLP) in *extensive form* (see [201]). To this end, let $\mathcal{S}$ denote the set of all (stage-T) scenarios. Then, for each scenario $s \in \mathcal{S}$ a separate copy x_t^s of variables x_t can be introduced, so that the optimization over implementable policies translates to an optimization over a finite number of decision variables. However, the problem size grows exponentially in the number of stages T. Therefore, even for a finite number of scenarios, this large-scale LP is too large to be solved by off-the-shelf solvers for all but very small instances.

A preferable solution approach is therefore to use tailored solution techniques which decompose (MSLP) into smaller subproblems. Note that from Assumption 7 and the definition of $\mathcal{X}_t(\cdot)$ in (2), it is evident that the constraints of (MSLP) are block-diagonal, as only consecutive stages are coupled in the constraints. This is visualized in Figure 3.

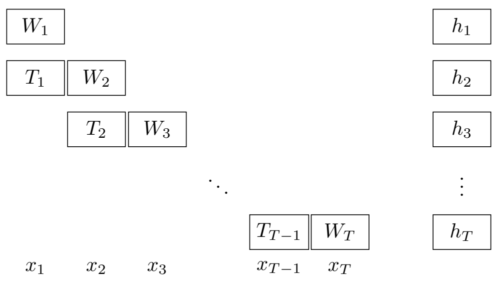


Figure 3: Block-diagonal structure of constraints in (MSLP).

This sequential and block-diagonal structure can be exploited to achieve the required decomposition. This is crucial for the derivation of SDDP. Interestingly, this decomposition idea directly leads to the second common modeling approach for our decision problem.

2.4 Dynamic Programming Equations

An alternative, but equivalent way to model (MSLP) is to exploit the well-known optimality principle by Bellman [13] and to formulate a recursion of so-called *dynamic programming equations* (DPE), where a multistage decision process with stagewise independent (or Markovian) uncertainty is modeled as a coupled sequence of optimization problems.

Whereas this modeling approach is often applied in stochastic programming as a way to reformulate and decompose the single problem (3) into a computationally tractable form, in dynamic programming it often serves as the starting point of modeling decision problems. However, in contrast to many approaches in dynamic programming we do not discretize x_t, see also Sect. 5.1.

Under Assumptions 1 to 9, for $t = T, \dots, 2$, the DPE are given by

$$Q_t(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t)\big)^\top x_t + \mathcal{Q}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t), \end{cases} \tag{4}$$

where

$$\mathcal{Q}_{t+1}(x_t) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1})\right] \tag{5}$$

and $\mathcal{Q}_{T+1}(x_T) \equiv 0$. $Q_t(\cdot,\cdot)$ is called *value function* and $\mathcal{Q}_t(\cdot)$ is called *expected value function*, *(expected) cost-to-go function*, *future cost function* or *recourse function*. For the first stage, we obtain

$$v^* = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathcal{Q}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{6}$$

For a formal proof of the equivalence of (3) and its DPE, we refer to [201] and Sect. 12. Importantly, in subproblem (4) x_t is a deterministic variable and not a function because a fixed realization of ξ_t is considered.

We should emphasize that the equivalence of (3) and its DPE does not require Assumption 5. This implies that also the DPE (4)-(6) are computationally intractable in case of general continuous random variables. While the subproblems are deterministic and finite-dimensional, there exist infinitely many value functions $Q_t(\cdot,\cdot)$ and the evaluation of $\mathcal{Q}_t(\cdot)$ requires the evaluation of (multidimensional) integrals. Therefore, also from this perspective Assumption 5 is crucial.

Remark 2.4 (Dynamic programming control perspective)**.** *Recall Remark 2.2. Using a distinction between state variables r_t and controls u_t, the DPE to (MSLP) can be formulated as*

$$Q_t(r_t, \xi_t) = \min_{u_t \in U_t(r_t, \xi_t)} f_t(u_t, \xi_t) + \mathcal{Q}_{t+1}(\mathcal{T}_t(r_t, u_t, \xi_t)). \tag{7}$$

Bellman Operator. In the French literature on SDDP, in addition to taking the optimal control perspective discussed in Remarks 2.2 and 2.4, a more formal way to define the DPE is prevalent, see [71, 119] for instance. To this end, a linear Bellman operator $\widehat{\mathfrak{B}}_t$ is introduced, which applied to some lower semicontinuous function $V : \mathbb{R}^{n_t} \to \mathbb{R} \cup \{+\infty\}$ is defined as [71]

$$\widehat{\mathfrak{B}}_t(V)(x_{t-1}, \xi_t) := \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)} \big(c_t(\xi_t)\big)^\top x_t + V(x_t), \tag{8}$$

i.e., it maps (x_{t-1}, ξ_t) to the optimal value of an optimization problem containing function $V(\cdot)$. We can then further define the operator

$$\mathfrak{B}_t(V)(x_{t-1}) := \mathbb{E}\Big[\widehat{\mathfrak{B}}_t(V)(x_{t-1}, \boldsymbol{\xi}_t)\Big]. \tag{9}$$

Setting V to $\mathcal{Q}_t(\cdot)$ for $t = 2, \dots, T$, the (expected) value functions can then be recursively defined in a very compact form. We summarize the different notations for a better overview:

$$\boxed{\begin{gathered}\widehat{\mathfrak{B}}_t(\mathcal{Q}_{t+1})(x_{t-1},\xi_t) = Q_t(x_{t-1},\xi_t) \\ \mathfrak{B}_t(\mathcal{Q}_{t+1})(x_{t-1}) = \mathcal{Q}_t(x_{t-1})\end{gathered}}$$

In the remainder of this work, we stick to notation (4), as it is most common in the literature on SDDP which we reference in this paper.

We obtain the following properties for the DPE which are standard for SDDP:

Lemma 2.5. *Under Assumptions 1 to 9, for the DPE defined by* (4)-(6) *the following properties hold:*

(a) We have relatively complete recourse, i.e., *for any $x_{t-1} \in \mathcal{X}_{t-1}$, the stage-t subproblem Equation* (4) *is feasible for all $\xi_t \in \Xi_t$.*

(b) The value functions $Q_t(\cdot,\cdot)$ and expected value functions $\mathcal{Q}_t(\cdot)$ are finite-valued on conv$(\mathcal{X}_{t-1})$ for all $t = 2,\dots,T$ and all $\xi_t \in \Xi_t$.

(c) Problem Equation (6) *is feasible and bounded.*

Remark 2.6. *In addition to Remark 2.2, we should highlight that (MSLP) (both, in single-problem formulation* (3) *and DPE* (4)-(6)*) can be straightforwardly enhanced with local decision variables $y_t \in Y_t$ and local constraints, not appearing in different stages. In principle, they can even be incorporated without changes to our models by extending the dimension of the (state) variables x_t and adapting the matrices T_t and W_t accordingly. However, as we explain in Sect. 4, the complexity of SDDP grows exponentially in the dimension of the state space, so this is computationally detrimental and should be avoided. Instead, purely local variables and constraints should be handled separately from the ones we introduced above. This approach is referred to as* generalized dual dynamic programming *(GDDP) in [18].*

While almost every practical application will require the introduction of these additional elements, in this work, for the most part we restrict to coupling variables and constraints which are required to illustrate the mechanics of SDDP.

Remark 2.7. *In general, the local objective functions may also include the states x_{t-1} instead of only depending on x_t and ξ_t. For notational simplicity, we consider a less general form of the objective function in this review.*

2.5 Approximations of the Value Functions

The main challenge in exploiting the DPE to solve (MSLP) is that the (expected) value functions are not known in analytical form in advance. The key idea in SDDP is to iteratively approximate them from below using linear functions, which are called *cutting-planes*, or short *cuts*. Together, these linear functions build polyhedral outer approximations $\mathfrak{Q}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$ for all $t = 2,\dots,T$, which we refer to as *cut approximations*. In that regard, SDDP can be considered as a special variant of Kelley's cutting-plane method [111] and closely related to Benders decomposition [17], see also Sect. 5.2. Note that in contrast to SDP this avoids a state discretization, as $Q_t(\cdot,\cdot)$ and $\mathcal{Q}_t(\cdot)$ do not have to be evaluated at all possible states, but only at well-chosen trial points where new cuts are constructed, cf. Sect. 5.1.

For this approximation by cuts, the following properties are crucial.

Theorem 2.8 ([26]). *Let $x_{t-1} \in conv(\mathcal{X}_{t-1})$. Then, under Assumptions 1 to 9, for all $t = 2, \dots, T$ and a given noise realization ξ_t, the value function $Q_t(\cdot, \xi_t)$*

(a) is piecewise linear and convex in (h_t, T_{t-1}),

(b) is piecewise linear and concave in c_t,

(c) is piecewise linear and convex in x_{t-1} on $conv(\mathcal{X}_{t-1})$.

The main idea here is that given the definition of $\mathcal{X}_{t-1}(\cdot)$ in (2), h_t, T_{t-1} and x_{t-1} do only appear in the right-hand side (RHS) of problem (4). Therefore, the dual feasible set is independent of those elements. It possesses finitely many extreme points. This assures piecewise linearity of $Q_t(\cdot, \cdot)$, as known from parametric optimization. The convexity follows with the linearity (Assumption 6) and all vectors and matrices being part of convex sets.

Theorem 2.8 directly implies the piecewise linearity and convexity of $\mathcal{Q}_t(\cdot)$.

Corollary 2.9 ([26]). *Under Assumption 5 and the premises of Theorem 2.8, for all $t = 2, \dots, T$, $\mathcal{Q}_t(\cdot)$ is piecewise linear and convex in x_{t-1} on $conv(\mathcal{X}_{t-1})$.*

Theorem 2.8 and Corollary 2.9 also directly imply the Lipschitz continuity of the (expected) value functions.

Corollary 2.10. *Under Assumptions 1 to 9, for all $t = 2, \dots, T$ and all $\xi_t \in \Xi_t$, $Q_t(\cdot, \xi_t)$ and $\mathcal{Q}_t(\cdot)$ are Lipschitz continuous on $conv(\mathcal{X}_{t-1})$.*

Replacing the true expected value functions with cut approximations in (4), we can define *approximate value functions*

$$\underline{Q}_t(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t)\big)^\top x_t + \mathfrak{Q}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t). \end{cases} \tag{10}$$

Trivially, for $\mathcal{Q}_{T+1}(\cdot) \equiv 0$, we have $\mathfrak{Q}_{T+1}(\cdot) \equiv 0$.

Note that apart from x_{t-1} and ξ_t, $\underline{Q}_t(\cdot, \cdot)$ is also a function of the cut approximation $\mathfrak{Q}_{t+1}(\cdot)$. This is especially relevant when these approximations are iteratively updated in SDDP, leading to different approximate value functions. Using the Bellman operators defined in (8)-(9) this can be expressed in a very concise way:

$$\underline{Q}_t(\cdot, \cdot) = \mathfrak{B}_t(\mathfrak{Q}_{t+1})(\cdot, \cdot).$$

Similarly, we could express this by adding an argument to $\underline{Q}_t(\cdot, \cdot)$, *i.e.*, by writing $\underline{Q}_t(x_{t-1}, \xi_t \,|\, \mathfrak{Q}_{t+1})$ or $\underline{Q}_t(\mathfrak{Q}_{t+1})(x_{t-1}, \xi_t)$. However, for notational simplicity, we do not state this explicitly, but when dealing with SDDP use the iteration index i for distinction. This means that $\underline{Q}_t^i(\cdot, \cdot)$ indicates that $\underline{Q}_t(\cdot, \cdot)$ is considered with cut approximation $\mathfrak{Q}_{t+1}^i(\cdot)$.

We summarize the different notations for a better overview:

$$\boxed{\begin{aligned} \widehat{\mathfrak{B}}_t(\mathfrak{Q}_{t+1})(x_{t-1}, \xi_t) &= \underline{Q}_t(x_{t-1}, \xi_t) \\ \mathfrak{B}_t(\mathfrak{Q}_{t+1})(x_{t-1}) &= \underline{\mathcal{Q}}_t(x_{t-1}) := \mathbb{E}_{\boldsymbol{\xi_t}}\big[\underline{Q}_t(x_{t-1}, \boldsymbol{\xi_t})\big] \end{aligned}} \tag{11}$$

Finally, we can observe that given that the cut approximations $\mathfrak{Q}_{t+1}(\cdot)$ are polyhedral, the approximate value functions $\underline{Q}_t(\cdot, \cdot)$ inherit the previously stated properties from $Q_t(\cdot, \cdot)$. In particular:

Lemma 2.11. *Let $\mathfrak{Q}_{t+1}(\cdot)$ be a polyhedral function. Then, under Assumptions 1 to 9, for all $t = 2, \dots, T$ and a given noise realization ξ_t, $\underline{Q}_t(\cdot, \xi_t)$ is piecewise linear and convex in x_{t-1} on $conv(\mathcal{X}_{t-1})$.*

On the other hand, as they are polyhedral, the cut approximations $\mathfrak{Q}_t(\cdot)$ for $t = 2, \dots, T$ are *nonlinear* functions. Importantly for computations, however, subproblems (10) can be formulated as LPs by using a partial epigraph reformulation and the fact that $\mathfrak{Q}_t(\cdot)$ is defined as the maximum of finitely many affine functions (modeled by some set $\mathcal{K}$ with $|\mathcal{K}| \in \mathbb{N}$):

$$\underline{Q}_t(x_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t, \theta_{t+1}} & \big(c_t(\xi_t)\big)^\top x_t + \theta_{t+1} \\ \quad \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \\ & -\,(\beta^i_{t+1,k})^\top x_t + \theta_{t+1} \geq \alpha^i_{t+1,k}, \quad \forall i \; \forall k \in \mathcal{K}. \end{cases} \tag{12}$$

This LP contains an additional decision variable θ_{t+1} and finitely many additional linear constraints indexed by i and k. The structure and indexing of these constraints become clear in the next section when we present the cut generation process for SDDP.

3 Standard SDDP

We are now able to introduce SDDP in its standard form.

3.1 Main Principle

SDDP consists of two main steps in each iteration i, a *forward pass* and a *backward pass* through the stages $t \in [T]$.

In each forward pass, using the approximate value functions $\underline{Q}^i_t(\cdot, \cdot)$ (recall that this implies using cut approximation $\mathfrak{Q}^i_{t+1}(\cdot)$ in (10)), a sequence of *trial points* $(x_t)_{t \in [T]}$ is generated, at which then new cuts are constructed in the following backward pass to improve the approximation. These trial points are also called *incumbents* or *candidate* solutions, and their sequence is called a *state trajectory* (especially in optimal control). The idea behind this approach is that the approximate value functions implicitly define a feasible (suboptimal) policy for problem (MSLP). The trial points are generated by evaluating this policy for one or several scenarios which are sampled from $\mathcal{S}$, *i.e.*, by solving the respective subproblems. This has the advantage that cuts are constructed at points which (at least for some scenario) are optimal given the current cut approximation. This step can also be interpreted as a Monte Carlo *simulation* of the current policy.

In the backward pass, dual information of the subproblems at the trial points is used to construct cuts, passing them back to the previous stage and updating $\mathfrak{Q}^i_t(\cdot)$ to $\mathfrak{Q}^{i+1}_t(\cdot)$ for all $t = 2, \dots, T$. This way, if not optimal, the current policy is amended (at least if the *right* scenario is sampled). In this step, also a *true* lower bound $\underline{v}$ for v^* is determined.

Remark 3.1 (Statistical learning perspective)**.** *The basic principle of SDDP can also be interpreted from a perspective of supervized learning as* learning *a policy (or expected value functions $\mathcal{Q}_t(\cdot)$ for all $t = 2, \dots, T$) or* training *a model of this policy (or cut approximations $\mathfrak{Q}_t(\cdot)$ for all $t = 2, \dots, T$) using backpropagation. In the forward pass* *the inputs are propagated through the stages using the current model, and in the backward pass cuts (representing the error of the current approximation) are propagated back through the stages to update the model.*

Algorithm 1 provides a pseudo-code for SDDP. We now provide a more detailed and technical look at the algorithmic steps.

3.2 Forward Pass

At the start of each iteration i, at first a subset $\mathcal{K} \subseteq \mathcal{S}$ of scenarios is sampled with $|\mathcal{K}| \ll |\mathcal{S}|$ (note that we may equivalently sample stage by stage during the forward pass). The number of samples $|\mathcal{K}|$ may vary by iteration, but we do not state this possible dependence explicitly. Traditionally, and most commonly, in SDDP some random sampling is used, but also a deterministic sampling is possible. We further discuss sampling techniques in Sect. 6.

Then, at the first-stage, the approximate subproblem

$$\min_{x_1 \in \mathcal{X}_1(x_0)} c_1^\top x_1 + \mathfrak{Q}_2^i(x_1). \tag{13}$$

is solved, which yields the trial point $x_1^i = x_1^{ik}$ for all $k \in \mathcal{K}$. Afterwards, for each stage $t = 2, \dots, T$ and each sample $k \in \mathcal{K}$, recursively the approximate value functions $\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k)$ are evaluated (this means that the subproblems (10) are solved for x_{t-1}^{ik}, ξ_t^k and the current cut approximation $\mathfrak{Q}_{t+1}^i(\cdot)$). This way, for each sample $k \in \mathcal{K}$, a sequence of trial points $(x_t^{ik})_{t \in [T]}$ is obtained.

The forward pass of SDDP is illustrated in Figure 4 for the recombining scenario tree from Figure 1 and $\mathcal{K} = \{1, 3, 9\}$, *i.e.*, $|\mathcal{K}| = 3$. The three sampled scenario paths are highlighted in green. The figure shows that for sample paths ξ^3 and ξ^9 the same node is reached at stage 3.

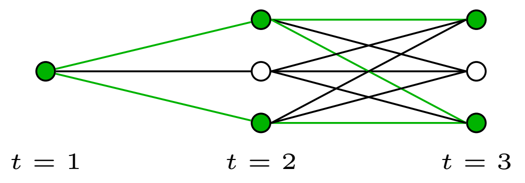


Figure 4: Illustration of SDDP forward pass for $|\mathcal{K}| = 3$.

Remark 3.2 (Initialization)**.** *Before the first backward pass, the cut approximations* $\mathfrak{Q}_t(\cdot)$ *are not initialized yet, so the forward pass subproblems* (13) *and* (10) *are not well-defined. This can be addressed using different strategies. First, in iteration* $i = 1$ *the forward pass can be skipped, and a feasible state trajectory* $(x_t^1)_{t \in [T]}$ *for the backward pass can be user-defined or taken randomly instead. Second, such a state trajectory can be computed heuristically via a greedy approach where* $\mathfrak{Q}_t^1(\cdot) \equiv 0$ *is assumed in the forward pass subproblems for all* $t \in [T]$*, so no coupling exists in the objective. Third, the cut approximations* $\mathfrak{Q}_t(\cdot)$ *may be initialized with a valid user-defined lower bound* $\underline{\theta}_t$ *for all* $t \in [T]$*. This approach is taken in the description in Algorithm 1.*

Algorithm 1 SDDP

```
Input: Problem (MSLP) satisfying Assumptions 1 to 9. Bounds $\underline{\theta}_t, t = 2, \dots, T$. Stopping criterion.
    Initialization
1:  Initialize cut approximations with $\theta_t \geq \underline{\theta}_t$ for all $t = 2, \dots, T$.
2:  Initialize lower bound with $\underline{v}^0 = -\infty$.
3:  Set iteration counter to $i \leftarrow 0$.
    SDDP Loop
4:  while Stopping criterion not satisfied do
5:      Set $i \leftarrow i + 1$.
        Forward Pass
6:      Sample a subset $\mathcal{K} \subseteq \mathcal{S}$ of scenarios.
7:      Solve the approximate first-stage problem (13) to obtain trial point $x_1^i = x_1^{ik}$ for all $k \in \mathcal{K}$.
8:      for stages $t = 2, \dots, T$ do
9:          for samples $k \in \mathcal{K}$ do
10:             Solve the approximate stage-$t$ subproblem (10) associated with $\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k)$ to obtain trial point $x_t^{ik}$.
11:         end for
12:     end for
        Backward Pass
13:     for stages $t = T, \dots, 2$ do
14:         for samples $k \in \mathcal{K}$ do
15:             for noise terms $j = 1, \dots, q_t$ do
16:                 Solve the updated approximate stage-$t$ subproblem (10) associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj})$. Store the optimal value and dual vector $\pi_t^{ikj}$.
17:             end for
18:             Use relations (16)-(17) and (19) to create an optimality cut for $\mathcal{Q}_t(\cdot)$.
19:             Update the cut approximation $\mathfrak{Q}_t^i(\cdot)$ to $\mathfrak{Q}_t^{i+1}(\cdot)$ using relation (18).
20:         end for
21:     end for
22:     Solve the approximate first-stage problem (20) to obtain a lower bound $\underline{v}^i$.
23: end while
Output: (Approximately) optimal feasible policy for (MSLP) defined by $x_1^i$ and cut approximations $\mathfrak{Q}_t^i(\cdot), t = 2, \dots, T$. $x_1^i$ defines an (approximately) optimal solution to problem (6) with $\overline{v}_{\mathcal{K}}^i \approx v^*$.
```

3.3 Backward Pass

Main Principle. The backward pass starts at stage T. Here, for all samples $k \in \mathcal{K}$, we consider subproblems (10) for the trial point x_{T-1}^{ik} computed in the forward pass, all noise realizations $\xi_{Tj}, j = 1, \dots, q_T$, and $\mathfrak{Q}_{T+1}^{i+1}(\cdot) \equiv 0$. That is, we consider functions $\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{tj})$ for $j = 1, \dots, q_T$.

As $\underline{Q}_T^{i+1}(\cdot, \xi_{Tj})$ is convex in x_{T-1} by Lemma 2.11, it can be underestimated by a linear function using some subgradient $\beta_{Tkj}^i \in \partial \underline{Q}_T^{i+1}(\cdot, \xi_{Tj})$ for any $j = 1, \dots, q_T$ and any $k \in \mathcal{K}$:

$$\underline{Q}_T^{i+1}(x_{T-1}, \xi_{Tj}) \geq \underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) + (\beta_{Tkj}^i)^\top (x_{T-1} - x_{T-1}^{ik}).$$

Since $\underline{Q}_T^{i+1}(\cdot, \xi_{Tj})$ is a lower approximation of the true value function $Q_T(\cdot, \xi_{Tj})$, this directly implies

$$Q_T(x_{T-1}, \xi_{Tj}) \geq \underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) + (\beta_{Tkj}^i)^\top (x_{T-1} - x_{T-1}^{ik}).$$

Taking expectations with respect to $\boldsymbol{\xi}_T$ on both sides, we obtain

$$\begin{aligned}
&\mathcal{Q}_T(x_{T-1}) \\
&\geq \mathbb{E}_{\boldsymbol{\xi}_T}\left[\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \boldsymbol{\xi}_T)\right] + \mathbb{E}_{\boldsymbol{\xi}_T}\left[(\boldsymbol{\beta}_{Tj}^i)^\top (x_{T-1} - x_{T-1}^{ik})\right] \\
&= \mathbb{E}_{\boldsymbol{\xi}_T}\left[\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \boldsymbol{\xi}_T) - (\boldsymbol{\beta}_{Tk}^i)^\top x_{T-1}^{ik}\right] + \Big(\mathbb{E}_{\boldsymbol{\xi}_T}\left[\boldsymbol{\beta}_{Tk}^i\right]\Big)^\top x_{T-1} \\
&= \underbrace{\sum_{j=1}^{q_T} p_{Tj} \left(\underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) - (\beta_{Tkj}^i)^\top x_{T-1}^{ik}\right)}_{=:\alpha_{Tk}^i} + \bigg(\underbrace{\sum_{j=1}^{q_T} p_{Tj} \beta_{Tkj}^i}_{=:\beta_{Tk}^i}\bigg)^\top x_{T-1},
\end{aligned} \tag{14}$$

where we exploit the finiteness of $\boldsymbol{\xi}_T$ (Assumption 5). α_{Tk}^i is called *cut intercept* and β_{Tk}^i is called *cut gradient.* Defining

$$\phi_{Tk}^i(x_{T-1}) := \alpha_{Tk}^i + (\beta_{Tk}^i)^\top x_{T-1},$$

we can express (14) as

$$\mathcal{Q}_T(x_{T-1}) \geq \phi_{Tk}^i(x_{T-1}). \tag{15}$$

Inequality (15) defines a cut for the expected value function $\mathcal{Q}_T(\cdot)$. Such a cut is constructed for each $k \in \mathcal{K}$. With these new cuts, the cut approximation $\mathfrak{Q}_T^i(\cdot)$ is updated to

$$\mathfrak{Q}_T^{i+1}(x_{T-1}) := \max\left\{\mathfrak{Q}_T^i(x_{T-1}),\ \phi_{T1}^i(x_{T-1}), \dots, \phi_{T|\mathcal{K}|}^i(x_{T-1})\right\}.$$

Thus, assuming that $|\mathcal{K}|$ does not change over the iterations, $\mathfrak{Q}_T^{i+1}(\cdot)$ consists of $i|\mathcal{K}|$ affine functions $\phi_{Tk}^i(\cdot)$, cf. formulation (12).

In the same way, for stages $t = T-1, \dots, 2$, cuts for $\mathcal{Q}_t(\cdot)$ can be constructed by solving subproblems (10) for the trial points x_{t-1}^{ik} computed in the forward pass and all noise realizations $\xi_{tj}, j = 1, \dots, q_t$. Importantly, by going backwards through the stages, at stage t we can already factor in the cuts that have been constructed at the following stage $t+1$, thus using a better approximation as the basis to construct a new cut. This means that we consider $\mathfrak{Q}_{t+1}^{i+1}(\cdot)$ and by that $\underline{Q}_t^{i+1}(\cdot, \cdot)$ with index $i+1$ in the backward pass of iteration i.

As for stage T, we obtain

$$\mathcal{Q}_t(x_{t-1}) \geq \underbrace{\sum_{j=1}^{q_t} p_{tj} \left(\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_t) - (\beta_{tkj}^i)^\top x_{t-1}^{ik}\right)}_{=:\alpha_{tk}^i} + \Bigg(\underbrace{\sum_{j=1}^{q_t} p_{tj}\beta_{tkj}^i}_{=:\beta_{tk}^i}\Bigg)^\top x_{t-1}, \tag{16}$$

where β_{tkj}^i denotes a subgradient of $\underline{Q}_t^{i+1}(\cdot, \xi_{tj})$ at x_{t-1}^{ik} for $k \in \mathcal{K}, j = 1, \ldots, q_t$. Again, by defining

$$\phi_{tk}^i(x_{t-1}) := \alpha_{tk}^i + (\beta_{tk}^i)^\top x_{t-1},$$

we can obtain a cut

$$\mathcal{Q}_t(x_{t-1}) \geq \phi_{tk}^i(x_{t-1}) \tag{17}$$

for each $k \in \mathcal{K}$ and can update the cut approximation to

$$\mathfrak{Q}_t^{i+1}(x_{t-1}) := \max\left\{\mathfrak{Q}_t^i(x_{t-1}),\ \phi_{t1}^i(x_{t-1}), \ldots, \phi_{t|\mathcal{K}|}^i(x_{t-1})\right\}. \tag{18}$$

Computing Subgradients. So far, we have discussed the main idea of the cut generation process in the backward pass of SDDP, which is based on evaluating approximate value functions $\underline{Q}_t^{i+1}(\cdot,\cdot)$ and using subgradients for them at trial points x_{t-1}^{ik}. For the interested reader, we now address in more detail how to compute those subgradients. This step uses *dual information*, *i.e.*, it is based on the duality theory of convex programs. For simplicity, we assume $X_t = \{x_t \in \mathbb{R}^{n_t} \ : \ x_t \geq 0\}$ for all $t \in [T]$.

Consider stage T, some $k \in \mathcal{K}$ and some $j \in \{1, \ldots, q_T\}$. Then, the dual problem to the linear stage-T subproblem (10) is

$$\begin{cases} \max\limits_{\pi_T} & \left(h_{Tj} - T_{T-1,j}x_{T-1}^{ik}\right)^\top \pi_T \\ \text{s.t.} & W_{Tj}^\top \pi_T \leq c_{Tj}. \end{cases}$$

Let π_T^{ikj} be an optimal dual basic solution. Such solution does always exist by relatively complete recourse and boundedness (see Assumption 9 and Lemma 2.5). By strong duality of linear programs, it follows

$$\begin{aligned} \underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) &= \left(h_{Tj} - T_{T-1,j}x_{T-1}^{ik}\right)^\top \pi_T^{ikj} \\ &= -(\pi_T^{ikj})^\top T_{T-1,j}x_{T-1}^{ik} + (\pi_T^{ikj})^\top h_{Tj}. \end{aligned}$$

Importantly, the dual feasible set does not depend on x_{T-1}, but remains unchanged for all trial points. In particular, π_T^{ikj} is always dual feasible, but not necessarily dual optimal for all x_{T-1}. Therefore, and because of minimization, it follows

$$\begin{aligned} \underline{Q}_T^{i+1}(x_{T-1}, \xi_{Tj}) &\geq -(\pi_T^{ikj})^\top T_{T-1,j}x_{T-1} + (\pi_T^{ikj})^\top h_{Tj} \\ &= -(\pi_T^{ikj})^\top T_{T-1,j}(x_{T-1} + x_{T-1}^{ik} - x_{T-1}^{ik}) + (\pi_T^{ikj})^\top h_{Tj} \\ &= \underline{Q}_T^{i+1}(x_{T-1}^{ik}, \xi_{Tj}) - (\pi_T^{ikj})^\top T_{T-1,j}(x_{T-1} - x_{T-1}^{ik}). \end{aligned}$$

Hence,

$$\beta^i_{Tkj} = -(\pi^{ikj}_T)^\top T_{T-1,j}$$

is a subgradient of $\underline{Q}^{i+1}_T(\cdot, \xi_{Tj})$ at x^{ik}_{T-1}.

The previous derivation provides some additional insight. Since the dual feasible set is polyhedral and does not depend on x_{T-1}, for each noise term $\xi_{Tj}, j = 1, \ldots, q_T$, there exist only finitely many dual extreme points (dual basic solutions) that can be attained. Therefore, only finitely many different cut coefficients can be generated. This is crucial for some convergence proofs of SDDP, as we discuss later.

For earlier stages $t = T-1, \ldots, 2$, the dual problem to subproblem (10) looks a bit more sophisticated, as the cut approximations $\mathfrak{Q}^{i+1}_{t+1}(\cdot)$ have to be taken into account, which requires additional dual multipliers ρ^r_t for all cuts $r \in \Gamma_{t+1}$, where Γ_{t+1} denotes the index set of cuts generated for the following stage. However, the derivation is completely analogous and, again, we arrive at

$$\underline{Q}^{i+1}_t(x_{t-1}, \xi_{tj}) \geq \underline{Q}^{i+1}_t(x^{ik}_{t-1}, \xi_{tj}) - (\pi^{ikj}_t)^\top T_{t-1,j}(x_{t-1} - x^{ik}_{t-1}),$$

so that

$$\beta^i_{tkj} = -(\pi^{ikj}_t)^\top T_{t-1,j} \tag{19}$$

is a subgradient of $\underline{Q}^{i+1}_t(\cdot, \xi_{tj})$ at x^{ik}_{t-1}. Interestingly, the optimal dual multipliers ρ^{rikj}_t are not explicitly required in this formula.

3.4 Bounds and Stopping

At the first stage, the subproblem

$$\underline{v}^i := \min_{x_1 \in \mathcal{X}_1(x_0)} c_1^\top x_1 + \mathfrak{Q}^{i+1}_2(x_1). \tag{20}$$

is solved. As $\mathfrak{Q}^{i+1}_2(\cdot)$ is a lower approximation of $\mathcal{Q}_2(\cdot)$, $\underline{v}^i$ is a valid lower bound to the optimal value v^* of (MSLP). This bound can be initialized with $\underline{v}^0 = -\infty$ or any a priori known lower bound for v^*.

In contrast, we are not guaranteed to obtain a valid upper bound for v^* during iterations of standard SDDP, as we only consider a small subset $\mathcal{K} \subseteq \mathcal{S}$ of all scenarios. This means that in the forward pass, the feasible policy for (MSLP), which is implicitly defined by the current cut approximations $\mathfrak{Q}^i_t(\cdot), t = 2, \ldots, T$, is only evaluated for a subset of all scenarios. By evaluating these scenarios in the objective of (MSLP) and taking the sample average

$$\overline{v}^i_{\mathcal{K}} := \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \underbrace{\sum_{t=1}^T \big(c_t(\xi^k_t)\big)^\top x^{ik}_t}_{=: v^i(\xi^k)} \tag{21}$$

we only obtain an unbiased estimator of the true upper bound $\overline{v}^i$ (a *statistical upper bound*) associated with the current policy, see Sect. 7 for more details.

After each iteration of SDDP, one or several stopping criteria are checked, which may or may not be based on $\overline{v}^i_{\mathcal{K}}$. We discuss different stopping criteria in detail in Sect. 7. If SDDP does not stop, a new iteration $i+1$ is started with a forward pass.

It is worth mentioning that the first-stage subproblems (13) and (20) are the same for consecutive backward and forward passes, and in principle only have to be solved once. The same is true for consecutive forward and backward pass problems at the final stage T.

3.5 Cut Properties

We discuss convergence of SDDP in Sect. 4. It relies on three key properties of the derived cuts:

Lemma 3.3. *For any stage $t = 2, \dots, T$ and any $k \in \mathcal{K}$, the functions $\phi_{tk}^i(\cdot)$ are*

(a) valid *lower approximations of $\mathcal{Q}_t(\cdot)$,*

(b) tight *for $\underline{\mathcal{Q}}_t^{i+1}(\cdot)$ (as defined in* (11)*) at x_{t-1}^{ik},*

(c) finite, *i.e., only finitely many different cuts can be generated, if we restrict to dual basic solutions to generate cuts.*

Proof. Property (a) follows immediately from (15) and (17). (b) holds because of strong duality for LPs and taking expected values over the obtained optimal values. Alternatively, we can rearrange the RHS of inequality (16) to obtain

$$\phi_{tk}^i(x_{t-1}) = \underline{\mathcal{Q}}_t^{i+1}(x_{t-1}^{ik}) + \sum_{j=1}^{q_t} p_{tj} (\beta_{tkj}^i)^\top (x_{t-1} - x_{t-1}^{ik}). \tag{22}$$

Inserting x_{t-1}^{ik} yields $\phi_{tk}^i(x_{t-1}^{ik}) = \underline{\mathcal{Q}}_t^{i+1}(x_{t-1}^{ik})$.

Property (c) follows by induction using the arguments on the dual feasible region previously discussed for stage T. □

Note that $\phi_{tk}^i(\cdot)$ is not necessarily tight for the true expected value function $\mathcal{Q}_t(\cdot)$ in early iterations for $t \neq T$, but rather might provide a loose cut only. However, by the finiteness and tightness properties it can be shown recursively, that eventually the derived cuts become tight for $\mathcal{Q}_t(\cdot)$ as well. In fact, after finitely many steps, the polyhedral function $\mathcal{Q}_t(\cdot)$ is represented exactly for all $t = 2, \dots, T$. This is a key property for the convergence of SDDP.

3.6 Illustrative Example

To illustrate the key steps of SDDP, we present a simple example.

Example 3.4. *Consider the 3-stage (MSLP)*

$$\begin{aligned} \min \quad & x_1 + x_2 + x_{31} + x_{32} \\ \text{s.t.} \quad & x_1 \leq 6 \\ & x_2 \geq \xi_2 - x_1 \\ & x_{31} - x_{32} = \xi_3 - x_2 \\ & x_1, x_2, x_{31}, x_{32} \geq 0, \end{aligned} \tag{23}$$

which is inspired by Example 2 in Chapter 5 of [26]. The uncertain data in the RHS is stagewise independent and uniformly distributed with $\xi_2 \in \{4, 5, 6\}$ and $\xi_3 \in \{1, 2, 4\}$.

Problem (23) *has not entirely the same structure as problem (MSLP), but can be easily converted to it by introducing slack variables. However, for illustrative purposes, we abstain from this. The problem can be expressed by means of the value functions*

$$Q_3(x_2,\xi_3) = \begin{cases} \min\limits_{x_3} & x_{31} + x_{32} \\ \text{s.t.} & x_{31} - x_{32} = \xi_3 - x_2 \\ & x_{31}, x_{32} \geq 0 \end{cases} \tag{24}$$

and

$$Q_2(x_1,\xi_2) = \begin{cases} \min\limits_{x_2} & x_2 + \mathcal{Q}_3(x_2) \\ \text{s.t.} & x_2 \geq \xi_2 - x_1 \\ & x_2 \geq 0. \end{cases}$$

The first-stage problem then is

$$v^* = \begin{cases} \min\limits_{x_1} & x_1 + \mathcal{Q}_2(x_1) \\ s.t. & x_1 \in [0,6]. \end{cases}$$

The optimal solution is given by $x_1^* = 3$ *with* $v^* = \frac{53}{9}$.

As shown in [26], the stage-3 value functions can be written in closed-form as $Q_3(x_2,\xi_3) = |\xi_3 - x_2|$ *for all scenarios. Taking expectations, a closed-form expression for* $\mathcal{Q}_3(\cdot)$ *can be derived, and by recursion we obtain*

$$\mathcal{Q}_2(x_1) = \begin{cases} \dfrac{23}{3} - \dfrac{16}{9}x_1, & x_1 \in [0,1] \\ \dfrac{67}{9} - \dfrac{10}{9}x_1, & x_1 \in [1,2] \\ \dfrac{59}{9} - \dfrac{10}{9}x_1, & x_1 \in [2,3] \\ \dfrac{47}{9} - \dfrac{2}{3}x_1, & x_1 \in [3,4] \\ \dfrac{31}{9} - \dfrac{2}{9}x_1, & x_1 \in [4,5] \\ \dfrac{7}{3}, & x_1 \in [5,6]. \end{cases}$$

The optimal value is $v^* = \frac{56}{9}$.

We apply SDDP for illustration. We assume loose initial bounds $\theta_2, \theta_3 \geq -10$ *for simplicity. In the forward pass, we sample one scenario path per iteration, i.e.,* $|\mathcal{K}| = 1$. *In iteration 1, let* $(\xi_2, \xi_3) = (5,4)$ *define this path. Solving the approximate subproblems* (10) *for all stages* $t = 1,2,3$ *and* $(\xi_2,\xi_3) = (5,4)$, *we obtain* $\overline{v}_{\mathcal{K}}^i = 6$. *In fact, this is no valid upper bound for* v^*.

In the backward pass, cuts for $\mathcal{Q}_t(\cdot), t = 2,3,$ *are derived at the trial points. For stage 3, the cut gradient is* $\beta_3(5) = 1$. *Moreover,* $\underline{\mathcal{Q}}_3^2(5) = \frac{8}{3}$. *With formulas* (17) *and* (22) *this yields the cut* $\mathcal{Q}_3(x_2) \geq -\frac{7}{3} + x_2$, *which is incorporated into the stage-2 subproblems. Solving these problems yields the cut* $\mathcal{Q}_t(x_1) \geq \frac{23}{3} - 2x_1$. *At the first stage, the lower bound computes to* $\underline{v}^1 = \frac{5}{3}$.

The expected value functions and the obtained cuts for three iterations are depicted in Figure 5. In the second and the third iteration, the same scenario path $(\xi_2,\xi_3) = (6,1)$ *is sampled in the forward pass.*

Figure 6 displays the bounds $\underline{v}^i$ and $\overline{v}^i_{\mathcal{K}}$ for ten iterations of SDDP. It shows that the lower bounds stabilize quickly at v^*, whereas the values of $\overline{v}^i_{\mathcal{K}}$ oscillate around v^*.

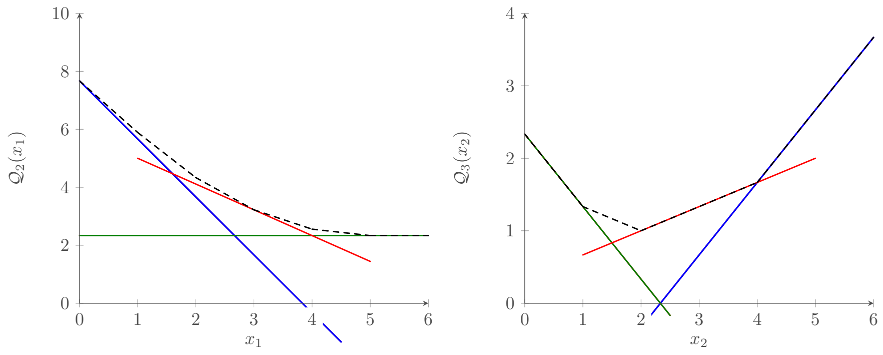


Figure 5: Expected value functions for Example 3.4 with cuts obtained in first three iterations depicted in blue, green and red.

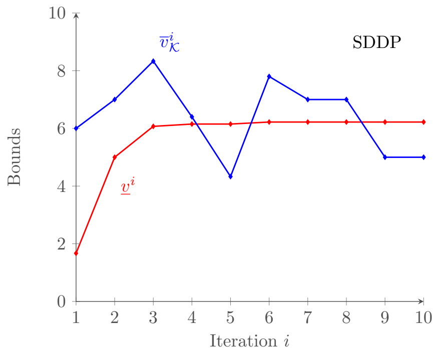


Figure 6: Bounds for 10 iterations of SDDP applied to Example 3.4.

3.7 Policy Assessment

As mentioned before, in standard SDDP no valid upper bound $\overline{v}$ for v^* is determined. While in each iteration a statistical upper bound (21) can be computed, the number of samples $|\mathcal{K}|$ may often be too small to appropriately assess the quality of the current policy. In particular, $|\mathcal{K}|$ is often chosen to be 1 in practice, and thus $\overline{v}^i_{\mathcal{K}}$ is not a meaningful estimate for $\overline{v}$.

Therefore, to assess the obtained policy, usually an additional forward simulation is conducted once SDDP has terminated. For this simulation a much higher number of sample paths through the scenario tree is used, *e.g.* $|\mathcal{K}| \in \{1000, 10000\}$, leading to a reasonable estimator $\overline{v}_{\mathcal{K}}$. In this step, the simulation can be either performed *in-sample* (using sample paths through the recombining scenario tree) or *out-of-sample* (using the true underlying distribution, *e.g.*, if $\boldsymbol{\xi}_t$ is a continuous random variable that is discretized to satisfy Assumption 5, see Sect. 11).

Remark 3.5. *In the light of Remark 3.1 this policy assessment step can also be interpreted from a statistical learning perspective. After the model has been trained, a* *model validation (using in-sample data) or a model test (using out-of-sample data) are performed.*

4 Convergence and Complexity

The convergence behavior of SDDP has been thoroughly analyzed over the years. We discuss the main convergence results in this section. We first focus on *finite* convergence of SDDP, and then discuss the actual convergence *rate*, *i.e.*, the computational complexity of SDDP. Our overview is loosely based on the review chapter in [71].

4.1 Finite Convergence

The first convergence analyses related to SDDP have been conducted in [38] and [125], however implicitly assuming independence of sampled random variables and convergent subsequences of algorithm iterates. A first complete convergence proof is given by Philpott and Guan in [164] for the case where uncertainty only enters the RHS of (MSLP) (in fact, they consider a more general algorithm than SDDP, including sampling in the backward pass). The same reasoning is used by Shapiro [198] for the case where also W_t, c_t and T_{t-1} are uncertain.

The convergence behavior of SDDP can be explained using two main arguments: First, as stated in Lemma 3.3, only finitely many different cuts, and by that only finitely many different cut approximations $\mathfrak{Q}_t(\cdot)$ can be constructed for all $t = 2, \dots, T$. This result requires linearity (Assumption 6) and finite random variables (Assumption 5). Moreover, these finitely many cuts also satisfy some tightness property, which implies that they are sufficient to exactly represent the polyhedral (expected) value functions (see Theorem 2.8). For a deterministic algorithm, this would result in finite convergence to the true optimal point and value (see the convergence properties of Benders decomposition [17] and Kelley's cutting-plane method [111]).

For SDDP, it has to be taken into account that scenarios are sampled in the forward pass. This means that the cut approximations might not further improve for some iterations if the *wrong* scenarios are sampled. Therefore, the second key argument for many proofs of finite convergence of SDDP is that each scenario is visited infinitely many times with probability 1 given that the algorithm does not terminate. Intuitively, this means that after finitely many iterations the *right* scenarios will be sampled with probability 1, leading to the construction of a new cut. This requirement is satisfied under *independent sampling*, that is, if the sampling in the forward pass of Algorithm 1 is random and independent of previous iterations. It is also satisfied for an exhaustive enumeration of all scenarios in the sampling process. We should emphasize that this argument is purely theoretical in order to establish convergence results for SDDP. When applying SDDP in practice, it is usually not even possible to sample each scenario *once* in reasonable time.

Using these two arguments, the following main convergence result can be obtained

Theorem 4.1 (Almost sure finite convergence of SDDP)**.** *Under Assumptions 1 to 9 and using an independent random sampling procedure in the forward pass, SDDP converges with probability 1 to an optimal policy of (MSLP) in a finite number of iterations.*

Importantly, almost sure finite convergence to an optimal policy of (MSLP) does not imply that the trajectories $(x_t^{ik})_{t\in[T]}, k \in \mathcal{K}$, and the corresponding sample averages

$\overline{v}^i_{\mathcal{K}}$ obtained in SDDP converge, as both are random and depend on the current sample $\mathcal{K}$. However, the lower bounds $\underline{v}^i$ obtained in SDDP converge to v^*.

Deterministic Sampling. Recently, convergence analyses of SDDP and related algorithms have often made use of deterministic sampling techniques instead of random sampling [10, 11]. Here, the idea is that the approximation error in SDDP can be controlled and guided to zero in a deterministic way if in each iteration scenarios are sampled for which the current approximation gap is maximized. This requires, however, that the approximation gap itself can be bounded rigorously. Therefore, in addition to the lower cut approximation $\mathfrak{Q}_t(\cdot)$ also an upper approximation $\overline{\mathfrak{Q}}_t(\cdot)$ is constructed and iteratively refined [10, 231], so that deterministic lower bounds $\underline{v}^i$ and upper bounds $\overline{v}^i$ are computed in each iteration. For more details on deterministic sampling and deterministic upper bounds we refer to Sect. 6 and 8.

Generalizations. It has been shown that some of the basic assumptions (Assumptions 1 to 9) can be relaxed without compromising convergence of SDDP. Girardeau et al. [78] analyze the case where SDDP is applied to multistage problems with nonlinear convex subproblems, *i.e.*, Assumption 6 is relaxed. In this case, the value functions $Q_t(\cdot)$ are no longer polyhedral, but still convex. The authors show that almost sure convergence is still satisfied as long as some convexity and compactness assumptions and some tightened recourse assumption are satisfied. We discuss this result in detail in Sect. 15 when we formally introduce convex multistage stochastic nonlinear problems. The main idea is that even without polyhedrality, $Q_t(\cdot)$ can be guaranteed to be Lipschitz continuous, so that the approximations of $\mathcal{Q}_t(\cdot)$ get better in a whole neighborhood of the trajectories $(x_t^{ik})_{t\in[T]}, k \in \mathcal{K}$.

Guigues generalizes this result to the risk-averse case where Assumption 8 is relaxed [85]. Forcier and Leclère prove convergence for (MSLP) without finite randomness, *i.e.*, dropping Assumption 5. Further convergence proofs are provided for multi-cut SDDP [8], SDDP with cut selection [8, 87], adaptive partition-based SDDP [207] (see also Sect. 21), using SDDP with saddle cuts [55] (see also Sect. 14) and variants of distributionally robust SDDP [65, 162] (see also Sect. 13), Another proof of almost sure finite convergence for extensions to non-convex problems is provided in [231].

4.2 Complexity

Theorem 4.1 guarantees almost sure finite convergence of SDDP. While this result is of theoretical interest, it may not be very relevant in practical applications, as it provides no result on the rate of convergence. As pointed out in [71] and mentioned before, especially the argument of scenarios being sampled repeatedly (infinitely many times) is almost never applicable to SDDP in practice due to the sheer amount of scenarios in $\mathcal{S}$. Important for the rate of convergence are the computational cost per iteration and the required number of iterations.

Cost per Iteration. For the computational cost per iteration, the number of LPs to be solved in the backward pass is crucial. Per sample $k \in \mathcal{K}$ in the forward pass, q_t subproblems are solved for each stage except for $t = 1$ in the backward pass. Therefore, the total number of LPs solved is $1 + |\mathcal{K}| \sum_{t=2}^{T} q_t$. Hence, the number of problems to be solved grows linearly in the number of stages T, in the number of samples $|\mathcal{K}|$ and in the number of noise terms q_t [179].

Expected Number of Iterations. The computational bottleneck for SDDP is the expected required number of iterations to achieve convergence. Recently, there has been active research on computing theoretical bounds on this number, with Lan

[117] as well as Zhang and Sun [231] publishing similar results using slightly different approaches. In both cases, the authors start by considering some case of deterministic sampling (in [117] the associated algorithm is referred to as *explorative dual dynamic programming* (EDDP)) before enhancing their results to the random sampling variant of SDDP. We discuss deterministic sampling in more detail in Sect. 6. The main idea to derive iteration bounds is the following: By exploiting Lipschitz continuity of $\mathfrak{Q}_t(\cdot)$ and $\mathcal{Q}_t(\cdot)$, it is possible to control the approximation error also at points where no cuts are constructed, as long as they lie in a neighborhood of some trial point x_t^{ik}. If the state space is bounded for all $t \in [T]$ (cf. Assumption 9), it can be completely covered by finitely many such neighborhoods [231]. A similar reasoning is applied in [71].

More formally, Lan [117] introduces the notion of *saturated points* $\bar{x}_{t-1}$, in which the approximation of $\mathcal{Q}_t(\cdot)$ is already ε-close for some predefined tolerance $\varepsilon > 0$, *i.e.*,

$$\mathcal{Q}_t(\bar{x}_{t-1}) - \mathfrak{Q}_t^i(\bar{x}_{t-1}) \leq \varepsilon,$$

and *distinguishable points* $\bar{x}_{t-1}$, which have at least a δ-distance to the set X_{t-1}^{sat} of already saturated points for some $\delta > 0$, that is

$$\|\bar{x}_{t-1} - x_{t-1}\| > \delta, \quad \forall x_{t-1} \in X_{t-1}^{sat}.$$

If some trial point x_t^{ik} is saturated and distinguishable, the iteration i can be called *effective* [71]. Using deterministic sampling, all iterations in SDDP can be shown to be effective, and thus the number of iterations can be bounded in the aforementioned way. For random sampling, this is not true, but the probability for an effective iteration is at least $\frac{1}{N}$ with $N := \Pi_{t=2}^{T-1} n_t$.

In the light of Assumption 9 (b), for any $t \in [T]$, we call the bound D_t satisfying

$$\|x_t - x_t'\| \leq D_t, \ \forall x_t, x_t' \in \mathcal{X}_t$$

the *diameter* of the state space. Additionally, let L denote a Lipschitz constant for the objective function of (MSLP), which exists due to Corollary 2.10.

Then, the following complexity results are satisfied by SDDP.

Theorem 4.2 (Complexity of SDDP [117, 231])**.** *Let $D_t \leq D$ for all $t \in [T]$. For some $\varepsilon > 0$, the (expected) number of required iterations of SDDP (Algorithm 1) to obtain*

- *an ε-optimal solution using deterministic sampling is*
  - *polynomial in $T, (\frac{1}{\varepsilon}), L$ and D,*
  - *exponential in n_t,*
- *an ε-optimal solution using deterministic sampling, given that $\mathcal{X}_t$ is finite with cardinality $|\mathcal{X}_t| \leq \overline{X}$, is*
  - *linear in T and $\overline{X}$*
- *a $(T\varepsilon)$-optimal solution using deterministic sampling is*
  - *linear in T,*
  - *polynomial in $T, (\frac{1}{\varepsilon}), L$ and D,*
  - *exponential in n_t,*
- *an ε-optimal solution using random sampling is*
  - *polynomial in $q_t, (\frac{1}{\varepsilon}), L$ and D,*

– *exponential in T and n_t.*

This means that for standard SDDP (using random sampling) the expected number of iterations grows exponentially in the horizon T and the dimension n_t of the state space. This is computationally important. The exponential complexity with respect to the state dimension is not that surprising, as it is well-known for cutting-plane methods [146] and inherited by SDDP. Similarly, the exponential complexity with respect to the number of stages directly follows from the exponential number of scenarios that may have to be sampled in the worst-case. Interestingly, under deterministic sampling, the complexity is independent of the number q_t of noise terms per stage, as this number only affects the computational cost per iteration.

We see that using some deterministic sampling scheme a polynomial or even linear iteration complexity in T can be achieved, whereas the iteration complexity in the state space cannot be alleviated [231].

The complexity results in [117, 231] have been further generalized by Forcier and Leclère [71]. They provide results for a generalized framework of SDDP-related algorithms, including SDDP with inexact cuts or regularization (see also Sect. 21), risk-averse SDDP (see also Sect. 12) and extensions to convex nonlinear or non-convex mixed-integer (nonlinear) problems (see also Sect. 15 and 16).

5 Comparison with Related Methods

We briefly compare SDDP to solution methods that it is (historically) related to, as discussed in Sect. 1.

5.1 Relation to SDP

SDDP is closely related to stochastic dynamic programming (SDP). SDP usually is applied in a setting where not only state variables, but additional local variables are considered, see Remarks 2.2 and 2.4. Therefore, the DPE and value functions are considered in the form of (7), which we repeat here for convenience:

$$Q_t(x_{t-1}, \xi_t) = \min_{u_t \in U_t(x_{t-1}, \xi_t)} f_t(u_t, \xi_t) + \mathcal{Q}_{t+1}(\mathcal{T}_t(x_{t-1}, u_t, \xi_t)).$$

The main idea of SDP is to explicitly evaluate the (expected) value functions for all possible cases during a forward or backward iteration through the stages $t \in [T]$. This is only possible if the support Ξ_t of $\boldsymbol{\xi}_t$ and the state space $\mathcal{X}_t \subset X_t$ are finite for all $t \in [T]$. Otherwise, infinitely many evaluations would be required. Additionally, it is required that also the action space $U_t(x_{t-1}, \xi_t)$ is finite for all $x_{t-1} \in X_{t-1}, \xi_t \in \Xi_t$, so that the minimum in (7) can be computed by finitely many evaluations. For this reason, all these sets may have to be discretized first [169].

The computational effort of SDP scales linearly in T and in the cardinalities $|X_t|$, $|U_t(x_{t-1}, \xi_t)|$ and $|\Xi_t|$. The three sets might be multidimensional, and thus require to be discretized in each dimension n_t, $\tilde{n}_t$, and κ_t. Hence, their cardinality grows exponentially in these dimensions, which is computationally prohibitive for high-dimensional problems. This is known as the curse of dimensionality of SDP, see also Sect. 1.

SDDP avoids the requirements of state space and action space discretization by not evaluating $\mathcal{Q}_t(\cdot), t \in [T]$, exactly for all (finitely many) possible actions and states, but approximating them by an iteratively refined polyhedral outer approximation $\mathfrak{Q}_t(\cdot)$, constructed by linear cuts. It can thus be considered an *approximate dynamic programming* (ADP) method.

5.2 Relation to NBD

In stochastic programming, it is common practice to consider problems (MSLP) with finite randomness (Assumption 5), but without the requirement of stagewise independence of $\boldsymbol{\xi}_t$ (Assumption 2). In that case the uncertainty can be modeled by a finite scenario tree, which compared to the recombining tree from Sect. 2 exhibits some path dependence and satisfies the usual *tree* property that each node n has a finite set of child nodes $\mathcal{C}(n)$, but a unique parent node $a(n)$. An example of a scenario tree with $T = 3$ and $|\mathcal{S}| = 9$ is illustrated in Figure 7. This scenario tree represents the same number of scenarios $|\mathcal{S}|$ as the recombining one in Figure 1, but requires $\sum_{t=2}^{T} q_t^{t-1} + 1$ instead of $\sum_{t=2}^{T} q_t + 1$ nodes.

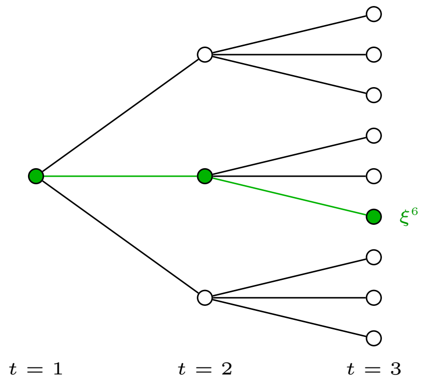


Figure 7: Scenario tree with 9 scenarios and ξ^6 highlighted.

To solve (MSLP) associated with a general scenario tree, in principle the same approximation approach as in SDDP can be used. However, due to the path dependence, the value functions $Q_t(\cdot, \cdot)$ and expected value function $\mathcal{Q}_t(\cdot, \cdot)$ depend on the history $\xi_{[t-1]}$ of the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$. In other words, each node n has its own value function $Q_n(\cdot)$, and with each node (except for leaf nodes) is associated an expected value function $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$. Therefore, to update the approximations $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$ of all $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ in each iteration, all nodal subproblems have to be solved in the backward pass, which in turn requires to compute trial points $x^i_{a(n)}$ for all nodes, *i.e.*, solving all nodal subproblems in the forward pass as well.

Because of its close relation to the L-shaped method for solving two-stage stochastic linear programs [228] and to Benders decomposition [17] this solution method is called *nested Benders decomposition* (NBD) or just *nested decomposition.* It was first proposed by Birge in 1985 [25] and can be interpreted as a decomposition method for the extensive form of the deterministic equivalent of (MSLP). Contrary to SDDP, NBD guarantees that valid lower bounds $\underline{v}$ and upper bounds $\overline{v}$ of v^* are determined in each iteration and by that allows for a deterministic stopping criterion in a straightforward way. The upper bounds can be computed as

$$\overline{v}^i := \mathbb{E}\left[\sum_{n\in\mathcal{T}} c_n^\top x_n^i\right],$$

where $\mathcal{T}$ is the set of all nodes in the scenario tree.

On the other hand, due to the sheer amount of subproblems to be solved in each iteration, which grows exponentially in T, it is only computationally tractable for problems of moderate size. By moderate we mean instances with some hundreds or a few thousand scenarios, and 4 or 5 stages at maximum [227].

Furthermore, for general scenario trees also sampling scenarios from $\mathcal{S}$ in the forward pass does not necessarily help to reduce the computational burden and to speed-up the solution process, as it reduces the computational effort per iteration, but at the same time implies that the cut approximations $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$ are only improved for some $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ in each iteration.

Under stagewise independence (Assumption 2) this is different. The scenario tree collapses to a recombining tree. This means that for any stage t, many differing scenarios share the same nodes, and thus value functions. In particular, there exists only one expected value function $\mathcal{Q}_t(\cdot)$ for each $t = 2, \dots, T$. Therefore, even if only a sample $\mathcal{K} \subset \mathcal{S}$ of scenarios is considered in each iteration i, still the cut approximations $\mathfrak{Q}^i_t(\cdot)$ for all $\mathcal{Q}_t(\cdot)$ are updated with new cuts.

From this perspective, SDDP can be interpreted as a sampling variant of NBD which reduces the computational effort per iteration significantly [179], but heavily relies on stagewise independence of $(\boldsymbol{\xi}_t)_{t\in[T]}$ in order to leverage the sampling with respect to value function approximations.

Remark 5.1 (Cut-sharing)**.** *In the literature, the aforementioned property of SDDP is often referred to as* cut-sharing*. This is best understood by representing the stagewise independent data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ using a standard scenario tree. In this case, at any stage t, all nodes have the same set of successor nodes. If now only a sample $\mathcal{K}$ of scenarios is considered in iterations of SDDP, only a subset of nodes is visited. Nonetheless, the cuts constructed for a specific node are valid for all equivalent nodes in the tree as well, so they are* shared *with other nodes/scenarios.*

As mentioned above, a recombining scenario tree provides a more precise picture. For each stage, scenarios share nodes *in the recombining tree and there exists only* one *function $\mathcal{Q}_t(\cdot)$ to be approximated, so there is no need to actually share cuts. Therefore, the phrase* cut-sharing *is sometimes considered misleading.*

5.3 Complexity Comparison

We summarize the main complexity results for SDDP and the related methods in Table 3.

In contrast to SDP, SDDP does not require a state space and action space discretization. Especially, the latter is computationally important in practice, whereas the former may yield some computational improvements, but at least does not translate into an improvement of the worst-case complexity class. On the other hand, SDDP does not have linear complexity in T.

Another difference is that SDDP (as NBD and most solvers for the deterministic equivalent) approximates v^* with improving lower (and upper) bounds. This means that if the computation time is increased also the quality of the approximation improves. On the contrary, standard solution methods for SDP, such as backward induction, either manage to solve a problem in a given time limit or not, but do not use improving approximations. In particular, stopping SDP prematurely does not provide valid bounds for v^*.

Table 3: Complexity of SDDP and related solution methods.

	Det. Equiv.	NBD	SDDP	SDP
Requirements				
stagewise independence	no	no	yes	yes*
state discretization	no	no	no	yes
action discretization	no	no	no	yes
Complexity				
in T	exponential	exponential	exponential	linear
in n_t	polynomial	exponential	exponential	exponential
in q_t	polynomial	polynomial	polynomial	linear
Progressivity				
of bounds	yes	yes	yes	no

* Markovian uncertainty is possible as well.

Compared to NBD, SDDP mainly reduces the computational effort per iteration significantly, but does not get rid of the exponential growth of the computational cost with respect to T. In return, it heavily relies on stagewise independence (Assumption 2) and has worse complexity with respect to the state dimension n_t.

We can conclude that SDDP, while mitigating some of the weaknesses of SDP and NBD (sometimes advertized as "breaking the curse of dimensionality"), does not manage to leave the respective worst-case complexity classes. On the contrary, it inherits some of the complexity drawbacks of both methods. Still, in many applications (where not worst-case complexity is decisive) it shows considerable performance improvements compared to SDP and NBD, especially for problems with continuous action space, a medium number of stages T and a moderate state dimension n_t. While Theorem 4.2 indicates that convergence may take extremely long in large-scale applications, and too long to be computationally tractable, SDDP has shown good performance for large-scale instances of (MSLP) in many applications, as we discuss in Sect. 9. This is also due to various improvements, which we address in the following sections.

6 Sampling

Sampling is a central element of SDDP, see Sect. 3. In the forward pass, a finite number $|\mathcal{K}|$ of scenarios is sampled to simulate the current policy and compute a trajectory of trial points $(x_t^{ik})_{t\in[T]}$ for all $k \in \mathcal{K}$. Often, this sampling is done from a finite set of scenarios $\mathcal{S}$ (see Assumption 5), with $|\mathcal{K}| \ll |\mathcal{S}|$. Alternatively, it is possible to directly sample from a given (continuous) distribution.

In this section, we discuss different sampling techniques which can be used in SDDP. As indicated in Sect. 3 and 4, we can distinguish between *random sampling* and *deterministic sampling* methods. In standard SDDP, as originally proposed in [152], random sampling is used. Here, the main requirement is that the samples should be independent and identically distributed (i.i.d.). This is important for two reasons:

(1) This way, almost sure finite convergence of SDDP can be ensured, as any scenario is sampled infinitely many times with probability 1, assuming that the algorithm does not terminate, see Sect. 4.

(2) In the originally proposed stopping criterion of SDDP a confidence interval is used, which is built using the sample mean $\overline{v}^i_{\mathcal{K}}$ (21), see Sect. 7. However, by the Central Limit Theorem, even an approximate confidence interval can only be obtained for a sequence of i.i.d. random variables.

6.1 Monte Carlo Sampling

The simplest sampling method satisfying the above requirement is Monte Carlo (MC) sampling. Here, samples are drawn randomly from the probability distribution of $\boldsymbol{\xi}_t$ in each iteration, by first sampling from a uniform distribution and then using appropriate transforms. Under stagewise independence (Assumption 2), this is done independently for each stage $t \in [T]$.

As the quantities $v^i(\xi^k)$ are i.i.d., the value $\overline{v}^i_{\mathcal{K}}$ (21) that can be computed in the SDDP forward pass is an unbiased estimator of $\overline{v}^i$ and according to the Strong Law of Large Numbers converges to $\overline{v}^i$ for $|\mathcal{K}|$ approaching infinity. Still, the sampling error can be significant. The variance of $\overline{v}^i_{\mathcal{K}}$ can be estimated by $\frac{1}{|\mathcal{K}|}\left(\sigma^i_{\overline{v},\mathcal{K}}\right)^2$. This means that the variance can be reduced either by increasing the number of samples $|\mathcal{K}|$ or by reducing the sample variance $\left(\sigma^i_{\overline{v},\mathcal{K}}\right)^2$. Increasing the sample size may look promising at first glance, but may become computationally intractable in practice [150]. Recall that for every sample $k \in \mathcal{K}$ a number of $1 + \sum_{t=2}^T q_t$ subproblems has to be solved in the backward pass of each iteration. Therefore, the more promising approach is combining MC sampling with variance reduction techniques [150].

6.2 Variance Reduction Techniques

Incorporating variance reduction techniques into sampling in SDDP is studied extensively in [105, 150]. For a review on sampling techniques in stochastic programming in general, we refer to [104].

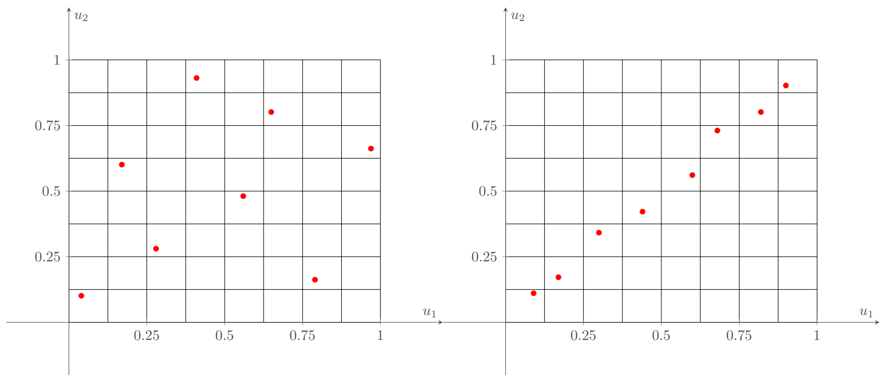
Randomized QMC Sampling. In [105], it is proposed to use Quasi-Monte Carlo (QMC) sampling within SDDP. In this case, instead of randomly sampling from the uniform distribution, a deterministic sequence of points $u^1, \dots, u^N$ from $(0,1)^{\kappa_t}$ is chosen. This is done in such a way that the sampled points fill $(0,1)^{\kappa_t}$ as homogeneously as possible (so the empirical distribution is as close to a uniform distribution as possible). Then, after an appropriate transformation, they provide a better representation of $\boldsymbol{\xi}_t$ than randomly sampled points.

A drawback of QMC methods is that the sample points are not random, the obtained estimator is biased and no confidence interval can be established. *Randomized* QMC (RQMC) methods, where the choice of QMC points is combined with some kind of randomness, avoid this drawback and allow for standard error estimation [105].

Compared to MC sampling, RQMC methods achieve better convergence rates of $\mathcal{O}\big(|\mathcal{K}|^{-1}(\log|\mathcal{K}|)^{\kappa_t}\big)$, and thus are considered more efficient. However, the convergence rate depends on the dimension κ_t of $\boldsymbol{\xi}_t$ [105].

Latin Hypercube Sampling. In Latin Hypercube Sampling (LHS) [141], the space $(0,1)^{\kappa_t}$ is divided into equidistant subintervals and then scenarios are sampled from each subinterval in such a way that in each row and column of the grid only one point is sampled. This is illustrated in Figure 8 (a).

In this way, again, a more homogeneous distribution of the sample points can be obtained, and compared to MC sampling, the variance can be reduced. On the flipside, poor space-filling or correlation between the sample points has to be ruled out, see Figure 8 (b), which requires significant additional effort.



(a) Good space-filling. (b) Correlated sample points.

Figure 8: Latin Hypercube Sampling for two dimensions.

Incorporation into SDDP. It is important to notice that while reducing the variance compared to the classical MC estimators, scenarios sampled by RQMC and LHS are no longer i.i.d. Therefore, both sampling techniques cannot be incorporated into SDDP without modification, if convergence properties should not be compromised. Homem-de-Mello et al. [105] therefore suggest to build sampling blocks. This means that the total number of samples $|\mathcal{K}|$ is divided into M blocks $\ell = 1, \dots, M$ with $M \geq 5$ a divisor of $|\mathcal{K}|$. Then, for each block ℓ, $|\mathcal{K}'| := |\mathcal{K}|/M$ scenarios are obtained using conditional sampling with RQMC or LHS, which are not independent. For each $k' \in \mathcal{K}'$, values $v^i(\xi^{k'})$ are determined and averaged to $\overline{v}^{i,\ell}$.

This is repeated for each block ℓ. Then, the mean $\overline{v}^i_{\mathcal{K}}$ of all values $\overline{v}^{i,\ell}, \ell = 1, \dots, M$, and the sample variance are determined. As the scenarios of different blocks are independent, this still yields a useful confidence interval to stop the algorithm.

Another challenge reported in [105] is that it is computationally expensive to generate samples using RQMC for high dimensions. To reduce the computational effort, it may be reasonable to apply RQMC only to important components, *e.g.*, to early stages in $[T]$, and standard MC or LHS to the other ones. This strategy is called *padding* and applied after 6 or 12 stages for numerical tests in [105].

Experiments in [105] imply that RQMC and LHS both lead to upper bounds $\overline{v}_{\mathcal{K}}$ oscillating around the lower bound $\underline{v}$ more quickly compared to MC sampling.

6.3 Importance Sampling

In [150], Parpas et al. propose incorporating importance sampling into SDDP. In contrast to the previously described techniques, it can be used to obtain i.i.d. samples in the forward pass.

The main idea of importance sampling in general is to attach different importance to subregions of the sample space and to sample more often from subregions of higher importance. In the context of SDDP, this means that it is sampled with priority from scenarios that contribute more to the value of the expected value functions $\mathcal{Q}_t(\cdot)$.

This is achieved by sampling from a different distribution than the original one, the so-called *importance sampling distribution*, but correcting the bias introduced by this difference. Then, an importance sampling estimate of $\overline{v}$ can be calculated as

$$\overline{v}_{\mathcal{K}}^{IS,i} := \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} v^i(\xi^k) \Lambda(\xi^k)$$

with $\Lambda(\boldsymbol{\xi}) := \frac{f(\boldsymbol{\xi})}{g(\boldsymbol{\xi})}$, where f denotes the original distribution and g the importance sampling distribution. The likelihood function $\Lambda(\cdot)$ is used to correct for sampling from the wrong distribution. It can be shown that importance sampling can reduce the variance of sampling estimators significantly. In the SDDP case, as shown in [150], the variance is minimized for the choice

$$g_t^*(\xi_t) := \frac{|Q_t(x_{t-1}^{ik}, \xi_t)|}{\mathbb{E}_f |Q_t(x_{t-1}^{ik}, \boldsymbol{\xi}_t)|} f_t(\xi_t).$$

However, clearly, this *zero-variance distribution* is a theoretical construct and not known, which is referred to as the *curse of circularity.* Therefore, it is proposed to first approximate g^* using a framework including Kernel density estimation [150].

In numerical experiments, SDDP with importance sampling is shown to outperform MC and QMC sampling based methods, given that it is difficult to sample from the original probability distribution and that the original problem has moderate or high variance [150].

6.4 Deterministic Sampling

As already discussed in Sect. 4, in step 6 of SDDP (Algorithm 1) also some deterministic sampling can be used. In this case, $|\mathcal{K}| = 1$. In the literature, two different approaches are considered.

Worst Approximation Sampling. The first one requires that in addition to the (lower) cut approximation $\mathfrak{Q}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$ also an upper approximation $\overline{\mathfrak{Q}}_t(\cdot)$ is constructed and iteratively refined in SDDP. Assume that in the forward pass on stage $t-1$ the trial point x_{t-1}^i has been computed. Then, for stage t the approximate subproblem (10) is solved for x_{t-1}^i and for all noise terms $\xi_{tj}, j = 1, \dots, q_t$, yielding optimal states x_{tj}. For the next stage, the trial point $x_t^i = x_{tj'}$ is chosen such that

$$j' \in \underset{j=1,\dots,q_t}{\arg\max} \left\{ \overline{\mathfrak{Q}}_t^i(x_{tj}) - \mathfrak{Q}_t^i(x_{tj}) \right\},$$

i.e., that the gap between the current upper and lower approximations is maximized. This corresponds to sampling noise term $\xi_{tj'}$ on stage t.

This form of deterministic sampling is used for SDDP in [231]. Its computational drawback is that at each stage q_t subproblems have to be solved instead of only $|\mathcal{K}| \ll q_t$. A similar approach was first proposed by Baucke et al. in [10, 11] and called *problem-child node selection.* However, their setting differs a bit from original SDDP, as each subproblem contains specific variables $x_{tj}, j = 1, \dots, q_t$, for all random outcomes, and therefore in their case only one subproblem has to be solved in the sampling step. Another related sampling scheme is used in *robust dual dynamic programming* (RDDP) [76]. In that case, $\xi_{tj'}$ is determined by solving a special upper bounding problem containing $\overline{\mathfrak{Q}}_t^i(\cdot)$

Explorative Sampling. Explorative deterministic sampling is proposed in [117] as part of EDDP. It is based on the concepts of saturated and distinguishable points, which we introduced in Sect. 4.2. As for the previous sampling scheme, the idea is to solve the forward pass subproblems for all $\xi_{tj}, j = 1, \dots, q_t$. Instead of maximizing an approximation gap, however, the trial point $x_t^i = x_{tj'}$ is chosen such that

$$j' \in \arg\max_{j=1,\dots,q_t} \min_{x_t \in X_t^{sat}} \|x_{tj} - x_t\|,$$

i.e., the minimum distance to already saturated points is maximized. In other words, a maximum distinguishable point is chosen.

As shown in [71], worst approximation sampling and explorative sampling are equivalent in the sense that both approaches are guaranteed to lead to effective iterations, see Sect. 4.2.

7 Stopping Criteria

In each iteration i of SDDP, a valid lower bound $\underline{v}^i$ for the optimal value v^* is determined. Additionally, a statistical upper bound $\overline{v}^i_{\mathcal{K}}$ can be computed. Since the latter is not necessarily valid, an important question is when to consider an obtained policy $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ as (approximately) *optimal* and to stop the SDDP method. If the stopping criterion is too conservative, the algorithm may iterate much longer than required, if it is too optimistic, then SDDP may stop prematurely.

Confidence Stopping Criteria. In their seminal work on SDDP, Pereira and Pinto propose to use a confidence interval based stopping criterion [153]. An approximate confidence interval for a true valid upper bound $\overline{v}^i$ is determined as follows using the estimates $v^i(\xi^k)$ from (21).

Under random independent sampling, the values $v^i(\xi^k)$ are i.i.d. random variables with expected value $\overline{v}^i$ and variance $(\sigma^i)^2$. Moreover, knowing the sample mean $\overline{v}^i_{\mathcal{K}}$ (21), we can define a standardized random variable

$$Z^i_{\mathcal{K}} := \frac{\overline{v}^i_{\mathcal{K}} - \overline{v}^i}{\frac{\sigma^i}{\sqrt{K}}}. \tag{25}$$

According to the Central Limit Theorem, this random variable asymptotically, that is, for $|\mathcal{K}| \to \infty$, follows a standard normal distribution $\mathcal{N}(0,1)$. This implies that for sufficiently large $|\mathcal{K}|$, $Z^i_{\mathcal{K}}$ is approximately standard normal distributed.

Due to symmetry of the standard normal distribution, it follows

$$\mathbb{P}(-z_{1-\alpha/2} \leq Z^i_{\mathcal{K}} \leq z_{1-\alpha/2}) \approx 1 - \alpha,$$

where $z_{1-\alpha/2}$ denotes $(1-\frac{\alpha}{2})$-quantiles of $\mathcal{N}(0,1)$ for some level $\alpha \in (0,1)$.

Inserting (25) and rearranging yields an approximate $(1-\alpha)$-confidence interval for the true upper bound $\overline{v}^i$:

$$\left[\overline{v}^i_{\mathcal{K}} - z_{1-\frac{\alpha}{2}} \frac{\sigma^i}{\sqrt{|\mathcal{K}|}}, \overline{v}^i_{\mathcal{K}} + z_{1-\frac{\alpha}{2}} \frac{\sigma^i}{\sqrt{|\mathcal{K}|}}\right].$$

As σ^i is unknown, it can be replaced by the sample standard distribution $\sigma^i_{\overline{v},K}$ which is defined by the sample variance

$$(\sigma^i_{\overline{v},\mathcal{K}})^2 := \frac{1}{|\mathcal{K}|-1} \sum_{k\in\mathcal{K}} (v^i(\xi^k) - \overline{v}^i_{\mathcal{K}})^2.$$

In that case, the standardized variable approximately follows a Student's t-distribution with degree of freedom $|\mathcal{K}|-1$. In the literature on SDDP, even in this case, the $(1-\alpha)$-confidence interval for the true upper bound $\overline{v}^i$ is usually approximated using a standard Normal distribution [201], though, which yields:

$$\left[\overline{v}^i_{\mathcal{K}} - z_{1-\frac{\alpha}{2}} \frac{\sigma^i_{\overline{v},\mathcal{K}}}{\sqrt{|\mathcal{K}|}}, \overline{v}^i_{\mathcal{K}} + z_{1-\frac{\alpha}{2}} \frac{\sigma^i_{\overline{v},\mathcal{K}}}{\sqrt{|\mathcal{K}|}}\right]. \tag{26}$$

Pereira and Pinto propose choosing $\alpha = 0.05$, which implies $z_{1-\alpha/2} = 1.96$, and stopping SDDP if the lower bound $\underline{v}^i$ is included in this confidence interval [153].

As pointed out by Shapiro [198], this stopping criterion has several flaws. For instance, the higher the sample variance $(\sigma^i_{\overline{v},\mathcal{K}})^2$, the earlier $\underline{v}^i$ exceeds the lower end of the confidence interval, which provides a misguided incentive to increase $(\sigma^i_{\overline{v},\mathcal{K}})^2$. The same is true for increasing the confidence $1-\alpha$, which contradicts the intuition behind α. Additionally, faster stopping can be achieved by reducing the sample size $|\mathcal{K}|$. Finally, the above stopping criterion may favor premature stopping, as it is rather unlikely that $\overline{v}^i$ is located exactly at the lower bound of the confidence interval.

For these reasons, Shapiro proposes a more conservative stopping criterion where SDDP terminates if the difference between the upper bound of the confidence interval (26) and $\underline{v}^i$ is sufficiently small.

Sometimes it is also suggested to include values $v^j(\xi^k)$ from previous iterations $j < i$ in (21), for instance if $|\mathcal{K}|$ is too small to obtain a reasonable bound. However, this destroys the independence between the different samples. Thus, the Central Limit Theorem can no longer be applied and the confidence-based stopping criteria are not applicable. [49].

A Hypothesis Test Perspective. Considering that hypothesis tests and confidence intervals are closely related, the above stopping criterion can also be interpreted in terms of a hypothesis test with hypotheses [105]:

$$H_0 : \overline{v}^i = \underline{v}^i, \quad \text{against} \quad H_1 : \overline{v}^i \neq \underline{v}^i.$$

The null hypothesis H_0 is tested using the test statistic $\overline{v}^i_{\mathcal{K}}$, which is assumed to be approximately normal distributed. This can again be reasoned using the Central Limit Theorem for sufficiently large $|\mathcal{K}|$. Then, the region of acceptance for H_0 in iteration i is given by the interval (26): If the lower bound $\underline{v}^i$ does not exceed the lower bound of this region, then optimality is rejected. Otherwise, there is no compelling reason to reject it, so it is retained. By choosing α, the type I error (rejecting optimality although SDDP has converged) can be controlled. However, this comes at the cost of a possibly high type II error (stopping the algorithm prematurely) [105].

Different Hypothesis Tests. To avoid stopping prematurely, Homem-de-Mello et al. [105] propose a modified hypothesis test controlling type I and type II errors simultaneously. The basic principle is very similar to above, even if it is presented for a one-sided hypothesis test with $H_0 : \overline{v}^i \leq \underline{v}^i$. The key difference is that if $\underline{v}^i$ lies inside of the region of acceptance, the hypothesis of optimality is not necessarily retained, but still may be rejected. In particular, stopping SDDP should be prevented if the true upper bound $\overline{v}^i$ exceeds the lower bound $\underline{v}^i$ considerably. As $\overline{v}^i$ is not known, we cannot observe when this event occurs, but we can predefine a bound $\gamma > 0$ on the probability of stopping given that it happens. For fixed γ and α, and given sample estimates, we can then compute a percentage difference δ^i between $\overline{v}^i$ and $\underline{v}^i$, for which the probability of a type II error (premature stopping) is bounded by γ:

$$\delta^i = (z_{1-\alpha} + z_{1-\gamma}) \frac{\sigma^i_{\overline{v},\mathcal{K}}}{\underline{v}^i \sqrt{|\mathcal{K}|}}.$$

If δ^i is below some predefined threshold $\overline{\delta}$, the sample estimates guarantee that for deviations larger than $\overline{\delta}$, the type II error is under control. Therefore, SDDP stops. Otherwise, the control of the type II error is not considered sufficient, and the algorithm proceeds. In other words, SDDP only terminates when $\underline{v}^i$ lies inside of the region of acceptance *and* when the type II error is bounded by γ for a sufficiently small percentage difference δ^i.

Computational experiments with $\overline{\delta} = 0.1$ and $\gamma = 0.05$ indicate that this stopping criterion is effective in preventing SDDP from premature stopping [105]. Still, it is a heuristic, and so far, no proposed statistical testing procedure guarantees that the probability of stopping prematurely is bounded by some $\gamma > 0$ in general.

Predefined Criteria. The previous statistical stopping criteria are computationally demanding and require $|\mathcal{K}|$ to be sufficiently large to yield reasonable approximate confidence intervals. Furthermore, in practical applications (MSLP) is often too large to achieve convergence in reasonable time. Finally, the statistical stopping criteria do not necessarily generalize to extensions of SDDP, such as risk-averse variants, see Sect. 12. Therefore, in practice often more convenient stopping criteria are used for SDDP. For instance, it is common to stop SDDP after a fixed number of iterations $I \in \mathbb{N}$, after a fixed number of cuts $|\mathcal{K}|I$, after a predefined time or if the lower bounds $\underline{v}^i$ have stalled. Neither guarantees that an optimal policy is determined, though.

Deterministic Stopping. Finally, SDDP can be stopped deterministically as long as valid upper bounds $\overline{v}^i$ for v^* are computed in addition to lower bounds $\underline{v}^i$. In that case, for some predefined optimality tolerance $\varepsilon > 0$, SDDP stops with an (approximately) optimal policy if $\overline{v}^i - \underline{v}^i \leq \varepsilon$.

This stopping criterion requires significant additional computational effort to determine true upper bounds $\overline{v}^i$. Hence, there is a trade-off between achieving a more reasonable stopping criterion and spending computational resources on computations offside of the core elements of SDDP. We address how such exact upper bounds can be computed in the next section.

Summarizing, despite various attempts at developing reasonable termination criteria for SDDP, optimally stopping SDDP remains an open challenge.

8 Exact Upper Bounds and Upper Approximations

The idea of computing deterministic upper bounds $\overline{v}$ for v^* and deterministic upper approximations $\overline{\mathfrak{Q}}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$ has drawn a lot of interest in the research community recently, both in analyzing the convergence behavior of SDDP, see Sect. 4, and in developing deterministic stopping criteria, see Sect. 7.

An intuitive way to determine upper approximations $\overline{\mathfrak{Q}}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$ is based on the observation that due to convexity of $\mathcal{Q}_t(\cdot)$ all its secants lie above or on its graph. Therefore, an upper approximation is possible by a convex combination of points $(x_{t-1}, \mathcal{Q}_t(x_{t-1}))$. To obtain an approximation on the whole state space, it can be extended using a regularization with a Lipschitz constant L_t of $\mathcal{Q}_t(\cdot)$. Such constant exists according to

Corollary 2.10. In this light, $\overline{\mathfrak{Q}}_t(\cdot)$ can be constructed as [231]

$$\overline{\mathfrak{Q}}_t(x_{t-1}) = \operatorname{co}\left(\min_{m=1,\dots,M_t} \left\{ \mathcal{Q}_t(x_{t-1}^m) + L_t \|x_{t-1} - x_{t-1}^m\| \right\} \right), \tag{27}$$

where $\operatorname{co}(f)$ denotes the convex envelope of function f. This is illustrated in Figure 9.

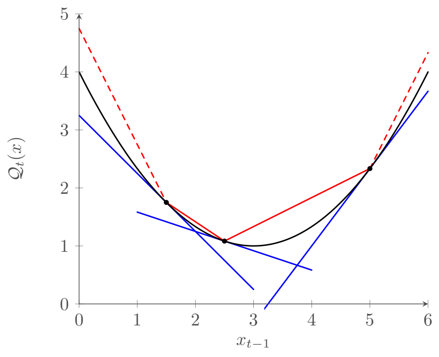


Figure 9: Inner and outer approximation of $\mathcal{Q}_t(\cdot)$.

Alternatively, by interpreting this idea from a set perspective, the convex epigraph $\operatorname{epi}(\mathcal{Q}_t)$ of $\mathcal{Q}_t(\cdot)$ can be approximated by the convex hull $\operatorname{conv}(w_{t-1}^1, \dots, w_{t-1}^{M_t})$ of finitely many points $w_{t-1} := (x_{t-1}, \mathcal{Q}_t(x_{t-1}))$ in $\operatorname{epi}(\mathcal{Q}_t)$.

In principle, there are two different approaches to realize this idea. One uses the above perspectives, which we refer to as *primal*, and one is related to some *dual* perspective on SDDP and its value functions [97, 119].

8.1 Primal Inner Approximation

Similar to subproblems (10), based on upper approximations $\overline{\mathfrak{Q}}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$, approximating subproblems can be defined by replacing $\mathcal{Q}_t(\cdot)$ with $\overline{\mathfrak{Q}}_t(\cdot)$ in the DPE for all $t \in [T]$. This idea is first introduced by Philpott et al. [161]. As they consider only the RHS of (MSLP) to be uncertain, we adopt this assumption, although it is not required.

For stages $t = T-1, \dots, 2$, each element m in a given set of points $x_t^1, \dots, x_t^{M_{t-1}}$ and each $\xi_{tj}, j = 1, \dots, q_t$, the following subproblem can be solved by backward recursion:

$$\overline{Q}_t(x_{t-1}^m, \xi_{tj}) := \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \overline{\mathfrak{Q}}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^m, \xi_{tj}). \end{cases} \tag{28}$$

Here, as indicated in (27), the upper approximation $\overline{\mathfrak{Q}}_{t+1}(\cdot)$ is defined as a convex combination of points $\big(x_t^m, \overline{\mathcal{Q}}_{t+1}(x_t^m)\big), m = 1, \dots, M_t$. The key difference is that instead of $\mathcal{Q}_{t+1}(x_t^m)$ here $\overline{\mathcal{Q}}_{t+1}(x_t^m) := \mathbb{E}\left[\, \overline{Q}_{t+1}(x_t^m, \boldsymbol{\xi}_{t+1})\right]$ is used, as $\mathcal{Q}_{t+1}(\cdot)$ is not known:

$$\overline{\mathfrak{Q}}_{t+1}(x_t) := \begin{cases} \min\limits_{w} & \sum\limits_{m=1}^{M_t} w_m \overline{\mathcal{Q}}_{t+1}(x_t^m) \\ \text{s.t.} & \sum\limits_{m=1}^{M_t} w_m x_t^m = x_t \\ & \sum\limits_{m=1}^{M_t} w_m = 1 \\ & w_m \geq 0, \quad m = 1, \dots, M_t. \end{cases} \tag{29}$$

Furthermore, compared to (27) no regularization is used.

By recursion, it can be shown that

$$\overline{Q}_t(x_{t-1}^m, \xi_{tj}) \geq Q_t(x_{t-1}^m, \xi_{tj})$$

for all $m = 1, \dots, M_{t-1}$ and $j = 1, \dots, q_t$. This implies

$$\overline{\mathcal{Q}}_t(x_{t-1}^m) \geq \mathcal{Q}_t(x_{t-1}^m).$$

The first-stage problem then yields

$$\overline{v}^{IA} := \begin{cases} \min\limits_{x_t} & c_1^\top x_1 + \overline{\mathfrak{Q}}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1, \end{cases}$$

with $\overline{v}^{IA}$ an exact valid upper bound to v^*.

The main challenge with this approach is to appropriately choose the set of points $x_{t-1}^m, m = 1, \dots, M_{t-1}$. On the one hand, they should be chosen such that as much of $\mathcal{X}_{t-1}$ is spanned as possible. On the other hand, choosing (at least some of) those points as extreme points leads to $M_t \geq 2^{n_t}$ points, *i.e.*, the number of required points grows exponentially in the dimension of the state space.

An alternative is to use the trial points from the SDDP forward pass [161]. Even using these points, the computational effort may become excessive, though. Similarly to the SDDP backward pass, subproblems (28) have to be solved for each stage $t \in [T]$, each point $x_{t-1}^m, m = 1, \dots, M_{t-1}$, and each noise term $\xi_{tj}, j = 1, \dots, q_t$. However, the number M_{t-1} of points to be considered grows with each iteration, as it contains all previous trial solutions. It is therefore suggested to only use the upper bound computation every few hundred iterations, and not to permanently incorporate it into the backward pass [161]. This hinders using the upper bounds $\overline{v}^{IA}$ in the stopping criterion of SDDP in each iteration, though.

Moreover, the obtained bounds $\overline{v}^{IA}$ may be very loose, especially in problems (MSLP) with a high number of stages. Computational tests are required to assess whether the information gain justifies the additional computational effort and, possibly, higher number of iterations.

Baucke et al. provide a different perspective on the previous inner approximation idea [10]. Instead of (29), they use its dual representation

$$\overline{\mathfrak{Q}}_{t+1}(x_t) = \begin{cases} \max\limits_{\mu,\lambda} & x_t^\top \lambda + \mu \\ \text{s.t.} & (x_t^m)^\top \lambda + \mu \leq \overline{\mathcal{Q}}_{t+1}(x_t^m), \quad m = 1, ..., M_t. \end{cases} \tag{30}$$

This shows that $\overline{\mathfrak{Q}}_{t+1}(\cdot)$ can be equivalently described by maximizing over the coefficients of all supporting hyperplanes for points $\big(x_t^m, \overline{\mathcal{Q}}_{t+1}(x_t^m)\big), m = 1, \dots, M_t$.

In [10], the dual problem is additionally regularized, *i.e.*, enhanced by constraint $\|\lambda\| \leq L_t$, with L_t denoting a Lipschitz constant of $\overline{Q}_t(\cdot, \cdot)$. This is equivalent to regularizing the primal problem (29) with the dual norm to $\|\cdot\|$, cf. (27). This way, a reasonable approximation is also achieved for points outside of the convex hull of the set defined by the points $x_t^m, m = 1, \dots, M_t$.

Using this expression for the inner approximation functions, Baucke et al. propose a deterministic algorithm for multistage stochastic convex programs. In their case, subproblems (28) are solved in each backward pass iteration, and $\overline{\mathfrak{Q}}_{t+1}^i(\cdot)$ is updated by adding constraint $(x_t^{\widetilde{m}})^\top \lambda + \mu \leq \overline{\mathcal{Q}}_{t+1}^i(x_t^{\widetilde{m}})$ for the current iterate $x_t^{\widetilde{m}}$. The proposed algorithm differs in further regards from standard SDDP, for instance it requires a multi-cut approach, see Sect. 21. Moreover, choosing a reasonable and valid value for L_t can be very challenging, but is crucial for the proposed method to work as intended.

8.2 Dual SDDP

To compute deterministic upper bounds $\overline{v}$ for v^* recently a dual perspective on SDDP and the DPE (4) has gained attention.

Using Convex Conjugates of Value Functions. The first proposal in this context, by Leclère et al. [119], exploits convex conjugates and the related duality concepts to derive *dual value functions* for (MSLP) where uncertainty only appears in the RHS $h_t(\xi_t)$.

Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{-\infty, \infty\}$. Then its *convex conjugate* $f^\star(\cdot)$ is defined as [186]

$$f^\star(\lambda) := \sup_{x \in \mathbb{R}^n} \lambda^\top x - f(x).$$

For (MSLP), the convex conjugates $D_t(\cdot) := Q_t^\star(\cdot)$ of the value functions $Q_t(\cdot)$ can be considered as dual value functions for $t = 2, \dots, T$. It can be shown that these functions also satisfy some DPE with linear subproblems on each stage. Whereas Leclère et al. consider a more general setting including control variables u_t (see Remark 2.2), for (MSLP) as defined in Sect. 2 (and especially under Assumption 5), for $t = 2, \dots, T$, these subproblems can be expressed by

$$D_t(\lambda_{t-1}) := \begin{cases} \min\limits_{\lambda_t, \mu_t, \gamma_t} & \sum\limits_{j=1}^{q_t} p_{tj} \Big(-h_{tj}^\top \mu_{tj} + D_{t+1}(\lambda_{tj}) \Big) \\ \text{s.t.} & T_{t-1}^\top \left(\sum\limits_{j=1}^{q_t} p_{tj} \mu_{tj} \right) - \sum\limits_{j=1}^{q_t} p_{tj} \gamma_{tj} + \lambda_{t-1} = 0 \\ & W_t^\top \mu_{tj} = \lambda_{tj} + c_t, \qquad j = 1, \dots, q_t \\ & \gamma_{tj} \leq 0, \qquad j = 1, \dots, q_t. \end{cases} \tag{31}$$

For the first stage, we obtain a deterministic problem, which by $T_0 \equiv 0$ simplifies to

$$D_1(\lambda_0) = \min_{\mu_1} \; h_1^\top \mu_1 + D_t\big(W_1^\top \mu_1 - c_1\big)$$

for some arbitrary initial $\lambda_0 \leq 0$ (note that more general formulations of (MSLP) may lead to a dependence on λ_0).

Using this dynamic recursion, it is possible to apply an SDDP-type algorithm, called *dual SDDP*, to $D_t(\cdot)$, using iteratively improving outer approximations $\mathfrak{D}_t^i(\cdot)$ for $D_t(\cdot)$. Analogously to SDDP, this iterative method yields a converging deterministic lower bound for the first-stage optimal value, *i.e.*, $\mathfrak{D}_1^i(\lambda_0) \leq D_1(\lambda_0)$. Applying conjugacy theory again, we obtain

$$\overline{v}^i = \big(\mathfrak{D}_1^i\big)^\star(x_0) \geq D_1^\star(x_0) = Q_1^{\star\star}(x_0) = Q_1(x_0) = v^*.$$

Hence, deterministic upper bounds for v^* can be obtained as conjugates of the first-stage approximations $\mathfrak{D}_t^i(\cdot)$ evaluated at $x_0 = 0$, and $(\overline{v}^i)_i$ defines a sequence converging to v^* [119].

Using the Dual of (MSLP). Guigues et al. propose an alternative way to define dual value functions and DPE that can be exploited in a dual SDDP algorithm [97]. Instead of working with conjugates of the primal value functions $Q_t(\cdot)$, they first derive the dual to (MSLP) formulated as a single problem (3), and then show that this dual problem can be decomposed using DPE and dual value functions

$$\widetilde{D}_t(\pi_{t-1}) := \begin{cases} \max\limits_{\pi_t} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(- h_{tj}^\top \pi_{tj} + \widetilde{D}_{t+1}(\pi_{tj})\Big) \\ \text{s.t.} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(T_{t-1,j}^\top \pi_{tj}\Big) + W_{t-1}^\top \pi_{t-1} \leq c_{t-1}. \end{cases} \tag{32}$$

It can be argued that these dual DPE are simpler and more intuitive, as they do not require conjugacy theory. Moreover, we immediately obtain that the first-stage optimal value $\widetilde{D}_1(\pi_0)$ equals v^* by strong duality for linear programs. Therefore, using outer approximations $\widetilde{\mathfrak{D}}_t^i(\cdot)$ of these value functions in dual SDDP, again a sequence $(\overline{v}^i)_i$ of deterministic and valid upper bounds can be computed which converges to v^* [97]. On the other hand, the dual value functions $\widetilde{D}_t(\cdot)$ cannot be directly related to the original value functions $Q_t(\cdot)$.

Remark 8.1. *Even if the dual DPE* (31) *and* (32) *are derived using different tools and perspectives, they are still closely related. Note that subproblem* (31) *can be reformulated as*

$$D_t(\lambda_{t-1}) = \begin{cases} \min\limits_{\lambda_t, \mu_t} & \sum\limits_{j=1}^{q_t} p_{tj}\Big(- h_{tj}^\top \mu_{tj} + D_{t+1}(\lambda_{tj})\Big) \\ s.t. & T_{t-1}^\top \left(\sum\limits_{j=1}^{q_t} p_{tj}\mu_{tj} \right) + \lambda_{t-1} \leq 0 \\ & W_t^\top \mu_{tj} = \lambda_{tj} + c_t, \qquad j = 1, \dots, q_t. \end{cases}$$

Using the last constraint, the state λ_{t-1} can be expressed through the dual variables μ_{t-1} from the previous stage: $\lambda_{t-1} = W_{t-1}^\top \mu_{t-1} - c_{t-1}$. Exploiting this, the subproblems only contain dual variables μ_t, which have to be considered as state variables. By *adapting the optimization sense in the objective, we get exactly the structure of* (32).

We can make the following additional observations with respect to the dual DPE (31) and (32). First, in both cases, the subproblems are not necessarily bounded. Therefore, in both cases, artificial bounds are introduced. In [97] they are chosen as $\pi_t \in [\underline{\pi}_t, \overline{\pi}_t]$, whereas in [119] Lipschitz continuity of $\mathcal{Q}_t(\cdot)$ is exploited to impose the bounds $\|\lambda_t\|_\infty \leq L_t$ for Lipschitz constants $L_t, t = 2, \ldots, T$. It is assumed that these bounds are chosen sufficiently large to not affect the optimal solutions.

Second, even if the primal DPE (4) are assumed to have relatively complete recourse (see Assumption 9 and Lemma 2.5), this does not necessarily translate to the dual subproblems. To ensure feasibility, Guigues et al. propose to either use feasibility cuts (also see Sect. 17) or a penalization approach [97].

Third, in contrast to the primal perspective, the subproblems do not decompose by realizations of $\boldsymbol{\xi}_t$, but contain separate dual variables π_{tj} (or $\lambda_{tj}, \mu_{tj}, \gamma_{tj}$, respectively) for all $j = 1, \ldots, q_t$. In the forward pass of dual SDDP the trial point π_t^i (or λ_t^i) that is used as a parameter in the following stage is sampled from these variables.

Finally, if W_t and c_t become uncertain as well, then the value functions and subproblems additional depend on ξ_t. In fact, in formulation (32) the state space has to be extended to include the history ξ_{t-1} of the stochastic process, as the problem contains W_{t-1} and c_{t-1} [97].

Again, an SDDP-type algorithm, also referred to as *dual SDDP* in [97], can be applied to the DPE (32). This algorithm is presented in Algorithm 2. The two variants of dual SDDP have been extended to the risk-averse case [40] (see also Sect. 12) and to problems with infinite horizon (see also Sect. 19) [200].

Dual Inner Approximation. First and foremost, dual SDDP is an alternative to (primal) SDDP to approximate v^* by converging deterministic upper bounds $\overline{v}^i$. However, as shown in [119], if the dual DPE (31) are used, then the obtained approximations $\mathfrak{D}_t^i(\cdot)$ may be translated to inner approximations $\overline{\mathfrak{Q}}_t^i(\cdot)$ of the *primal* value functions $Q_t(\cdot)$. This way, policies $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ for (MSLP) can be computed. The inner approximations can be computed as Lipschitz regularizations (see Sect. 17) of the convex conjugate of the outer approximations $\mathfrak{D}_t^i(\cdot)$, which is shown to be equivalent to solving problem (30) with regularization $\|\lambda\|_\infty \leq L_t$. The key difference to the approach in [10] is the way the primal supporting points x_t^m are determined, that is, by the slopes of the dual outer approximation [119].

Incorporation into SDDP. While dual SDDP can be applied on its own to approximate v^*, and even compute policies $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$, it seems reasonable to incorporate it into (primal) SDDP in order to compute deterministic upper *and* lower bounds for v^*. Guigues et al. suggest to use both variants of SDDP in parallel [97]. In contrast, Leclère et al. propose a framework where primal and dual SDDP are intertwined [119]:

1. Run a forward pass of (primal) SDDP, yielding trial solutions $(x_t^i)_{t\in[T]}$ for the sampled scenario path (the authors choose $|\mathcal{K}| = 1$).

2. Run a backward pass of (primal) SDDP using the trial solutions x_{t-1}^i, obtaining new slopes π_t^i from the cuts.

3. Run a backward pass of dual SDDP using the slopes $\lambda_t^i = \pi_t^i$, obtaining new cuts for the dual problem.

4. Run a forward pass of dual SDDP, to obtain a new dual trajectory $(\tilde{\lambda}_t^i)_{t\in[T]}$ and update the cuts along this trajectory.

Algorithm 2 Dual SDDP from [97]

```
Input: Dual to problem (MSLP) satisfying Assumptions 1 to 9. Appropriate multiplier
    bounds. Stopping criterion.
    Initialization
 1: Initialize cut approximations with bounded $\widetilde{\mathfrak{D}}_t^0(\cdot)$ for all $t = 2, \dots, T$.
 2: Initialize upper bound with $\overline{v}^0 = +\infty$.
 3: Set iteration counter to $i \leftarrow 0$.
    Dual SDDP Loop
 4: while Stopping criterion not satisfied do
 5:     Set $i \leftarrow i + 1$.
        Forward Pass
 6:     Solve the first-stage problem (defined by replacing $\widetilde{D}_2(\cdot)$ with $\widetilde{\mathfrak{D}}_2^i(\cdot)$ and adding
        multiplier bounds in (32)). Store the trial point $\pi_1^i$.
 7:     for stages $t = 2, \dots, T$ do
 8:         Solve the stage-$t$ subproblem (defined by replacing $\widetilde{D}_{t+1}(\cdot)$ with $\widetilde{\mathfrak{D}}_{t+1}^i(\cdot)$ and
            adding multiplier bounds in (32)) for $\pi_{t-1}^i$ to obtain $\pi_{tj}^i, j = 1, \dots, q_t$.
 9:         Sample $\widetilde{j}$ from $j = 1, \dots, q_t$ and set $\pi_t^i = \pi_{t\widetilde{j}}$.
10:     end for
        Backward Pass
11:     for stages $t = T, \dots, 2$ do
12:         Solve the updated stage-$t$ subproblem (10) (defined by replacing $\widetilde{D}_{t+1}(\cdot)$ with
            $\widetilde{\mathfrak{D}}_{t+1}^{i+1}(\cdot)$ and adding multiplier bounds in (32)) for $\pi_{t-1}^i$. Store the optimal
            value $\overline{D}_t(\pi_{t-1}^i)$ and the optimal dual vector $x_{t-1}^i$.
13:         Compute
                $$\alpha_t^{D,i} := \overline{D}_t(\pi_{t-1}^i) - \left(\beta_t^{D,i}\right)^\top \pi_{t-1}^i$$
            and
                $$\beta_t^{D,i} := -W_{t-1} x_{t-1}^i.$$
14:         Update the cut approximation of $\widetilde{D}_t(\cdot)$ to
                $$\widetilde{\mathfrak{D}}_t^{i+1}(x_{t-1}) := \min\left\{\widetilde{\mathfrak{D}}_t^i(x_{t-1}),\ \alpha_t^{D,i} + \left(\beta_t^{D,i}\right)^\top \pi_{t-1}\right\}.$$
15:     end for
16:     Solve the first-stage problem (defined by replacing $\widetilde{D}_2(\cdot)$ with $\widetilde{\mathfrak{D}}_2^{i+1}(\cdot)$ and adding
        multiplier bounds in (20)) to obtain an upper bound $\overline{v}^i$.
17: end while
Output: Upper bound $\overline{v}^i$ for $v^*$.
```

One computational drawback of this framework, and of dual SDDP in general, is that each iteration of dual SDDP is much more computational expensive than for standard (primal) SDDP. This hampers the application of a solely deterministic stopping criterion for very large problems [97, 119].

9 Applications

In this section, we present different application areas of SDDP. We also point out applications in which some of the Assumptions 1 to 9 are not satisfied, and therefore either modifications of (MSLP) or algorithmic extensions are required in order to apply SDDP. These use cases can be regarded as a motivation for the enhancements of SDDP that we cover in the following sections.

9.1 Power System Optimization

By far the dominating application field of SDDP is power system optimization, in particular, the operational planning of energy systems including hydro storages by a central planner. This is due to its adequacy for such problems, but also due to its origins in optimizing the operational planning of the Brazilian hydrothermal system [152, 153].

In general, solving power system optimization problems is a very complex task, as it allows for incorporation of various technical and economical details and uncertainties [110, 149, 182, 183, 184, 209, 210, 232]. Including all these details in one single problem is computationally intractable. Therefore, usually a hierarchy of problems is considered, dealing with different time-scales and perspectives [47], such as short-term dispatch (a few days or weeks), mid-term operational planning (1-2 years) and long-term operational planning (3-5 years) [72, 81]. Results from a long-term model can then be incorporated into one with a shorter horizon, but more detail in other modeling aspects.

9.1.1 Long-term Operational Planning

SDDP is most prominently used for long-term operational planning (LTOP) of hydrothermal power systems, also called long-term hydrothermal scheduling (LTHS). In the research literature, SDDP has been applied to LTOP of various hydrothermal systems, with the most prominent ones being the hydro power dominated systems in Brazil [15, 30, 31, 32, 42, 47, 48, 49, 52, 84, 97, 105, 126, 127, 129, 133, 136, 161, 166, 203, 204, 205, 208, 216, 226], other Central or South American countries [6, 70, 179, 213], Norway [80, 187] and New Zealand [160, 162, 229]. Additionally, to this day, SDDP is applied by the Brazilian system operator ONS in practice [134, 135].

The aim in LTOP is to determine an optimal policy for the amount of power to be generated by thermal and hydroelectrical utilities over some planning horizon of several years (usually with monthly resolution) such that demand is satisfied, technical constraints are fulfilled and the expected cost is minimized [160]. The main focus is on managing hydro reservoirs, and thus the water resource efficiently. This is not trivial. While there is an incentive to use all the water in a reservoir immediately, as no fuel costs occur, also the potential value of storing water for later stages has to be considered, with taking into account the uncertainty of future inflows. For this reason, it can be beneficial to retain water in wet periods for following dryer periods. The ability to store water in reservoirs leads to a temporal coupling of the stages. The number of inflow realizations q_t per stage is typically chosen in a range between 20 and 100. For $T = 60$, this yields a scenario tree with about $1.15 \cdot 10^{78}$ or 10^{120} scenarios. Per forward pass, either a single scenario [48] or 100 to 200 scenarios are sampled.

LTOP can be used to illustrate some of the challenges and limits of standard SDDP, and thus motivate the necessity of extensions.

Autoregressive Uncertainty. In LTOP, the main source of uncertainty are future (usually monthly) inflows into the reservoirs. These inflows often show seasonality and a temporal or spatial coupling which has to be considered in modeling. Therefore, usually *autoregressive* (AR) processes are used to model and forecast them, in particular *periodic autoregressive* (PAR) [133, 134] and related models [139]. This means that for each reservoir and each month a different AR model is fitted, or in other words, that the parameters in the AR model are allowed to differ between months.

Additionally, often hydro reservoirs are organized in cascade systems. Then, the generation of one turbine may affect the inflow of downstream reservoirs, such that they cannot be managed separately. For this reason, inflows often do not only show temporal correlation and seasonality, but also spatial correlation. To address this, instead of PAR, *spatial periodic autoregressive* (SPAR) models can be used [127]. These models are still linear, but instead of only autoregressive components, *i.e.*, lags of ξ_{it} for some reservoir i, also lags of the inflows of neighboring reservoirs i' are used to explain ξ_{it}. Apart from inflow lags, also different exogenous variables, such as climate indices, precipitation or sea temperature can be used to explain inflows [124, 165].

Whenever an AR process is used for the uncertain data, the assumption of stagewise independence (Assumption 2) is not satisfied. This motivates an extension of SDDP able to handle stagewise dependent uncertainty. We discuss this in Sect. 14.

Nonlinear Uncertainty. When modeling hydro inflows, the error terms in the AR process are usually assumed to be i.i.d. with normal or log-normal distribution [47, 127]. In the latter case, the model is also referred to as a geometric PAR (GPAR) model [129]:

$$\ln(\xi_t) = \gamma_t + \Phi_t \ln(\xi_{t-1}) + \eta_t. \tag{33}$$

GPAR models are usually more accurate in modeling inflows, as these often tend to positive skewness and are thus not normally distributed. Moreover, they have the advantage that the requirement of non-negative inflows is naturally satisfied.

On the other hand, solving (33) for ξ_t yields an AR process with multiplicative instead of additive error terms [204], which is a nonlinear model. Incorporating this into the DPE destroys the convexity of $\mathcal{Q}_t(\cdot)$, making a direct application of SDDP impossible. Instead, the nonlinear model has to be approximated linearly [204]. Another idea is to normalize the inflows first using a Box-cox transformation. As such a transformation is nonlinear, still a linear approximation is required afterwards, though [168]. Further strategies to avoid non-negative inflows and nonlinearities are discussed in [47, 177]. In [45] it is suggested to apply bootstrapping to resample directly from the historical residuals instead of applying a nonlinear transformation.

Continuous Uncertainty and Distributional Uncertainty. As stated before, usually a normal or log-normal distribution is assumed for the error terms in the inflow models, both being continuous distributions (an exception is [171] where inflows are modeled as a continuous process with discrete random errors). For this reason, the assumption of finite discrete random variables (Assumption 5) is not satisfied. Additionally, the chosen distribution for the model may not coincide with the *true distribution* of the uncertain data. This raises the questions of how to handle continuous uncertainty and distributional uncertainty in SDDP. We address these questions in Sect. 11 and Sect. 13.

Computational Performance. Despite the amenities of SDDP, its performance may suffer for problems with a large number of state variables, due to its exponential complexity in the state dimension n_t, see Sect. 4.2. For instance, SDDP is computationally prohibitive for a complete model of the Brazilian energy system consisting of about 150 thermal plants and more than 150 hydro storages [48]. This is aggravated if the state dimension is artificially increased, *e.g.*, in order to deal with stagewise dependent uncertainty, see Sect. 14. As a relief, it is common practice to aggregate reservoirs based on their region and hydrological properties in so-called *energy equivalent reservoirs* (EER) [4], thus reducing the state dimension [135]. However, this comes with an increased abstraction, and may lead to suboptimal policies. Moreover, as outlined in [47], the EER modeling may introduce some nonlinearities into the system, which have to be mitigated by linearization.

The computational complexity with respect to the state space also makes general performance improvements for SDDP indispensable, which we discuss in Sect. 21.

End-of-horizon Effect. Another challenge when applying SDDP to LTOP in practice is the so-called *end-of-horizon effect.* It relates to the effect that obtained policies do not guarantee a continuous and reliable energy supply *after* the planning period, because in an optimal policy, all energy remaining in the reservoirs will be used at the end of the planning period. A typical planning horizon for LTOP are 5 years with a monthly resolution, leading to 60 stages. A common practice to mitigate the end-of-horizon effect is to add 60 more stages to the problem, *i.e.*, to consider a problem with 120 stages [204], even if only decisions of the first half are about to be implemented. Alternatively, it seems natural to analyze how SDDP can be applied for problems with an infinite horizon or with a random horizon, where Assumption 1 is not satisfied. We address this in Sect. 19 and 20.

Risk-aversion. Due to the high importance of system reliability and stability to prevent outages and electricity shortages, system planners may favor more risk-averse policies compared to the risk-neutral ones obtained by standard SDDP. Therefore, there has been an increased interest recently to take risk aversion into account when applying SDDP to LTOP [108, 208]. However, as Assumption 8 is no longer satisfied, this requires to extend standard SDDP to a risk-averse variant. We discuss different approaches to achieve this in Sect. 12.

9.1.2 Medium-term Operational Planning

Structurally, medium-term operational planning problems (MTOP) do not differ much from LTOP. The main difference is that a shorter, one- or two-year time horizon is considered [47, 160, 161, 179].

Price Uncertainty in the Objective. Especially on a medium-term time horizon, SDDP has also been adopted from the traditional setting with a single system operator to more market-driven systems, in which several electricity suppliers are active. In such systems, besides inflows also spot prices can be considered uncertain. This imposes an additional challenge to SDDP, as it leads to stagewise dependent uncertainty in the objective. We discuss this in detail in Sect. 14. To deal with this challenge, for instance, for the operational planning of the Norwegian hydro-storage system, usually a combined SDP/SDDP approach is used [79, 80, 81, 100, 101].

Water Head Effect. In LTOP the so-called *water head effect* of hydro storages is often disregarded, but it may become decision-relevant in (MTOP). This effect describes that the production of a hydro plant increases with the net head of the reservoir. As this production function is multiplied with the water discharge, it introduces non-convexities to the problem. Therefore, if this nonlinear effect is explicitly considered, suitable extensions of SDDP to non-convex problem are required [36, 103, 163]. We cover such extensions in Sect. 16.

9.1.3 More Energy Applications

We briefly summarize further applications of SDDP in power system optimization.

Short-term Dispatch. SDDP is particularly suited for long-term planning, but it can also be applied to short-term economic dispatch problems [37, 51, 118, 148]. For shorter time horizons, it may be reasonable to include additional system aspects, for instance power flow and security constraints, reserve energy or different ancillary services [132, 216]. If security constraints are considered, usually linear DC power flow models are used, but recently also AC power flow has gained interest [112].

Another research stream considers CO_2 emissions, which can be covered by imposing an emission quota system [14, 180, 178] or by introducing emission trading [181]. The first approach leads to an (MSLP) which has no block-diagonal structure (Assumption 7). We discuss how SDDP can be applied in this case in Sect. 18.

Using a reasonable extension to mixed-integer programs, see Sect. 16, also unit commitment problems are accessible by the SDDP idea [234].

Different Storage Systems. As different types of storage systems can be modeled similar to hydro storages, SDDP is also applicable to such systems, for instance, to optimize gas storage facilities [227] or energy storages in microgrids [22].

Optimal Bidding. Instead of minimizing expected system cost from the perspective of a central system operator, in strategic bidding problems power plant operators attempt to determine an optimal bidding policy in order to maximize their expected revenue, while taking into account information uncertainty, for example with respect to inflows or the market-clearing price; see [212, 214] for an overview.

Since the future revenue functions of the price-maker have a sawtooth shape, the resulting problem is non-convex [213]. Therefore, to apply the SDDP idea, tailor-made extensions are required, *e.g.*, convexifications, approximations by saddle cuts [55] or by step functions [163, 229]. For methodological details, we refer to Sect. 16.

Recently, also applying SDDP to optimize trading in continuous intraday markets has gained attention [206].

Investment Planning. An important long-term optimization problem in power systems is to make optimal (risk-averse) investment decisions, either with respect to the expansion of renewables [33, 123, 215] or to conventional projects.

For conventional power systems, common investment problems address the questions of generation expansion or transmission expansion. The main challenge with such problems is that they naturally impose the introduction of integer decision variables. Therefore, in such a case relaxations [147] or appropriate extensions of SDDP, *e.g.*, SDDiP [234], have to be used (see Sect. 16). Alternatively, SDDP can be incorporated into a larger Benders decomposition framework, where at the first stage binary investment decisions are taken and at the second stage a multistage stochastic linear program is solved by SDDP [178]. Similar applications are considered in [52] and [41] with a special focus on risk and reliability constraints.

Coping with Renewable Uncertainty. An increasing share of renewable energy sources introduces more variability to an energy system, which has to be taken into account and balanced by appropriate mechanisms. The usage of distributed grid-level storage, such as batteries or electric vehicles, for smoothing out the variable generation of renewables is examined using SDDP in [66, 235].

9.2 Water Resource Management

In many energy applications of SDDP, managing water resources plays a key role, as it couples subsequent stages. Apart from energy optimization, SDDP is also applied to more general water resource management problems, where not only energy production, but also water usage for irrigation in agriculture [155, 221], flow requirements for navigation [221], groundwater [137] or ecological constraints [220] are taken into account in the operational planning of reservoirs. Also related is the problem of river basin management [188].

Additionally, SDDP is used for assessing various quantities in hydrological systems, *e.g.*, the value of water [224], risk for dam projects [2, 223], resource vulnerabilities [189] or benefits and costs of cooperation or non-cooperation [138, 222].

9.3 Portfolio Management

The optimal management of a portfolio of investments, also referred to as *asset allocation*, can be modeled as an (MSLP) [43]. The aim is to distribute a fixed investment sum among a finite number of assets with uncertain returns, in such a way that the expected return at the end of the considered horizon is maximized. By selling or buying certain amounts of assets, the investor can restructure his portfolio in each time period. Usually, both operations are associated with transactions costs, which leads to a very complex problem [225].

In the literature on SDDP, asset allocation problems are quite popular to test proposed improvements and enhancements of SDDP, such as regularization [90], cut-sharing [84] or inexact cuts [8]. Since most investors are risk-averse, asset allocation problems are a popular application [60, 63, 64, 106, 113, 114], but also one of the main drivers for the development of risk-averse SDDP, which we introduce in Sect. 12.

For applications of practical interest, asset allocation becomes very challenging, as pointed out in [225]. First, risk aversion parameters such as λ_t or α_t, see Sect. 12, are not intuitive to choose in such a way that the true preferences of an investor are appropriately represented. For this reason, the authors propose to solve a risk-constrained model with one-period conditional AVaR constraints instead of a usual risk-averse SDDP approach. Second, assuming stagewise independence of asset returns may prove unrealistic, requiring a more sophisticated approach such as incorporating a Markov chain, see Sect. 14. Moreover, the large supply of potential assets leads to a high-dimensional state space.

9.4 Further Applications

Although the focus is on the previous applications, occasionally also other types of applications are investigated using SDDP. Among those applications are dairy farming [61, 83], newsvendor problems [5, 150], inventory management [8, 59, 87, 97], lot-sizing

[218] and routing problems [60]. In [50] and [234] airline revenue management is explored, which is an established problem in dynamic programming, but requires integer variables.

10 Software

Until recently, SDDP implementations have been solely restricted to closed research projects or commercial products. For commercial products, most established is the SDDP implementation by PSR, a Brazilian energy consultancy [172]. A newer stochastic programming software, which also includes SDDP ideas, is provided by Quantego and can be accessed using MATLAB, Python and Java [173]. For research projects, various different implementations exist, covering programming languages like AMPL, C++, Fortran, GAMS, Java or MATLAB, see [57].

In the last few years, open-source implementations have gained more and more interest, with the aim to increase research transparency, enhance research exchange and benchmarking, and facilitate access to SDDP in industry and science [57]. The most prominent programming language in this regard is Julia [21], which provides its own algebraic modeling language JUMP [62] and is increasingly used in operations research and especially stochastic progamming. By now, with `StochDynamicProgram.jl` [120], `StructDualDynProg.jl` [121] and `SDDP.jl` [57] there exist three SDDP implementations in Julia. Similarly, SDDP packages are available in MATLAB (`FAST` [34]), C++ (`StOpt` [77]) and Python (`msppy` [50]).

Currently, `SDDP.jl`, which is based on the concept of policy graphs [56], can be considered the most comprehensive package. It provides many of the features described in this paper, such as cut selection, parallelism, Markov chain SDDP, objective states, belief states, SDDiP, as well as different stopping criteria and sampling approaches. Moreover, it includes some of the approaches discussed for distributionally robust and risk-averse SDDP. However, as most other packages, it requires the underlying stochastic process to be finite. Thus, if Assumption 5 is not satisfied, some discretization has to be applied a priori. Then, the results obtained by SDDP are valid for the discretized problem, but not put into perspective with respect to the *true* problem. `msppy`, on the other hand, integrates both, the discretization by SAA and the solution by SDDP in one package, and thus can also be applied to problems with continuous uncertainty [50].

A more detailed comparison of currently available libraries is presented in [57].

11 SDDP for Continuous Uncertainty [relaxing Assumption 5]

So far, we assumed the uncertainty in (MSLP) to be modeled by some discrete and finite random process, see Assumption 5, in order for SDDP to be applicable. Until the recent work by Forcier and Leclère [71], also all convergence proofs for SDDP leveraged Assumption 5. However, in many practical applications, this assumption is not justified. For example, if the stochastic process governing the uncertain data is modeled by a time series model, the random error terms are usually assumed to follow a continuous distribution [198], see Sect. 9. In the remainder of this section, we denote a problem with such a continuous data process by $(\widetilde{P})$.

As pointed out in Sect. 2.3, for problems with sizes of practical interest, problem $(\widetilde{P})$ is computationally intractable. Therefore, if the true distribution $F_{\boldsymbol{\xi}}$ of the stochastic process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is continuous, usually an approximation with finitely many scenarios is used. In the literature on multistage stochastic programming, a variety of techniques are proposed to generate (and reduce) scenario tree approximations of continuous stochastic processes. For an overview we refer to [128].

11.1 Sample Average Approximation (SAA)

The most common approximation approach is to use random sampling. That means that the distribution $F_{\boldsymbol{\xi}}$ is *approximated* using an empirical distribution $\widetilde{F}_N$ with a finite number N of scenarios, which is obtained by sampling from $F_{\boldsymbol{\xi}}$ [198]. This yields an approximating problem $(\widetilde{P}_N)$, which then can be handled by SDDP. Often, this technique is referred to as *sample average approximation* (SAA), especially, if classical Monte Carlo sampling is used. We discuss SAA and the application of SDDP to solve an SAA problem in more detail now. For a general analysis of SAA, we refer the interested reader to [201].

SAA and SDDP. Under stagewise independence of $(\boldsymbol{\xi}_t)_{t\in[T]}$ (Assumption 2), it is desirable to preserve this property in the SAA problem, especially if the latter should be solved by SDDP. To achieve this, random sampling can be applied to each stage $t = 2, \dots, T$ independently with sample size $\widetilde{q}_t$ [198]. The obtained SAA has a total number of $N = \prod_{t=2}^{T} \widetilde{q}_t$ scenarios, *i.e.*, the number of scenarios is exponentially growing in the number of stages [198].

For the SAA problem $(\widetilde{P}_N)$, for each stage $t = 2, \dots, T$ and each sample $j = 1, ..., \widetilde{q}_t$, the DPE can be written as

$$\widetilde{Q}_t\big(x_{t-1}, \widetilde{\xi}_{tj}\big) := \begin{cases} \min\limits_{x_t} & \big(c_t(\widetilde{\xi}_{tj})\big)^\top x_t + \widetilde{\mathcal{Q}}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \widetilde{\xi}_{tj}) \end{cases} \tag{34}$$

where

$$\widetilde{\mathcal{Q}}_{t+1}(x_t) := \frac{1}{N_{t+1}} \sum_{j=1}^{\widetilde{q}_{t+1}} \widetilde{Q}_{t+1}(x_t, \widetilde{\xi}_{t+1,j}) \tag{35}$$

and $\widetilde{\mathcal{Q}}_{T+1} \equiv 0$. For the first stage, we obtain

$$\widetilde{v}_N := \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \widetilde{\mathcal{Q}}_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{36}$$

The DPE (34)-(36) can be approached by SDDP as described in Sect. 3. However, in contrast to the problems considered there, the SAA problems are random, as they depend on a sample from the true data process $(\boldsymbol{\xi}_t)_{t\in[T]}$.

SAA Properties. Since the aim is to solve the original problem $(\widetilde{P})$, the central question is how the solution and the bounds obtained by applying SDDP to the SAA problem $(\widetilde{P}_N)$ relate to the solution of $(\widetilde{P})$. We denote the optimal value of $(\widetilde{P})$ by $\widetilde{v}^*$ and the bounds obtained by SDDP in iteration i with $\underline{\widetilde{v}}^i$ and $\overline{\widetilde{v}}_{\mathcal{K}}^i$. We summarize important properties of SAA.

(P.11.1) *Consistency.* It can be shown that the optimal value $\widetilde{\boldsymbol{v}}_N$ provides a consistent estimator of the true optimal value $\widetilde{v}^*$, *i.e.*, $\lim_{\widetilde{q}_2,\dots,\widetilde{q}_T \to \infty} \widetilde{\boldsymbol{v}}_N = \widetilde{v}^*$ with probability 1 [198, 201]. The intuition behind this is that asymptotically, the structure of the true process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is recovered. In practical applications, increasing $\widetilde{q}_t$ to infinity is computationally intractable, though.

(P.11.2) *Bias.* $\widetilde{\boldsymbol{v}}_N$ is a biased estimator of $\widetilde{v}^*$, more precisely, $\mathbb{E}[\widetilde{\boldsymbol{v}}_N] \leq \widetilde{v}^*$ for all N [201], since only a subset of all scenarios is considered and the decisions are optimized with respect to these scenarios [48]. This means that solving the SAA problem provides a (converging) estimator of a lower bound for $\widetilde{v}^*$ [195].

(P.11.3) *Lower Bounds.* In each iteration i of SDDP, we have $\underline{\widetilde{v}}^i \leq \widetilde{v}_N$. Therefore, $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}^i] \leq \widetilde{v}^*$ [198], and the SDDP lower bound is a statistical lower bound for $\widetilde{v}^*$. Note, however, that both, $\widetilde{\boldsymbol{v}}_N$ and $\underline{\widetilde{\boldsymbol{v}}}^i$, are lower bounds in expectation only, whereas this is not clear for one specific SAA problem $(\widetilde{P}_N)$.

(P.11.4) *Upper Bounds.* Applying SDDP to the DPE (34)-(36) yields a policy. Under relatively complete recourse (see Assumption 9) with respect to the true data process $(\boldsymbol{\xi}_t)_{t\in[T]}$, this policy also yields feasible decisions if applied to any realization $(\xi_t)_{t\in[T]}$ of this true process. By computing

$$\mathbb{E}\left[\sum_{t=1}^{T} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t^i(\xi_{[t]})\right] \tag{37}$$

with the expectation taken with respect to the true process, a valid upper bound for $\widetilde{v}^*$ can be obtained [198].

(P.11.5) The sample mean $\overline{\widetilde{v}}_{\mathcal{K}}^i$ determined in iteration i in SDDP is an unbiased and consistent estimator of (37). Hence, $\mathbb{E}\left[\overline{\widetilde{\boldsymbol{v}}}_{\mathcal{K}}^i\right] \geq \widetilde{v}^*$.

Even with these theoretical properties, solving $(\widetilde{P})$ using SAA may be computationally intractable. Shapiro shows that even under relatively complete recourse (see Assumption 9) and stagewise independence (Assumption 2) of the true data process $(\boldsymbol{\xi}_t)_{t\in[T]}$, the total number of scenarios required in SAA problem $(\widetilde{P}_N)$ to solve $(\widetilde{P})$ with a reasonable accuracy $\varepsilon > 0$ grows exponentially in the number of stages [196]. Therefore, he proposes to use smaller sample sizes $\widetilde{q}_t$ for later stages, although then the accuracy of the solution cannot be guaranteed anymore [197].

Clearly, there exists a trade-off between the quality of the obtained bounds for $\widetilde{v}^*$ and the computational tractability of the SAA problem. Approximating $F_{\boldsymbol{\xi}}$ with F_N using very large sample sizes $\widetilde{q}_t$ for all $t = 2, \ldots, T$, a much better representation of the original process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is obtained, leading to a better approximation of $\widetilde{v}^*$. However, in this case, it may be even impossible to solve the SAA problem to optimality in reasonable time, as it may take too long until all scenarios are *eventually* sampled [198]. On the other hand, a very rough approximation yields a problem $(\widetilde{P}_N)$, which can be solved efficiently by SDDP, but does not provide reasonable information about the solution to the true problem $(\widetilde{P})$ [114].

11.2 Assessing Policy Quality

As it is computationally intractable to solve an SAA problem of $(\widetilde{P})$ with a sample size that guarantees a predetermined accuracy, in practice, usually moderate sample sizes are used. For example, in [48], sample sizes with branching numbers $\tilde{q}_t$ between 5 and 200 are tested.

The bounds $\underline{\widetilde{v}}^i$ and $\overline{\widetilde{v}}_{\mathcal{K}}^i$ in SDDP are determined using one specific sample of $(\boldsymbol{\xi}_t)_{t\in[T]}$. Therefore, they only measure the *in-sample* performance of the determined feasible policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$. To assess its quality for the original problem $(\widetilde{P})$, *i.e.*, its *out-of-sample* performance, it is required to evaluate it with respect to the original process $(\boldsymbol{\xi}_t)_{t\in[T]}$. Such an evaluation also allows one to compare policies obtained for different SAA problems, which can be helpful in designing appropriate sampling techniques and sample sizes [48].

Various techniques have been proposed in stochastic programming to measure the performance of feasible policies, such as analyzing optimality conditions, assessing solution stability or estimating the optimality gap [48]. Specifically for SDDP, Morton et al. have made substantial contributions [39, 48, 114], which are based on estimating the optimality gap ([114] analyzes a risk-averse variant of SDDP, see Sect. 12). We discuss their ideas for the risk-neutral case thoroughly in the remainder of this subsection. In accordance with [48], we only consider uncertainty in the RHS of $(\widetilde{P})$.

Estimating the Optimality Gap. For some feasible policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$, let $\widetilde{v}(\xi) = \sum_{t=1}^T c_t x_t\big(\xi_{[t]}\big)$ denote the random cost for some arbitrary scenario path $\xi = (\xi_1,\dots,\xi_T)$. From (P.11.4) we have $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)] \geq \widetilde{v}^*$. Therefore, the optimality gap induced by policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$ can be expressed as

$$\Delta := \mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)] - \widetilde{v}^* \geq 0.$$

This gap cannot be directly evaluated because the optimal value $\widetilde{v}^*$ is not known. Using some lower bound for $\widetilde{v}^*$, Δ can be overestimated though. Such lower bound is given by $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}]$, see (P.11.3). This yields

$$\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)] - \mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}] \geq \Delta \geq 0. \tag{38}$$

Still, the left-hand side of (38) is computationally infeasible to evaluate. It requires excessive computational effort to evaluate policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$ for all possible scenarios to obtain $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)]$. Furthermore, from SDDP only one specific realization of $\underline{\widetilde{v}}$ is known. Therefore, in [48] it is proposed to use estimators for both terms to derive an approximate one-sided confidence interval bounding Δ from above.

Upper Bound Estimation. The SDDP policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$ is feasible for the original problem $(\widetilde{P})$, see (P.11.4). Hence, it can be evaluated for any realization of $(\boldsymbol{\xi}_t)_{t\in[T]}$ to assess its out-of-sample performance. Let us sample M_u i.i.d. scenario paths from $(\boldsymbol{\xi}_t)_{t\in[T]}$. For each of those sampled scenarios $\xi^\ell, \ell = 1,\dots,M_u$, the SDDP subproblems (10) are solved in forward direction, yielding $x_t(\xi^\ell_{[t]})$ and $\widetilde{v}(\xi^\ell)$ [48]. An upper bound estimator is then defined by the sample mean

$$U_{M_u} := \frac{1}{M_u}\sum_{\ell=1}^{M_u} \widetilde{v}(\xi^\ell). \tag{39}$$

Similarly to the in-sample estimator, this estimator is an unbiased and consistent estimator of $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)]$. Its sample variance is given by [48]

$$\sigma_U^2 := \frac{1}{M_u - 1}\sum_{\ell=1}^{M_u} (\widetilde{v}(\xi^\ell) - U_{M_u})^2. \tag{40}$$

Alternatively, an upper bound estimator can be obtained by sampling a finite number of different SAA problems, and applying the SDDP policy $\big(\boldsymbol{x}_t(\xi_{[t]})\big)_{t\in[T]}$ to each of them [39]. This comes at the cost of increased computational effort.

Lower Bound Estimation with Several SAA Problems. From SDDP, only one single realization of $\underline{\widetilde{v}}$ is known. Hence, it is not directly possible to determine a sampling error for this point estimate and to derive a confidence interval for $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}]$.

One approach to derive a lower bound estimator is to solve a finite number of different SAA problems with SDDP and to determine the mean of the lower bounds $\underline{\widetilde{v}}$. To be precise, M_l different SAA problems are constructed, each by sampling $\widehat{q}_t$ realizations per stage from $(\boldsymbol{\xi}_t)_{t\in[T]}$. Then SDDP is run, yielding the lower bounds $\underline{\widetilde{v}}^\ell, \ell = 1,\dots,M_l$ [48]. The sample mean

$$L_{M_l} := \frac{1}{M_l}\sum_{\ell=1}^{M_l}\underline{\widetilde{v}}^\ell \tag{41}$$

then defines an estimator for $\mathbb{E}[\underline{\widetilde{\boldsymbol{v}}}]$ with sample variance

$$\sigma_l^2 := \frac{1}{M_l-1}\sum_{\ell=1}^{M_l}(\underline{\widetilde{v}}^\ell - L_{M_l})^2.$$

Note that instead of lower bounds $\underline{\widetilde{v}}^\ell$, also the optimal values $\widetilde{v}_N^\ell$ could be used in estimator (41) [48]. We already discussed in Sect. 11.1 that it may be computationally intractable to solve one single SAA problem to optimality, though. Thus, using $\underline{\widetilde{v}}^\ell$ may be computationally preferable.

In principle, applying SDDP to not only one, but several SAA problems and building the mean of the obtained bounds seems very reasonable from a statistical perspective, as the outcome of one SAA problem is random. This also has another possible benefit: If SDDP is run for M_l different SAA problems $(\widetilde{P}_N^l)$, each of these problems yields a different feasible policy. By calculating the upper bound estimator U_{M_u} (39) for each of them, directly M_l different policies could be compared.

However, for problems with multiple stages and for sufficiently high $\widehat{N}_t$, this becomes computationally intractable, even without solving $(\widetilde{P}_N^l)$ exactly. Therefore, de Matos et al. [48] follow the strategy to run SDDP once for some SAA problem with larger branch size $\widetilde{q}_t$ to determine a high quality policy and then, afterwards, to run SDDP for M_l SAA problems with smaller branch size $\widehat{q}_t$ only to produce the lower bound estimate L_{M_l} and assess the quality of that policy. In their numerical tests, they choose values between 5 and 200 for $\widetilde{q}_t$ and 5 for $\widehat{q}_t$. In general, it is not clear though, how to choose $\widehat{q}_t$ to reach a reasonable trade-off between computational tractability and an appropriate quality of the lower bound estimator.

Lower Bound Estimation with One SAA Problem. An alternative and less costly lower bound estimator is derived by only using the existing SAA problem, which has been applied to determine the policy that is to be assessed [48].

The idea is then to use the SDDP outcome $\underline{\widetilde{v}}$ as the point estimate L_{M_l} for the lower bound. To estimate the unknown sampling error of $\underline{\widetilde{v}}$, the sampling error of the in-sample upper bound estimator is used. This means that M_l scenarios are sampled from F_N (the SAA problem distribution), and formulas (39) and (40) with M_l in the role of M_u are used to compute an upper bound estimate $\overline{\widetilde{v}}_{M_l}$ and sample error σ_l^2. The idea behind applying this sampling error is that $\underline{\widetilde{v}}$ and $\mathbb{E}[\overline{\widetilde{\boldsymbol{v}}}_{M_l}]$ are equal if SDDP has been run to optimality. However, this also implies that if SDDP has not converged (or if $\widetilde{q}_t$ is not sufficiently large) the sampling error may be underestimated, and thus the confidence intervals drawn from this become overly optimistic [48].

Confidence Intervals. Using the bound estimators and their sample variances, asymptotically valid confidence intervals can be derived [48].

$$\left(-\infty, U_{M_u} + t_{M_u-1,\alpha}\frac{\sigma_U}{\sqrt{M_u}}\right]$$

is an asymptotically valid, and for finite M_u approximate, $(1-\alpha)\%$ confidence interval for $\mathbb{E}[\widetilde{\boldsymbol{v}}(\xi)]$. Here, $t_{M_u-1,\alpha}$ denotes the $(1-\alpha)$-level quantile of a student's t distribution with $M_u - 1$ degrees of freedom. Similarly,

$$\left[L_{M_l} - t_{M_l-1,\alpha}\frac{\sigma_l}{\sqrt{M_l}}, \infty\right)$$

is an asymptotically valid, and for finite M_l approximate, $(1-\alpha)\%$ confidence interval for $\widetilde{v}^*$. Using only one SAA problem, this confidence interval is only valid if SDDP has converged and if $\widetilde{q}_t$ is sufficiently large. Combining both intervals yields

$$\left[0, [U_{M_u} - L_{M_l}]_+ + t_{M_l-1,\alpha}\frac{\sigma_l}{\sqrt{M_l}} + t_{M_u-1,\alpha}\frac{\sigma_U}{\sqrt{M_u}}\right]$$

as a one-sided approximate confidence interval for the optimality gap Δ [48]. Here, $[x]_+ := \max\{x, 0\}$.

11.3 Variance Reduction Techniques

Instead of MC sampling, also importance sampling [150] and variance reduction techniques (see Sect. 6.2) can be applied to obtain SAA estimators with reduced bias and variance.

In [105], numerical tests comparing MC, LHS and RQMC indicate that RQMC yields the most promising results when it comes to determining representative SAA problems. In [48] also MC, LHS and RMC are compared for different branch sizes and policy evaluation strategies. The results indicate that with both LHS and RQMC, a reduction of bias and sampling error, a higher policy quality and tighter confidence intervals can be achieved in comparison with MC sampling, especially for smaller branch sizes $\widetilde{q}_t$. For smaller branch sizes LHS appears to be superior, while RQMC yields better results for larger branch sizes. While showing higher variability for MC sampling, if combined with RQMC and LHS sampling, the computationally preferable lower bound estimator using only in-sample scenarios from the existing SAA yields comparable results to the approach solving several SAA problems [48].

12 Risk-averse SDDP [relaxing Assumption 8]

In SDDP, as described in Sect. 3, a risk-neutral optimal policy is determined for (MSLP) (see Assumption 8). More precisely, (MSLP) minimizes the expectation of the total objective value over all stages $t \in [T]$ over feasible policies $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$, which satisfy non-anticipativity and all constraints. Hence, it can be formulated as the single problem

Equation (3) with objective

$$\min_{x_1, \boldsymbol{x_2}, \dots, \boldsymbol{x_T}} \mathbb{E}\left[\sum_{t \in [T]} \left(\boldsymbol{c}_t(\xi_t)\right)^\top \boldsymbol{x}_t(\xi_{[t]})\right]. \tag{42}$$

As discussed in Sect. 2.4, this problem can be expressed equivalently using the DPE (4)-(6). This equivalence is based on two important properties of expected values, first the so-called *tower property*

$$\mathbb{E}_{\boldsymbol{\xi}_t}[\boldsymbol{Z}_t(\xi_t)] = \mathbb{E}_{\xi_{[t-1]}}\left[\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{[t-1]}}[\boldsymbol{Z}_t(\xi_t)]\right] \tag{43}$$

for some random variable $\boldsymbol{Z}_t$, and second its strict monotonicity (see property (R2') below for a formal definition) [199].

Recall that the objective value $\sum_{t \in [T]} \left(\boldsymbol{c}_t(\xi_t)\right)^\top \boldsymbol{x}_t(\xi_{[t]})$ is random, and its realizations depend on realizations of $(\boldsymbol{\xi}_t)_{t \in [T]}$. For some specific realization, the SDDP policy may produce an objective value which widely deviates from the expectation in (42). In practice, decision makers are often anxious not only to find a policy causing low costs *on average*, but also to avoid the risk of extremely high cost situations. This motivates to consider *risk-averse* approaches in stochastic programming.

For multistage stochastic programming, incorporating risk-aversion has been a popular research topic in the last decade. This includes theoretical fundamentals on dynamic risk measures [192] as well as algorithmic developments, such as rolling horizon approaches with chance constraints or AVaR constraints, which take risk aversion into account in the constraints of (MSLP) [95, 96]. For SDDP, most focus has been on replacing expectations in the objective (42) with some multi-period risk measure $\mathcal{R}[\cdot]$ (see below for a formal definition). This yields the multistage risk-averse problem $(P_{\mathcal{R}})$:

$$\begin{aligned}
\min_{x_1, \boldsymbol{x_2}, \dots, \boldsymbol{x_T}} \quad & \mathcal{R}\left[\left(\boldsymbol{c}_1(\xi_1)\right)^\top \boldsymbol{x}_1(\xi_{[1]}), \dots, \left(\boldsymbol{c}_T(\xi_T)\right)^\top \boldsymbol{x}_T(\xi_{[T]})\right] \\
\text{s.t.} \quad & x_1 \in \mathcal{X}_1 \\
& \boldsymbol{x}_t \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_t \in \Xi_t \ \forall t = 2, \dots, T \\
& \boldsymbol{x}_t(\cdot) \ \mathscr{F}_t\text{-measurable} \quad \forall t = 2, \dots, T.
\end{aligned} \tag{44}$$

We cover risk-averse SDDP in detail in the remainder of this section, but start with some theoretical concepts.

12.1 Risk Measures

In this section, we introduce some required foundations of risk measures, especially for multistage problems. As our focus is on algorithmic aspects of SDDP, we refer to the comprehensive coverage of this topic in [199, 201] for technical definitions and derivations.

12.1.1 Static Risk Measures

A *static* (or *one-period*) risk measure is a function $\rho : \mathcal{Z} \to \bar{\mathbb{R}}$ from the space $\mathcal{Z}$ of random variables $\boldsymbol{Z}$ to $\bar{R} := \mathbb{R} \cup \{-\infty, +\infty\}$. Often, $\mathcal{Z}$ is assumed to be $\mathcal{L}_1(\Omega, \mathscr{F}, \mathbb{P})$, *i.e.*, the space of all $\mathscr{F}$-measurable functions with finite first moments, as this ensures well-definedness and finiteness of many common risk measures. Importantly, since random variables are functions themselves, risk measures are actually *functionals*. This is sometimes emphasized by calling them *risk functionals* or *risk mappings*.

We summarize some well-known static risk measures:

- The expected value $\mathbb{E}[\cdot]$ is the most common risk measure. It is completely risk-neutral.

- The *value-at-risk* $\mathrm{VaR}_\alpha[\cdot]$ to level $\alpha \in (0,1)$ is defined as the left-side $(1-\alpha)$-quantile of the cumulative distribution of some random variable $\boldsymbol{Z}$:

$$\mathrm{VaR}_\alpha[\boldsymbol{Z}] := \inf \left\{ u \in \mathbb{R} \ : \ \mathbb{P}(Z \leq u) \geq 1 - \alpha \right\}. \tag{45}$$

  Note that this definition is not used consistently in the literature, and that the RHS of (45) may also be defined as $\mathrm{VaR}_{1-\alpha}[\boldsymbol{Z}]$.

- The *average value-at-risk* $\mathrm{AVaR}_\alpha[\cdot]$ to level $\alpha \in (0,1)$ for some random variable $\boldsymbol{Z}$ is defined by [185]

$$\mathrm{AVaR}_\alpha[\boldsymbol{Z}] := \inf \left\{ u \in \mathbb{R} \ : \ u + \frac{1}{\alpha}\mathbb{E}\left[[Z-u]_+\right] \right\}, \tag{46}$$

  where $[x]_+$ is defined as $\max\{x, 0\}$. Note that the infimum is always attained in our SDDP setting of finite randomness (Assumption 5) and finite value functions $Q_t(\cdot)$ (see Lemma 2.5).

  Remark 12.1. *$AVaR_\alpha[\cdot]$ is also called* conditional value-at-risk, expected shortfall *or* expected tail loss. *In the literature on risk-averse stochastic programming, the first alternative is most frequently used with notation $CVaR_\alpha[\cdot]$, but to avoid confusion when we introduce conditional risk measures later, we stick to average value-at-risk.*

  Remark 12.2. *For finite random variables $\boldsymbol{Z}$ (under Assumption 5 for SDDP), $AVaR_\alpha[\cdot]$ may as well be defined as*

$$AVaR_\alpha[\boldsymbol{Z}] = \mathbb{E}\big[\boldsymbol{Z} | \boldsymbol{Z} \geq VaR_\alpha[\boldsymbol{Z}]\big].$$

  It can be shown that an equivalent formulation of $\mathrm{AVaR}_\alpha[\boldsymbol{Z}]$ is given by [198]

$$\mathrm{AVaR}_\alpha[\boldsymbol{Z}] = \mathrm{VaR}_\alpha[\boldsymbol{Z}] + \frac{1}{\alpha}\mathbb{E}\Big[\big[\boldsymbol{Z} - \mathrm{VaR}_\alpha[\boldsymbol{Z}]\big]_+\Big], \tag{47}$$

  i.e., $u^* = \mathrm{VaR}_\alpha[\boldsymbol{Z}]$ minimizes the RHS in (46).

  $\mathrm{AVaR}_\alpha[\cdot]$ has some beneficial properties compared to $\mathrm{VaR}_\alpha[\cdot]$. It does not only consider the probability mass beyond $\mathrm{VaR}_\alpha[\cdot]$, but also its distribution, *e.g.*, if it has fat or long tails. Moreover, it allows to retain convexity of optimization problems, as we discuss later on. $\mathrm{VaR}_\alpha[\cdot]$ and $\mathrm{AVaR}_\alpha[\cdot]$ are illustrated in Figure 10.

- In stochastic programming, often a convex combination of $\mathbb{E}[\cdot]$ and $\mathrm{AVaR}[\cdot]$ is considered, that is

$$\widehat{\rho}_{\alpha,\lambda}[\boldsymbol{Z}] := (1-\lambda)\mathbb{E}[\boldsymbol{Z}] + \lambda \mathrm{AVaR}_\alpha[\boldsymbol{Z}] \tag{48}$$

  for some $\lambda \in [0,1]$. The parameters λ and α control the risk-aversion. Choosing $\lambda = 0$ yields the standard risk-neutral model.

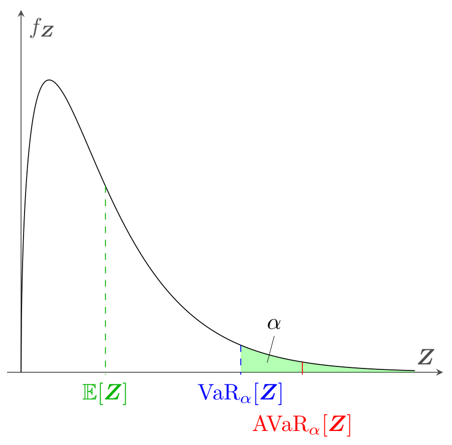


Figure 10: $\mathrm{VaR}_\alpha[\boldsymbol{Z}]$ and $\mathrm{AVaR}_\alpha[\boldsymbol{Z}]$ for a gamma distributed random variable $\boldsymbol{Z}$.

- For some $\gamma > 0$, the *entropic risk measure* is defined by

$$\mathbb{ENT}_\gamma[\boldsymbol{Z}] := \frac{1}{\gamma} \log\left(\mathbb{E}[e^{\gamma \boldsymbol{Z}}]\right). \tag{49}$$

  It generalizes $\mathbb{E}[\cdot]$ (for $\gamma \to 0$) and $\operatorname{ess\,sup}[\cdot]$ (for $\gamma \to \infty$), where $\operatorname{ess\,sup}[\boldsymbol{Z}]$ denotes the essential supremum of a random variable $\boldsymbol{Z}$.

It is often required that risk measures satisfy some special properties, especially in an optimization context. First, we assume that all considered risk measures are proper. Another desired property is *coherence*, a concept introduced by Artzner et al. [3]. We employ a slightly different definition from [201] and state it for the general case of continuous random variables:

Definition 12.3. *A risk measure $\rho : \mathcal{Z} \to \bar{\mathbb{R}}$ is called* coherent, *if it satisfies*

(R1) Convexity: for any $\boldsymbol{Z}_1, \boldsymbol{Z}_2 \in \mathcal{Z}$ and all $\lambda \in [0,1]$ it holds

$$\rho(\lambda \boldsymbol{Z}_1 + (1-\lambda)\boldsymbol{Z}_2) \leq \lambda \rho(\boldsymbol{Z}_1) + (1-\lambda)\rho(\boldsymbol{Z}_2),$$

(R2) Monotonicity: If $\boldsymbol{Z}_1 \leq \boldsymbol{Z}_2$ almost surely, then $\rho(\boldsymbol{Z}_1) \leq \rho(\boldsymbol{Z}_2)$,

(R3) Translation Equivariance: If $a \in \mathbb{R}$ and $\boldsymbol{Z} \in \mathcal{Z}$, then $\rho(\boldsymbol{Z} + a) = \rho(\boldsymbol{Z}) + a$,

(R4) Positive Homogeneity: If $\lambda > 0$ and $\boldsymbol{Z} \in \mathcal{Z}$, then $\rho(\lambda \boldsymbol{Z}) = \lambda \rho(\boldsymbol{Z})$.

A risk measure satisfying only properties (R1), (R2) and (R3) is called *convex*. In fact, a key feature of coherent risk measures is that they are convex, and thus convex objective functions as they appear in $(P_{\mathcal{R}})$ and its DPE remain convex if $\rho[\cdot]$ is applied to them. $\mathrm{VaR}_\alpha[\cdot]$ is not a coherent risk measure, but $\mathrm{AVaR}_\alpha[\cdot]$ is [157]. Therefore, in optimization $\mathrm{AVaR}_\alpha[\cdot]$ is usually preferred over $\mathrm{VaR}_\alpha[\cdot]$.

As we exploit later, for every coherent risk measure there exists a dual representation as the *worst-case expectation* over some class of probability distributions over $(\Omega, \mathscr{F})$ [3]. More precisely, let $\mathcal{P}$ be a convex set of probability measures, then a coherent risk

Table 4: Properties of common risk measures.

	(R1)	(R2)	(R3)	(R4)	(R2')	(R5)
$\mathbb{E}[\cdot]$	✓	✓	✓	✓	✓	✓
$\mathrm{VaR}_\alpha[\cdot]$	-	✓	✓	✓	-	✓
$\mathrm{AVaR}_\alpha[\cdot]$	✓	✓	✓	✓	-	✓
$\widehat{\rho}_{\alpha,\lambda}[\cdot]$	✓	✓	✓	✓	✓*	✓
$\mathbb{ENT}_\gamma[\cdot]$	✓	✓	✓	-	✓	✓

* only for $\lambda \in [0,1)$.

measure $\rho[\cdot]$ can be expressed as

$$\rho[Z] = \sup_{\mathbb{P}\in\mathcal{P}} \mathbb{E}_{\mathbb{P}}[Z]. \tag{50}$$

We introduce some additional relevant properties.

Definition 12.4. *Let $\rho : \mathcal{Z} \to \bar{\mathbb{R}}$ be some risk measure. Then, the following properties can be defined.*

(R2') If the inequalities in (R2) in Definition 12.3 are strict, we call this property strict monotonicity.

(R5) Law Invariance: $\rho[\cdot]$ is called law invariant *with respect to $\mathbb{P}$, if for all $\boldsymbol{Z}, \boldsymbol{Z}' \in \mathcal{Z}$ with the same distribution also $\rho(\boldsymbol{Z}) = \rho(\boldsymbol{Z}')$ holds.*

Property (R5) implies that the risk measure ρ only depends on the distribution of the considered random variable $\boldsymbol{Z}$.

We summarize properties of the previously introduced risk measures in Table 4.

Remark 12.5. *A classical approach in economics is to take risk aversion into account by means of non-decreasing and convex disutility (or concave utility) functions $g : \mathbb{R} \to \bar{\mathbb{R}}$ that are applied to some random variable $\boldsymbol{Z}$ before taking expectations. However, the obtained risk measure $\rho[\boldsymbol{Z}] = \mathbb{E}[g(\boldsymbol{Z})]$ does not satisfy property (R3) which is required to equivalently express $(P_{\mathcal{R}})$ using DPE.*

12.1.2 Multi-period Risk Measures

In a multistage setting, static, *i.e.*, one-period, risk measures have to be extended to several periods, more precisely, to a sequence of random variables $\boldsymbol{Z} := \boldsymbol{Z}_1, \dots, \boldsymbol{Z}_T$, which in our case model the stagewise objectives of (MSLP). We define such multi-period risk measures as functionals $\mathcal{R} : \mathcal{Z} \to \bar{\mathbb{R}}$ with $\mathcal{Z} = \mathcal{Z}_1 \times \mathcal{Z}_2 \times \ldots \times \mathcal{Z}_T$.

Choosing multi-period risk measures in a reasonable way is a challenging task. First, it is not clear how risk should be measured in a multistage setting [106]. Several different options exist [60, 106, 201], such as

$$\mathcal{R}[\boldsymbol{Z}] = \rho[\boldsymbol{Z}_1 + \cdots + \boldsymbol{Z}_T] \qquad \text{(end-of-horizon risk)} \tag{51}$$

$$\mathcal{R}[\boldsymbol{Z}] = \rho_1\Big[\boldsymbol{Z}_1 + \rho_{2|Z_1}\big[\boldsymbol{Z}_2 + \ldots + \rho_{T|Z_{T-1}}[\boldsymbol{Z}_T]\cdots\big]\Big] \qquad \text{(nested risk)} \tag{52}$$

$$\mathcal{R}[\boldsymbol{Z}] = \rho[\boldsymbol{Z}_1] + \ldots + \rho[\boldsymbol{Z}_T] \qquad \text{(stage-wise risk).} \tag{53}$$

Here, $\rho[\cdot]$ is some static risk measure, and $\rho_{t|Z_{t-1}}[\cdot], t = 2, \dots, T$, is a family of *conditional* risk measures, each mapping from $\mathcal{Z}_t$ to $\mathcal{Z}_{t-1}$ and defined as the static risk measure $\rho_t[\cdot]$ conditioned on $\mathscr{F}_{t-1}$ (or $\xi_{[t-1]}$, respectively). If $\rho_t[\cdot]$ is law-invariant (property (R5) in Definition 12.4), then $\rho_{t|Z_{t-1}}[\cdot]$ can be obtained by replacing the given distribution with the corresponding conditional distribution [201]. Usually the same static risk measure $\rho[\cdot]$ is chosen for all $\rho_t[\cdot], t = 2, \dots, T$. Note that coherence of conditional risk measures can be defined completely analogously to unconditional ones. The idea of nested conditional risk measures goes back to Ruszczyński and Shapiro [193].

Remark 12.6. *Under stagewise independence (Assumption 2), as we assume it for SDDP, the conditional risk measures $\rho_{t|Z_{t-1}}[\cdot]$ in* (52) *no longer depend on Z_{t-1}, and thus coincide with $\rho_t[\cdot]$ [201].*

Second, in an optimization context, multi-period risk measures have to be carefully chosen, in such a way that the resulting problem $(P_{\mathcal{R}})$ possesses desirable properties. In addition to convexity, especially time-consistency is a crucial property.

12.1.3 Time Consistency

In the literature, various different definitions of time consistency exist, see among others [35, 106, 46, 158, 199] and references within. The term is ambiguous in the sense that it is used for risk measures, policies and optimization problems. We only state some of these concepts that are relevant for SDDP, and for technical definitions and detailed discussions refer to [64, 106, 199, 201].

A common definition is that an optimal policy $\left(\bar{\boldsymbol{x}}_t(\xi_{[t]})\right)_{t\in[T]}$ for $(P_{\mathcal{R}})$ (see (44)) is called *time consistent* if for any $\tau \in [T]$, the policy $\left(\bar{\boldsymbol{x}}_t(\xi_{[t]})\right)_{t=\tau,\dots,T}$ is optimal for $(P_{\mathcal{R}})$ restricted to horizon $t = \tau, \dots, T$ conditional on $\mathscr{F}_{\tau-1}$ and $\bar{x}_{\tau-1}$ [201]. This means that the optimal policy remains optimal after some of the uncertain data has been revealed. The problem $(P_{\mathcal{R}})$ is then called *weakly time consistent*, if at least one of its optimal policies is time consistent, or *time consistent*, if every optimal policy is time consistent [201] (note that there exist deviating definitions in the literature).

Policies obtained using DPE (such as (4)-(6)) naturally satisfy time consistency. Therefore, the concept of time consistency is closely related to equivalently reformulating $(P_{\mathcal{R}})$ (see (44)) into DPE [201]. For nested risk measures $\mathcal{R}[\cdot]$, see (52), this equivalence holds under strict monotonicity (property (R2') in Definition 12.4) of ρ_t (or $\rho_{t|\xi_{[t-1]}}$, respectively) for all $t = 2, \dots, T$. More precisely, under (R2'), by interchanging risk measures and minimization operators, $(P_{\mathcal{R}})$ with nested risk can be expressed in the nested fashion [201]

$$
\begin{aligned}
\min_{x_1\in\mathcal{X}_1} c_1^\top x_1 + \rho_2\Bigg[\min_{\boldsymbol{x}_2\in\mathcal{X}_2(x_1)} (\boldsymbol{c}_2(\xi_2))^\top \boldsymbol{x}_2 + \rho_{3|\xi_{[2]}}\Bigg[\dots \\
\dots + \rho_{T|\xi_{[T-1]}}\Big[\min_{\boldsymbol{x}_T\in\mathcal{X}_T(x_{T-1})} (\boldsymbol{c}_T(\xi_T))^\top \boldsymbol{x}_T \Big] \dots \Bigg]\Bigg],
\end{aligned}
\tag{54}
$$

which naturally allows for a reformulation to DPE. Note that for stage 2 no conditional expectation is used as the first-stage data is deterministic. If ρ_t (or $\rho_{t|\xi_{[t]}}$) only satisfy (R2) instead of (R2'), then only weak consistency of $(P_{\mathcal{R}})$ is guaranteed, as any optimal policy for the DPE is also optimal for problem $(P_{\mathcal{R}})$ with nested risk, but not necessarily vice versa.

As indicated by Table 4, $\mathrm{AVaR}_\alpha[\cdot]$ is not strictly monotone. Therefore, even if applied in a nested conditional way, time consistency is not assured. In contrast, it can be ensured using risk measure $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ defined in (48), given that $\lambda \in [0,1)$. A drawback of nested risk is that it is less amenable to suitable interpretations, although some economic interpretations are possible [190].

For one-period risk measures $\rho[\cdot]$ that are applied as an end-of-horizon risk measure (51), it is well known that time consistency is often not satisfied. For instance, some simple examples in [64, 106] show that using a one-period risk measure $\rho[\cdot]$, such as $\mathrm{VaR}_\alpha[\cdot]$ or $\mathrm{AVaR}_\alpha[\cdot]$, in this setting leads to time-inconsistent decisions. Moreover, in [190], an illustrative example is presented in which even under stagewise independence (Assumption 2), the risk measure $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ does not yield time-consistent policies from an end-of-horizon perspective. To achieve time consistency, it is required that problem $(P_{\mathcal{R}})$ (see (44)) with end-of-horizon risk measure $\rho[\cdot]$ can be converted to an equivalent problem with nested risk using the corresponding conditional risk measures $\rho_{|\xi[t]}$. For this reason, Dowson et al. [60] define time consistency (in their case referred to as *conditional consistency*) of a one-period risk measure $\rho[\cdot]$ as an equivalence between the associated end-of-horizon risk and nested risk.

In fact, the only law-invariant coherent one-period risk measures $\rho[\cdot]$ allowing for such an equivalent reformulation between an end-of-horizon risk and a nested risk perspective are $\mathbb{E}[\cdot]$ and $\operatorname{ess\,sup}[\cdot]$ [201]. Therefore, the coherent and law-invariant risk measure $\mathrm{AVaR}_\alpha[\cdot]$ does not even guarantee weak time consistency for $(P_{\mathcal{R}})$ if it is applied as an end-of-horizon risk measure. It can be shown, though, that the non-coherent, but convex risk measure $\mathbb{ENT}_\gamma[\cdot]$ from (49) is conditionally consistent, and thus is sufficient to ensure time consistency of $(P_{\mathcal{R}})$. The equivalence of different formulations for problem $(P_{\mathcal{R}})$ is illustrated in Figure 11.

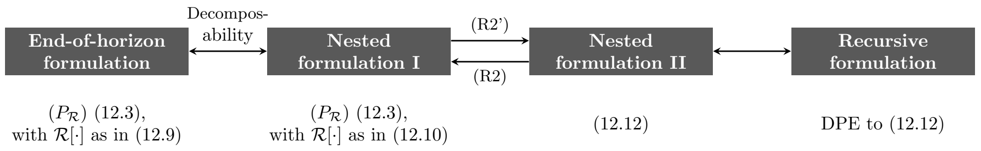

Decomposability
End-of-horizon formulation
Nested formulation I
(R2')
(R2)
Nested formulation II
Recursive formulation
$(P_{\mathcal{R}})$ (12.3), with $\mathcal{R}[\cdot]$ as in (12.9)
$(P_{\mathcal{R}})$ (12.3), with $\mathcal{R}[\cdot]$ as in (12.10)
(12.12)
DPE to (12.12)

Figure 11: Different forms of $(P_{\mathcal{R}})$ and conditions for their equivalence.

Remark 12.7. *In view of conditional consistency, note that nested risk measures $\mathcal{R}[\cdot]$ from* (52) *can always be expressed equivalently using an associated end-of-horizon risk measure* (51)*, the so-called* composite risk measure*. However, as the previous discussion shows, this composite risk measure only equals $\rho[\cdot]$ if the latter allows for a decomposition using its conditional analogues; similar to* (43) *[199, 201].*

Additionally, some notion of time consistency can be satisfied using *expected conditional risk measures* $\mathcal{R}[\cdot]$, which measure the risk stage by stage (see (53)), as long as the included (conditional) risk measures are coherent [106]. Applying such a risk measure in $(P_{\mathcal{R}})$ (problem (44)), we obtain the problem

$$\begin{aligned}
\min_{x_1,\boldsymbol{x_2},\dots,\boldsymbol{x_T}} \quad & c_1^\top x_1 + \rho_2\big[\big(\boldsymbol{c}_2(\xi_2)\big)^\top \boldsymbol{x}_2(\xi_{[2]})\big] + \mathbb{E}_{\xi_{[2]}}\Big[\rho_{3|\xi_{[2]}}\big[\big(\boldsymbol{c}_3(\xi_3)\big)^\top \boldsymbol{x}_3(\xi_{[3]})\big]\Big] \\
& \qquad + \dots + \mathbb{E}_{\xi_{[T-1]}}\Big[\rho_{T|\xi_{[T-1]}}\big[\big(\boldsymbol{c}_T(\xi_T)\big)^\top \boldsymbol{x}_T(\xi_{[T]})\big]\Big] \\
\text{s.t.} \quad & x_1 \in \mathcal{X}_1 \\
& \boldsymbol{x}_t \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}), \xi_t) \quad \forall \xi_t \in \Xi_t \ \forall t = 2,\dots,T \\
& \boldsymbol{x}_t(\cdot)\ \mathscr{F}_t\text{-measurable} \quad \forall t = 2,\dots,T.
\end{aligned} \tag{55}$$

12.1.4 Polyhedral Risk Measures

Multiperiod polyhedral risk measures $\mathcal{R}[\cdot]$ are a special type of risk measure, which for a time horizon of $T \in \mathbb{N}$ can be formulated as the optimal value of certain T-stage stochastic linear programs [67]. The arguments of the risk measure, *e.g.*, in our case the objective function of (MSLP), enter these linear programs on the RHS.

In [93], multiperiod extended polyhedral risk measures are introduced, for which the corresponding linear program has a slightly more general form. This class comprises polyhedral risk measures, spectral risk measures and also $\mathrm{AVaR}_\alpha[\cdot]$. These risk measures can be shown to be convex and coherent under certain assumptions [93].

The main strength of (extended) polyhedral risk measures is that they can naturally be used in a multistage stochastic programming setting. The LP representation of $\mathcal{R}[\cdot]$ and the original LP formulation of (MSLP) can be conflated to a single large-scale risk-neutral linear programming problem $(P_{\mathcal{R}})$, which allows for a reformulation by means of DPE [93].

12.2 Towards Considering Risk in SDDP

In the remainder of this section, we discuss the incorporation of risk-aversion into SDDP from an algorithmic perspective.

The first two methodological studies of risk-averse SDDP are [93] for problems with end-of-horizon risk (51), in particular using polyhedral risk measures, and [198] for problems with nested conditional risk mappings (52). Since then several extensions of SDDP have been proposed based on various risk measures. While some articles on this topic also cover SAA [114, 198, 204], see Sect. 11, we restrict to finite random variables here.

Remark 12.8 (SDDP with Polyhedral Risk Measures)**.** *As stated in Sect. 12.1.4, polyhedral risk measures have the advantage that DPE can be derived in a straightforward way. These DPE can then be approached by standard risk-neutral SDDP. Guigues and Römisch derive the associated cut formulas and give a convergence proof for some special cases of extended polyhedral risk measures [93] and the special case of spectral risk measures [94]. This approach to SDDP has been successfully applied for* $AVaR_\alpha[\cdot]$ *in [84].*

Despite this straightforward approach, polyhedral risk measures also pose a significant challenge to SDDP. The stage-t subproblems have to be enhanced with additional state variables z_{t-1} *and* $y_1, \dots, y_{t-1}$*, which are required to store the history of previous decisions. In general, this is unfavorable, as it may lead to prohibitive computational cost [161], compare Sect. 4.2. The specific computational cost depends on the chosen extended polyhedral risk measure.*

12.3 SDDP with Nested Risk Measures

As mentioned in Sect. 12.1.3, to obtain a risk-averse problem $(P_{\mathcal{R}})$ with time-consistent solutions, it is often proposed to use (conditional) coherent one-period risk measures $\rho[\cdot]$ (or $\rho_{t|\xi_{[t]}}[\cdot]$) for all $t \in [T]$ in a nested fashion. This yields the nested problem (54). We denote its optimal value by $v^*_{\mathcal{R}}$. As indicated before, we can derive an equivalent formulation using DPE [201]. Using Remark 12.6 they become

$$Q_{\mathcal{R},t}(x_{t-1},\xi_t) := \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t)\big)^\top x_t + \mathcal{Q}_{\mathcal{R},t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1},\xi_t) \end{cases} \tag{56}$$

with some *risk-adjusted value function*

$$\mathcal{Q}_{\mathcal{R},t+1}(x_t) := \rho_{t+1}\left[Q_{\mathcal{R},t+1}(x_t,\boldsymbol{\xi}_{t+1})\right] \tag{57}$$

and $\mathcal{Q}_{\mathcal{R},T+1}(\cdot) \equiv 0$. The corresponding first-stage problem is

$$v^*_{\mathcal{R}} = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathcal{Q}_{\mathcal{R},2}(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{58}$$

Fortunately, for coherent risk measures $\rho_t[\cdot], t \in [T]$, also the nested risk measure $\mathcal{R}[\cdot]$ preserves convexity of $\mathcal{Q}_{\mathcal{R},t+1}(\cdot)$. Therefore, a cutting-plane approximation as in SDDP can be applied.

Nested conditional risk measures are by far the most frequently chosen approach for risk-averse extensions of SDDP [64, 106, 114, 160, 161, 198, 204]. Most typically, the risk measure $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ (see Equation (48)) is used, which is coherent according to Table 4. For the remainder of Sect. 12.3, we therefore set $p_t[\cdot] = \widehat{\rho}_{\alpha_t,\lambda_t}[\cdot]$ for all $t \in [T]$, if not specified otherwise.

12.3.1 Reformulating the DPE

The general DPE for $(P_{\mathcal{R}})$ with nested risk measures are formulated in (56)-(58). To determine $\mathcal{Q}_t(\cdot), t \in [T]$, for $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ specifically, the AVaR of $Q_t(\cdot,\cdot)$ has to be evaluated. Using its definition as the optimal value of an optimization problem with decision variable $u \in \mathbb{R}$ [185], see (46), we are able to further reformulate the DPE.

Additional State Variable Approach. Using (46), the risk-adjusted value function (57) can be expressed as

$$\begin{aligned} \mathcal{Q}_{\mathcal{R},t+1}(x_t) = \min_{u_t \in \mathbb{R}}\ \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[&(1-\lambda_{t+1})Q_{\mathcal{R},t+1}(x_t,\boldsymbol{\xi}_{t+1}) \\ &+ \lambda_{t+1}\Big(u_t + \frac{1}{\alpha_{t+1}}\big[Q_{\mathcal{R},t+1}(x_t,\boldsymbol{\xi}_{t+1}) - u_t\big]_+\Big)\Big]. \end{aligned} \tag{59}$$

Recall that λ_t and α_t, $t = 2,\ldots,T$, are user-controlled parameters.

The minimization over u_t can be incorporated into the stage-t subproblems [198], which yields

$$\widetilde{Q}_{\mathcal{R},t}(x_{t-1},\xi_t) = \begin{cases} \min\limits_{x_t,u_t} & \big(c_t(\xi_t)\big)^\top x_t + \lambda_{t+1}u_t + \widetilde{\mathcal{Q}}_{\mathcal{R},t+1}(x_t,u_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1},\xi_t) \end{cases} \tag{60}$$

with some modified risk-adjusted value function

$$\begin{aligned}\widetilde{\mathcal{Q}}_{\mathcal{R},t+1}(x_t, u_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[&(1-\lambda_{t+1})\widetilde{Q}_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) \\ &+ \frac{\lambda_{t+1}}{\alpha_{t+1}}\big[\widetilde{Q}_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) - u_t\big]_+\Big],\end{aligned} \tag{61}$$

$\widetilde{\mathcal{Q}}_{\mathcal{R},T+1}(\cdot,\cdot) \equiv 0$ and $\lambda_{T+1} \equiv 0$ [198]. The first stage changes to

$$v_{\mathcal{R}}^* = \begin{cases} \min\limits_{x_1,u_1} & c_1^\top x_1 + \lambda_2 u_1 + \widetilde{\mathcal{Q}}_{\mathcal{R},2}(x_1, u_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{62}$$

The risk-adjusted value functions $\widetilde{\mathcal{Q}}_{\mathcal{R},t+1}(\cdot,\cdot)$ differ from the ones defined in (59), but can be proven to be convex as well.

With equations (60)-(62), the risk measures $\rho_{\alpha_t,\lambda_t}[\cdot]$ are incorporated into the subproblems, such that only expectations have to be evaluated in the DPE. However, as pointed out in [114], in comparison with the DPE (4)-(6) of the risk-neutral case, we still observe some fundamental differences: First, an additional, albeit one-dimensional, state variable $u_t \in \mathbb{R}$ is introduced at each stage to estimate the VaR-level, augmenting the state space by one. Second, the risk-adjusted value functions $\mathcal{Q}_{\mathcal{R},t+1}(\cdot,\cdot)$ do not only depend on x_t, but also on u_t and parameters λ_t, α_t. Third, they contain the nonlinear, *i.e.*, piecewise linear, function $[\cdot]_+$.

Philpott and de Matos provide an alternative reformulation of the DPE, eliminating the nonlinear expression via an epigraph reformulation [160]. To this end, the random term in the brackets in (61) is fully incorporated into the value functions. For $t = 2,\dots,T-1$, this yields

$$\begin{aligned}&\widehat{Q}_{\mathcal{R},t}(x_{t-1}, u_{t-1}, \xi_t) \\ &= \begin{cases} \min\limits_{x_t,u_t,w_t} & (1-\lambda_t)\Big(\big(c_t(\xi_t)\big)^\top x_t + \lambda_{t+1}u_t + \widehat{\mathcal{Q}}_{\mathcal{R},t+1}(x_t,u_t)\Big) + \dfrac{\lambda_t}{\alpha_t}w_t \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1},\xi_t) \\ & w_t - \big(c_t(\xi_t)\big)^\top x_t - \lambda_{t+1}u_t - \widehat{\mathcal{Q}}_{\mathcal{R},t+1}(x_t,u_t) \geq -u_{t-1}. \end{cases}\end{aligned} \tag{63}$$

Using this formulation, the risk value function is defined more naturally as

$$\widehat{\mathcal{Q}}_{\mathcal{R},t+1}(x_t, u_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[\widehat{Q}_{\mathcal{R},t+1}(x_t, u_t, \boldsymbol{\xi}_{t+1})\Big]. \tag{64}$$

Again, $\widehat{\mathcal{Q}}_{\mathcal{R},T+1}(\cdot,\cdot) \equiv 0$ and $\lambda_{T+1} \equiv 0$.

The first-stage problem reads then

$$v_{\mathcal{R}}^* = \begin{cases} \min\limits_{x_1,u_1,w_1} & c_1^\top x_1 + \lambda_2 u_1 + \widehat{\mathcal{Q}}_{\mathcal{R},2}(x_1, u_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{65}$$

In comparison to the formulation (60)-(62) by Shapiro [198], additional variables and constraints have to be introduced. Both formulations allow application of SDDP, but share the drawback of augmenting the state space. Since the computational effort of SDDP grows exponentially in the state space dimension, see Theorem 4.2, such increase should be avoided.

Modifying the Probability Measure. An alternative idea is to exploit that $u^* = \mathrm{VaR}_\alpha[\boldsymbol{Z}]$ in the definition of $\mathrm{AVaR}_\alpha[\boldsymbol{Z}]$ (see (46)) and that $\mathrm{VaR}_\alpha[\boldsymbol{Z}]$ is the $(1-\alpha)$-

quantile of a random variable $\boldsymbol{Z}$. As we assume finite randomness (Assumption 5) and solve the subproblems for all realizations $\xi_{tj}, j = 1, \dots, q_t$, in the backward pass of SDDP, this quantile can be manually determined for the value functions [204].

Without loss of generality, assume that for all $t = 2, \dots, T$ and any fixed trial solution $\bar{x}_{t-1}$ the values of $Q_{\mathcal{R},t}(\bar{x}_{t-1}, \xi_{tj})$ are ordered for all $j = 1, \dots, q_t$. That means, we have $Q_{\mathcal{R},t}(\bar{x}_{t-1}, \xi_{t1}) \leq \dots \leq Q_{\mathcal{R},t}(\bar{x}_{t-1}, \xi_{t,q_t})$. Then, in (59) the variable u_t can be replaced by the $(1-\alpha)$-quantile $Q_{\mathcal{R},t+1}(\bar{x}_t, \xi_{t+1,j^*})$ with j^* chosen such that $\sum_{j=1}^{j^*} p_{t+1,j} \geq 1 - \alpha_{t+1}$:

$$\begin{aligned} \mathcal{Q}_{\mathcal{R},t+1}(x_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}} \Big[(1-\lambda_{t+1}) Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) + \lambda_{t+1} \Big(Q_{\mathcal{R},t+1}(\bar{x}_t, \xi_{t+1,j^*}) \\ + \frac{1}{\alpha_t} \big[Q_{\mathcal{R},t+1}(x_t, \boldsymbol{\xi}_{t+1}) - Q_{\mathcal{R},t+1}(\bar{x}_t, \xi_{t+1,j^*}) \big]_+ \Big) \Big]. \end{aligned} \tag{66}$$

In SDDP, relation (66) cannot directly be applied, since $Q_{\mathcal{R},t+1}(\cdot, \xi_{t+1,j})$ is not known and also not evaluated for all $j = 1, \dots, q_{t+1}$. However, the same principle can also be applied to the approximate value functions $\underline{Q}_{\mathcal{R},t+1}(\cdot, \xi_{t+1,j})$.

In [161], this idea is considered from a dual perspective and used to reformulate the risk measure (48) even before formulating the DPE. The key idea is to use the dual representation of $\mathrm{AVaR}_\alpha[\cdot]$, see (50), which is given by

$$\mathrm{AVaR}_\alpha[\boldsymbol{Z}] = \begin{cases} \sup\limits_{\zeta} & \sum\limits_{j=1}^{q} p_j \zeta_j Z(\xi_j) \\ \text{s.t.} & \sum\limits_{j=1}^{q} p_j \zeta_j = 1 \\ & \zeta_j \geq 0, \quad j = 1, \dots, q \\ & \zeta_j \leq \dfrac{1}{\alpha}, \quad j = 1, \dots, q. \end{cases} \tag{67}$$

It shows that $\mathrm{AVaR}_\alpha[\cdot]$ can be interpreted as some worst-case probability measure $\widetilde{\mathbb{P}}$ with $\widetilde{p}_j := p_j \zeta_j$ for all $j = 1, \dots, q$.

As shown in [161], using this definition and explicitly computing the supremum, risk measure (48) can be written as

$$\widehat{\rho}_{\alpha_t, \lambda_t}[\boldsymbol{Z}_t] = \sum_{j=1}^{q_t} p_{tj} \zeta_{tj} Z_t(\xi_{tj}) \tag{68}$$

with

$$\zeta_{tj} = \begin{cases} (1-\lambda_t), & j < j^*, \\ (1-\lambda_t) + \dfrac{1}{p_{tj^*}} \Big(\lambda_t - \dfrac{\lambda_t}{\alpha_t} \sum\limits_{n=j^*+1}^{q_t} p_{tn} \Big), & j = j^*, \\ (1-\lambda_t) + \dfrac{\lambda}{\alpha_t}, & j > j^*. \end{cases} \tag{69}$$

Again, note that the true value functions $Q_t(\cdot)$ are not known explicitly in advance, and therefore the worst-case probability measure $\widetilde{\mathbb{P}}$ stemming from (67) is not known either. However, it can be approximated in SDDP. In particular, the DPE (56)-(58) and their approximations can be used with expectations as in standard SDDP, but with a modified probability measure that is iteratively updated. More precisely, as ζ_{tj} changes with $\bar{x}_{t-1}$, the modified probabilities have to be recomputed for each stage t, iteration i and sample k in SDDP. This principle is also extended to general coherent risk measures in [161].

Recently, this kind of change of the probability measure has also been discussed in [126]. Instead of determining the ordering and j^* based on $\underline{Q}_{t+1}^{i+1}(\cdot)$ for one specific iteration i, also all previous iterations are taken into account there. More precisely, the number of iterations in which an index j exceeds $\mathrm{VaR}_\alpha[\underline{Q}_{\mathcal{R},t+1}(\bar{x}_t, \boldsymbol{\xi}_t)]$ are counted. This is considered as a good proxy for the ordering of the actual value functions. The ordering, and thus the probability measure $\widetilde{\mathbb{P}}$, can either be updated dynamically within SDDP or be determined by running risk-averse SDDP once in advance to identify the outcomes contributing to $\mathrm{AVaR}_\alpha[\cdot]$. The latter approach has the advantage that the changed probability measure $\widetilde{\mathbb{P}}$ can be fixed for the following run, which yields a *risk-neutral* problem and allows for application of standard SDDP.

Additionally, as pointed out in [126], the approximation of $\widetilde{\mathbb{P}}$ may also be used in the forward pass to sample scenarios with "bad" outcomes with higher probability. This *biased sampling* can be considered similar to the importance sampling techniques presented in Sect. 6.

For the third-stage of Example 3.4, the expected risk value function $\mathcal{Q}_{\mathcal{R},3}(\cdot)$ obtained by applying (68) and (69) to (57) is illustrated in Figure 12 for $\alpha = 0.05$ and different values of λ. It can be seen that with choosing larger values for λ, representing a higher risk-aversion, the stage-3 cost increases compared to the risk-neutral case ($\lambda = 0$).

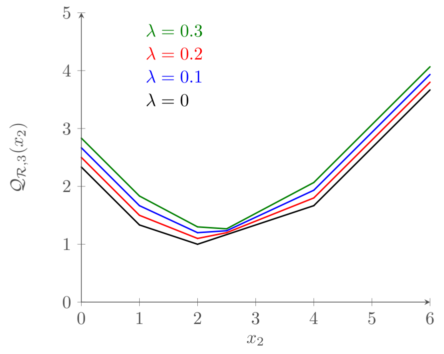


Figure 12: $\mathcal{Q}_{\mathcal{R},3}(\cdot)$ from Example 3.4 for $\alpha = 0.05$ and different values of λ.

As an overview, the different forms of DPE for $(P_\mathcal{R})$ using a nested (conditional) risk measure based on $\widehat{\rho}_{\alpha,\lambda}[\cdot]$ are summarized in Table 5.

12.3.2 Forward and Backward Pass

All approaches in Table 5 to formulate the DPE allow for a solution of a risk-averse problem $(P_\mathcal{R})$ using SDDP. Some approaches are more efficient, since the state space, the decision space or the number of constraints are not augmented. Others are advantageous in the sense that $\mathcal{Q}_{\mathcal{R},t}(\cdot)$ is expressed by a neat formula, and thus cut formulas can be derived more easily. With some epigraph reformulation, for all the approaches all subproblems can be formulated as LPs.

The forward pass of SDDP basically remains the same as for risk-neutral SDDP from Sect. 3. That is, $k \in \mathcal{K}$ scenarios are sampled and considered, with $\mathcal{K} \subset \mathcal{S}$ and

Description	Source	DPE
- general		(56)-(58)
- augmented state, sophisticated formula for $\mathcal{Q}_{\mathcal{R},t}(\cdot)$	[198]	(60)-(62)
- augmented state, additional constraints and variables	[160]	(63)-(65)
- $\mathrm{VaR}_{\alpha_t}[Q_t(\cdot)]$ explicitly determined, sophisticated formula for $\mathcal{Q}_{\mathcal{R},t}(\cdot)$	[204]	(60), (62), (66)
- modified probability measure	[161]	(56)-(58), (68)-(69)
- modified probability measure	[126]	(60), (62), (66)

Table 5: DPE formulations for $(P_{\mathcal{R}})$ using a nested (conditional) risk measure based on $\widehat{\rho}_{\alpha,\lambda}[\cdot]$.

$|\mathcal{K}| \ll |\mathcal{S}|$. However, the subproblems and the associated approximate value functions $\underline{Q}^i_{\mathcal{R},t}(x^{ik}_{t-1}, \xi^k_t)$ differ from the risk-neutral case. Instead of subproblems (10), one of the DPE from Table 5 are chosen and the occurring risk-adjusted value functions $\mathcal{Q}_{\mathcal{R},t+1}(\cdot)$ are replaced by cut approximations $\mathfrak{Q}^i_{\mathcal{R},t+1}(\cdot)$.

In the backward pass, as in risk-neutral SDDP, at each stage $t = T, \dots, 2$, those subproblems are solved for each trial solution $x^{ik}_{t-1}, k \in \mathcal{K}$, and possible stage-$t$ realization $\xi^k_{tj} \equiv \xi_{tj}, j = 1, \dots, q_t$, using an updated cut approximation $\mathfrak{Q}^{i+1}_{\mathcal{R},t+1}(\cdot)$. On stage t, a new cut for $\mathcal{Q}_{\mathcal{R},t}(\cdot)$ is derived and handed back to stage $t-1$. The main difference to risk-neutral SDDP is again the definition of $\mathcal{Q}_{\mathcal{R},t}(\cdot)$. Therefore, the cut formulas have to be adapted to the individual approach chosen. For the technical derivation of subgradients in such cases, we refer to the references in Table 5.

12.3.3 Upper Bound Determination and Stopping

A challenge in applying SDDP to risk-averse problems is to determine upper bounds for $v^*_{\mathcal{R}}$, and allowing for a reasonable stopping criterion. The reason is that most upper bound construction methods from the risk-neutral case, see Sect. 7 and 8, cannot be efficiently extended to the risk-averse case.

Recall that in the risk-neutral case, a feasible policy $(\boldsymbol{x}_t(\xi_{[t]}))_{t\in[T]}$ is determined in the backward pass and evaluated in the forward pass for different scenarios $k \in \mathcal{K}$, yielding a sequence of trial points $(x^{ik}_t)_{t\in[T]}$. Then, a statistical upper bound $\overline{v}_{\mathcal{K}}$ for v^* is determined as the sample average of the objective values of all these sample paths ξ^k, see (21). Analogously, a true upper bound $\overline{v}$ can be obtained by taking the expectation of such objective value for all scenarios $\xi^s, s \in \mathcal{S}$.

However, this is possible only due to the tower property (43) of expected values, which is required for the equivalence of the end-of-horizon formulation (42) and the nested formulation (54), see the discussion in Sect. 12.1.3. For most coherent risk measures this property does not hold, and thus a *direct analogue* to the (statistical) upper bound (21) from risk-neutral SDDP cannot be constructed.

As determining reasonable upper bounds is an important ingredient of SDDP, developing appropriate upper bound estimators has been an active research field in the last decade. In the following, we discuss different approaches that have been proposed. In reviewing them, we mostly follow the presentation of Kozmík and Morton [114], who provide a comprehensive study within their own work on upper bound estimators.

A Sample Average Estimator. In Sect. 12.3.1, we managed to formulate each $\rho_t[\cdot]$ only by means of expectations in (61). Still, this does not assure the tower property, since the risk-adjusted value functions $\mathcal{Q}_{\mathcal{R},t}(\cdot)$ contain a nested nonlinearity due to the $[\cdot]_+$-function. However, we can derive an estimator similar to (21) [114]. To this end, we remove the expectation in (61) to obtain

$$\begin{aligned}\hat{v}_t(\xi_t^k) := (1-\lambda_t)\Big(\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}(\xi_t^k)\Big) \\ + \lambda_t u_{t-1}^k + \frac{\lambda_t}{\alpha_t}\Big[\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}(\xi_t^k) - u_{t-1}^k\Big]_+,\end{aligned} \tag{70}$$

where we replace the value functions $Q_{\mathcal{R},t+1}(\cdot)$ by the estimator of the following stage. For stage T it follows $\hat{v}_{T+1}(\xi_T^k) \equiv 0$ and for the first stage

$$\hat{v}(\xi^k) := c_1^\top x_1 + \hat{v}_2(\xi_1^k). \tag{71}$$

Equation (71) provides a recursive estimator for the cost associated with sample path ξ^k. This estimator has to be evaluated by backward recursion starting with stage T. Importantly, formula (70) is only used for upper bound estimation, whereas the forward and backward problems in SDDP are still based on the original DPE (60)-(62). Determining estimator (71) for all scenarios $\xi^k, k \in \mathcal{K}$, sampled in the forward pass of SDDP, we can form an upper bound estimator

$$U^n := \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}} \hat{v}(\xi^k), \tag{72}$$

which resembles the sample average estimator (21) from risk-neutral SDDP.

It can be shown that $\mathbb{E}_{\boldsymbol{\xi}}[\hat{\boldsymbol{v}}(\xi)] \geq v_{\mathcal{R}}^*$ and that U^n is an unbiased and consistent estimator of $\mathbb{E}_{\boldsymbol{\xi}}[\hat{\boldsymbol{v}}(\xi)]$, so it is a statistical upper bound [114]. However, U^n is also observed to have a large variance. Kozmík and Morton [114] identify as the main reason that only a small portion of the sampled scenarios contributes to estimating $\text{AVaR}_\alpha[\cdot]$, while most scenarios solely contribute to the expectation. Therefore, a very large number of scenarios would be required for an appropriate estimate.

More crucially, because expectations are not taken conditionally on each stage as in (61), and due to to division by $\alpha_t \in (0,1)$, small or large values are very likely to propagate from late to earlier stages in the recursion to determine $\hat{v}(\xi^k)$ [114]. Therefore, the upper bound $\mathbb{E}_{\boldsymbol{\xi}}[\hat{\boldsymbol{v}}(\xi)]$ can significantly deviate from $\overline{v}_{\mathcal{R}}$, *i.e.*, the upper bound induced by the current policy in SDDP. In computational experiments, an upward bias is observed that makes U^n practically useless for large T [202].

Remark 12.9. *We should note that recently a very similar recursive upper bound estimator to $\hat{v}(\xi^k)$ and U^n has been proposed in [98], but for a general class of risk measures instead of only $\widehat{\rho}_{\alpha,\lambda}[\cdot]$. The main difference is that there SDDP is applied to a risk-averse stochastic optimal control model which deviates from our setting introduced in Sect. 2. In particular, states and controls are explicitly distinguished, and the decision on the controls is taken* before $\boldsymbol{\xi}_t$ *is realized. In this setting, the negative multiplicative effects observed in [114, 202] can be circumvented, and a computationally efficient statistical upper bound is obtained.*

Conditional Sampling Estimator. For the above reasons, estimator U^n in (72) is rarely considered in the literature on risk-averse SDDP. Instead, Shapiro discusses a conditional sampling estimator [198]. Here, the idea is to estimate the expectations (61) in the nested structure conditionally by sampling on each stage. Since in principle, the upper bound estimator can be determined independently of the scenarios sampled in the forward pass, we denote the set of samples by $\mathcal{M}$ instead of $\mathcal{K}$. $\mathcal{M}_t$ denotes the corresponding scenario set for stage t.

For each stage, $t = 2, \dots, T$, this yields [114]

$$\hat{v}_t^c(\xi_t^k) := \frac{1}{|\mathcal{M}_t|} \sum_{m_t \in \mathcal{M}_t} \Bigg[(1-\lambda_t)\Big(\big(c_t(\xi_t^{m_t})\big)^\top x_t^{m_t} + \hat{v}_{t+1}^c(\xi_t^{m_t}) \Big) \\ + \lambda_t u_{t-1}^{m_t} + \frac{\lambda_t}{\alpha_t} \Big[\big(c_t(\xi_t^{m_t})\big)^\top x_t^{m_t} + \hat{v}_{t+1}^c(\xi_t^{m_t}) - u_{t-1}^{m_t} \Big]_+ \Bigg],$$

and for the first stage the estimator

$$U^c := c_1^\top x_1 + \hat{v}_2^c(\xi_1).$$

As Shapiro himself points out, this estimator has two significant drawbacks. It requires $\prod_{t=2}^T |\mathcal{M}_t| + 1$ subproblems to be solved, which is exponentially growing in the number of stages. Moreover, the obtained upper bounds are typically not very tight. Therefore, estimator U^c should not be useful for large-scale problems [114].

Importance Sampling Estimators. Some of the drawbacks of estimator U^n may also be addressed by importance sampling [113, 114], see Sect. 6 for an introduction. By sampling scenarios associated with $\mathrm{AVaR}_\alpha[\cdot]$ with higher importance, it is possible to better represent it, and thus reduce the variance of the estimator. Based on this idea, Kozmík and Morton put forward different importance sampling upper bound estimators [114], which are further enhanced in [113].

Using importance sampling with respect to $\mathrm{AVaR}_\alpha[\cdot]$ creates a considerable challenge, though. In order to determine the importance sampling distribution for some stage t, it has to be identified which scenarios are associated with $\mathrm{AVaR}_\alpha[\cdot]$ on that stage, *i.e.*, which of them provide a value $Q_{\mathcal{R},t}(x_{t-1}^k, \xi_{tj}^k)$ beyond the $(1-\alpha)$-quantile. If we estimate this by solving subproblems for several ξ_{tj}^k and determining $Q_{\mathcal{R},t}(x_{t-1}^k, \xi_{tj}^k)$, we face a similar computational burden as for conditional sampling.

Kozmík and Morton propose the following approach: They use an *approximation function* $d_t(x_{t-1}, \xi_t)$, which estimates the recourse value of the decisions x_{t-1} after ξ_t has been observed [114]. Instead of solving the subproblems for several ξ_{tj}^k, they simply evaluate $d_t(x_{t-1}^k, \xi_{tj}^k)$ and sort these values. Based on the obtained order, it can be decided then which scenarios are used to estimate $\mathrm{AVaR}_\alpha[\cdot]$, *i.e.*, $u_t := \mathrm{VaR}_{\alpha_t}[d_t(x_{t-1}, \boldsymbol{\xi}_t)]$ is determined.

This allows defining an importance sampling distribution depending on x_{t-1} [114]. For simplicity, we assume that all scenarios are equally likely in the original distribution, that is, $f_t(\xi_{tj}) = \frac{1}{q_t}$ for all $j = 1, \dots, q_t$. Then, it follows:

$$g_t(\boldsymbol{\xi}_t | x_{t-1}) := \begin{cases} \dfrac{1}{2\lfloor \alpha_t q_t \rfloor}, & d_t(x_{t-1}, \boldsymbol{\xi}_t) \geq u_t, \\ \dfrac{1}{2(q_t - \lfloor \alpha_t q_t \rfloor)}, & d_t(x_{t-1}, \boldsymbol{\xi}_t) < u_t. \end{cases}$$

This distribution ensures that it is equally likely to draw sample observations above and below u_t. Note that the formula presented in [114] looks a bit different, since it is presented in the context of SAA.

Defining weights

$$\Lambda_t(\boldsymbol{\xi}_t|x_{t-1}) := \frac{f_t(\boldsymbol{\xi}_t)}{g_t(\boldsymbol{\xi}_t|x_{t-1})}$$

and multiplying them along the sample paths

$$\Lambda(\xi^k) := \prod_{t=2}^{T} \Lambda_t(\xi_t^k|x_{t-1})$$

we can derive the estimator

$$U^i := \frac{1}{\sum_{k\in\mathcal{K}} \Lambda(\xi^k)} \sum_{k\in\mathcal{K}} \Lambda(\xi^k)\hat{v}(\xi^k). \tag{73}$$

This estimator is similar to (72), as the same recursive term $\hat{v}(\xi^k)$ is used, but combined with importance instead of standard MC sampling.

With the assumptions of relatively complete recourse (based on Assumption 9) and stagewise independence (Assumption 2), estimator (73) is asymptotically valid, *i.e.*, for $|\mathcal{K}| \to \infty$, U^i converges to $\mathbb{E}_f[\hat{\boldsymbol{v}}(\xi)]$ with probability 1 (recall that $\mathbb{E}_f[\hat{\boldsymbol{v}}(\xi)] \geq v^*_{\mathcal{R}}$). Moreover, for sufficiently good choice of $d_t(\cdot)$, it can be expected that the variance is lower than for U^n [114].

Based on this idea, even better estimators are developed in [113, 114], for example by not only sampling scenarios associated with $\text{AVaR}_\alpha[\cdot]$ with higher priority, but also using only scenarios which contribute to the $[\cdot]_+$-term to estimate AVaR [114]:

$$\begin{aligned}\hat{v}_t^d(\xi_t^k) := (1-\lambda_t)\Big(\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}^d(\xi_t^k)\Big)\\ + \lambda_t u_{t-1}^k + \mathcal{I}[d_t(x_{t-1},\xi_t) \geq u_d]\frac{\lambda_t}{\alpha_{t-1}}\Big[\big(c_t(\xi_t^k)\big)^\top x_t^k + \hat{v}_{t+1}^d(\xi_t^k) - u_{t-1}^k\Big]_+.\end{aligned}$$

Here $\mathcal{I}[\cdot]$ denotes an indicator function. For the first stage it follows

$$\hat{v}^d(\xi^k) := c_1^\top x_1 + \hat{v}_2^d(\xi_1^k).$$

Combining this with (39), we obtain

$$U^d := \frac{1}{\sum_{k\in\mathcal{K}} \Lambda(\xi^k)} \sum_{k\in\mathcal{K}} \Lambda(\xi^k)\hat{v}^d(\xi^k).$$

The practical applicability of this estimator relies heavily on satisfaction of the following goodness assumption with respect to $d_t(\cdot)$:

$$Q_{\mathcal{R},t}(x_{t-1},\xi_t) \geq \text{VaR}_{\alpha_t}[Q_{\mathcal{R},t}(x_{t-1},\xi_t)] \;\Leftrightarrow\; d_t(x_{t-1},\xi_t) \geq \text{VaR}_{\alpha_t}[d_t(x_{t-1},\xi_t)],$$

which means that $d_t(\cdot)$ correctly classifies whether a realization is in the upper α-tail of the recourse value distribution.

It is proven that this estimator is asymptotically valid as well, but also provides tighter upper bounds than U^i in expectation, as long as the above goodness assumption is satisfied. Moreover, a smaller variance should be expected [114]. Numerical results in [114] illustrate that even for a medium number of stages, estimator U^d provides significantly better upper bounds than U^n, U^c and U^i and that also the variance of the estimators is reduced significantly. However, despite reducing the variance, even U^i and U^d may still show a considerable upward bias with respect to the upper bound $\overline{v}_{\mathcal{R}}$ induced by the current policy [202].

Apart from the above sampling estimators, some completely different strategies may be used to obtain upper bounds for $v^*_{\mathcal{R}}$ or to define some stopping criteria for SDDP in the risk-averse case.

Using Deterministic Upper Bounds. As already discussed in Sect. 8, we may circumvent the determination of sampling-based upper bound estimators completely if we resort to deterministic upper bounding procedures.

To this end, Philpott et al. [161] extend their inner approximation based upper bounding procedure from Sect. 8 to the risk-averse case with nested (conditional) coherent risk measures. The main downside of this procedure, to require prohibitively large computational effort for a large number of state variables and an increasing number of cuts, also holds in this case, though.

The alternative deterministic upper bounding procedure based on dual SDDP [97, 119] has been extended to a risk-averse setting as well [40].

Determining Bad Outcomes in Advance. As discussed in Sect. 12.3.1, following the approach of a change of probability measure, see (56)-(58) and (68), it is also possible to run (risk-averse) SDDP once in advance to approximate the probability measure $\widetilde{\mathbb{P}}$, and then a second time, this time fixing the probability measure to the approximation of $\widetilde{\mathbb{P}}$. This is referred to as solving the *change-of-measure risk-neutral problem* in [126]. Whereas this approach has a lot of computational overhead, the advantage is that a risk-neutral problem can be solved by SDDP and therefore, also the standard stopping, upper bounding and policy assessment techniques can be applied. Clearly, solving the change-of-measure risk-neutral problem is not guaranteed to yield optimal policies for $(P_{\mathcal{R}})$, however Liu and Shapiro report that the quality of the policies is similar to those obtained by risk-averse SDDP [126].

Fixing the Number of Iterations. This approach is proposed by Philpott and de Matos [160]. They run a risk-neutral variant of SDDP first and then fix the number of iterations required until termination. The same number of iterations is then used in the risk-averse case, avoiding the challenge of upper bound evaluation.

In some practical applications, in which it is computationally intractable to determine a sophisticated upper bound estimator, this approach may be useful. Promising results are reported in [160]. However, there is no theoretical guarantee to find a sufficiently good solution for a risk-averse version of $(P_{\mathcal{R}})$ in the same number of iterations as for a risk-neutral version. Additionally, for large problems it may already take considerably long to run SDDP one time. Running it a second time for risk-averse problem $(P_{\mathcal{R}})$ may partially annihilate the computational advantage of avoiding upper bound estimation.

Lower Bound Stabilization. As for risk-neutral SDDP, instead of using upper bounds at all, the algorithm can be terminated, once the lower bounds $\underline{v}^i_{\mathcal{R}}$ stabilize. This provides no convergence guarantee but may be worthwhile in large-scale practical applications where other approaches become computationally prohibitive.

Using Benefit Factors. Instead of the lower bounds $\underline{v}^i_{\mathcal{R}}$, it is also possible to condition termination of SDDP on the improvements of the cut approximations

$\mathfrak{Q}^i_{\mathcal{R},t}(\cdot), t = 2, \ldots, T$. For that purpose, Brandi et al. define a benefit factor

$$\mathcal{B}^i_{t,k} = \min \left\{ 1, \frac{\delta(x^{ik}_{t-1})}{\delta^i_{t,\max}} \right\},$$

which determines how much a new cut improves the current cut approximation $\mathfrak{Q}^i_{\mathcal{R},t}(\cdot)$ at x^{ik}_{t-1} [30]. $\delta(x^{ik}_{t-1})$ is the absolute increase, while $\delta^i_{t,\max}$ is a proxy for the maximum improvement possible. For each sample path $k \in \mathcal{K}$, a total benefit factor can be determined by

$$\mathcal{B}^i_k = \max \left\{ \mathcal{B}^i_{2,k}, \mathcal{B}^i_{3,k}, \ldots, \mathcal{B}^i_{T,k} \right\}.$$

The risk-averse SDDP method is then stopped if the values $\mathcal{B}^i_k$ for all $k \in \mathcal{K}$ are below a predefined tolerance, either for one iteration or, alternatively and more robustly, for a predefined larger number of iterations.

12.4 SDDP with Entropic Risk Measure

As discussed before, nested risk measures come with some drawbacks. Computation-wise, upper bound determination is very challenging. Additionally, applying a standard one-period risk measure $\rho[\cdot]$, *e.g.*, $\mathrm{AVaR}_\alpha[\cdot]$, as an end-of-horizon risk measure (51) and (possibly conditionally) in a nested risk measure (52) does not yield equivalent policies [60] (this is only the case if we take the composite risk measure associated with the nested risk measure as end-of-horizon risk; however, this risk measure is usually not known explicitly, see Remark 12.7). This makes nested risk measures difficult to interpret from an end-of-horizon perspective.

For this reason, Dowson et al. [60] propose to apply one-period *conditionally consistent* risk measures in the context of SDDP [60], see also [11, 158]. It can be proven that under some technical assumptions, the class of entropic risk measures $\mathbb{ENT}_\gamma[\cdot]$ (see (49)) is the only class of risk measures that is conditionally consistent.

As $\mathbb{ENT}_\gamma[\cdot]$ can be applied in a nested fashion, the DPE (56)-(58) are valid in this case. Moreover, since $\mathbb{ENT}_\gamma[\cdot]$ is a convex risk measure, the (risk-adjusted) value functions are convex. Therefore, SDDP can be applied to derive polyhedral outer approximations.

As for standard SDDP, first, for each scenario $k \in \mathcal{K}$ and all possible stage-t realizations $\xi^k_{tj} \equiv \xi_{tj}, j = 1, \ldots, q_t$, approximate versions of subproblems (56) are solved to obtain $\underline{Q}^i_{\mathcal{R},t}(x^k_{t-1}, \xi_{tj})$. Then, based on the dual form of $\mathbb{ENT}_\gamma[\cdot]$, the following auxiliary problem can be solved to evaluate the risk-adjusted value function:

$$\begin{aligned}&\mathbb{ENT}_\gamma \left[\underline{Q}^i_{\mathcal{R},t}(x^k_{t-1}, \xi_t) \right] \\ &= \begin{cases} \max\limits_{\tilde{p}_t} & \sum\limits_{j=1}^{q_t} \tilde{p}_{tj} \underline{Q}^i_{\mathcal{R},t}(x^k_{t-1}, \xi_{tj}) - \dfrac{1}{\gamma_t} \sum\limits_{j=1}^{q_t} \tilde{p}_{tj} \cdot \log \left(\dfrac{\tilde{p}_{tj}}{p_{tj}} \right) \\ \text{s.t.} & \sum\limits_{j=1}^{q_t} \tilde{p}_{tj} = 1 \\ & \tilde{p}_{tj} \geq 0, \quad j = 1, \ldots, q_t. \end{cases}\end{aligned} \tag{74}$$

Here, parameter p_{tj} denotes the nominal probabilities of realizations ξ_{tj}, which usually equal $\frac{1}{q_t}$, and the decision variable $\tilde{p}_{tj}$ denotes an alternative probability based on the entropic risk measure. In this way, problem (74) can be regarded as building the expectation based on some modified probability measure and with some additional penalty term. Problem (74) can be solved algorithmically, but as stated in [60], also a closed form for $\tilde{p}_{tj}^*$ can be derived. Using $\tilde{p}_{tj}^*$ and $\mathbb{ENT}_\gamma\big[\underline{Q}_{\mathcal{R},t}^i(x_{t-1}^k, \xi_t)\big]$, cuts can then be constructed and handed back to the previous stage.

The entropic risk measure does not only ensure conditional consistency of the obtained policies, but it also allows for upper bound computation as in standard SDDP, because the tower property can be employed for $\mathbb{ENT}[\cdot]$. However, these advantages come at the cost of an aggravated interpretation of the risk measure compared to AVaR-based ones. In this context, it is particularly difficult to make a reasonable choice for the parameter $\gamma_t > 0$ [60].

12.5 SDDP with Expected Conditional AVaR

Another class of multi-period risk measures that can be used as an alternative to nested risk measures are *expected conditional risk measures*, which we briefly introduced in Sect. 12.1.3 [64, 106]. Here, conditional expectations are used to avoid the risk measure nesting, which proves beneficial in determining upper bounds in SDDP, as it avoids the aforementioned computational difficulties, while still time consistency is ensured.

Recall the risk-averse problem ($P_{\mathcal{R}}$) using expected conditional risk measures stated in (55). Using $\rho_t[\cdot] = \mathrm{AVaR}_{\alpha_t}[\cdot]$ yields the so called $\mathbb{E}$-AVaR or *multi-period average value-at-risk* [106], which goes back to Pflug and Ruszczyński [159].

As stated in [106], by some lengthy reformulations, the objective function of problem (55) can be expressed in a nested way. Therefore, equivalent DPE can be derived and time consistency is assured. Moreover, the $[\cdot]_+$-function can be reformulated by an epigraph approach. Then, for $t = 2, \dots, T$, the DPE read

$$\check{Q}_{\mathcal{R},t}(x_{t-1}, u_t, \xi_t) = \begin{cases} \min\limits_{x_t, u_{t+1}, w_t} & \dfrac{1}{\alpha_t} w_t + u_{t+1} + \check{\mathcal{Q}}_{\mathcal{R},t+1}(x_t, u_{t+1}) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \\ & w_t - \big(c_t(\xi_t)\big)^\top x_t \geq -u_t \\ & w_t \geq 0 \end{cases} \tag{75}$$

with

$$\check{\mathcal{Q}}_{\mathcal{R},t+1}(x_t, u_{t+1}) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[\check{Q}_{\mathcal{R},t}(x_{t-1}, u_t, \boldsymbol{\xi}_t)\Big], \tag{76}$$

$\check{\mathcal{Q}}_{\mathcal{R},T+1}(\cdot,\cdot) \equiv 0$ and first stage

$$v_{\mathcal{R}}^* = \begin{cases} \min\limits_{x_1, u_2} & c_1^\top x_1 + u_2 + \check{Q}_{\mathcal{R},2}(x_1, u_2, \xi_t) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{77}$$

In contrast to using nested conditional risk measures, the DPE here only depend on nested sums of (conditional) expectations, *i.e.*, have the same structure as in the risk-neutral case. Hence, standard SDDP can be applied. This has the advantage to allow one to use upper bounding techniques developed for risk-neutral SDDP.

12.6 Bi-objective SDDP

An alternative to risk-averse formulations that allows one to achieve a trade-off between obtaining the best policy in expectation (*e.g.*, the policy with the lowest expected costs) and avoiding bad extreme outcomes (*e.g.*, power outages or load shedding in an electricity network) is to formulate a multistage problem (MSLP) with multiple competing objectives that are optimized simultaneously. Recently, a variant of SDDP for bi-objective problems has been put forward by Dowson et al. [58].

Let $\widetilde{c}_t(\xi_t)$ and $\widehat{c}_t(\xi_t)$ denote the objective coefficients for stage $t \in [T]$ and the two competing objectives. For all but trivial cases, there exists no policy which yields the best objective value with respect to both objectives

$$\widetilde{v}^* := \min_{x_1, \boldsymbol{x_2}, \dots, \boldsymbol{x_T}} \underbrace{\mathbb{E}\left[\sum_{t \in [T]} \left(\widetilde{\boldsymbol{c}}_t(\xi_t)\right)^\top \boldsymbol{x}_t(\xi_{[t]})\right]}_{=:\widetilde{v}(\boldsymbol{x})}$$

and

$$\widehat{v}^* := \min_{x_1, \boldsymbol{x_2}, \dots, \boldsymbol{x_T}} \underbrace{\mathbb{E}\left[\sum_{t \in [T]} \left(\widehat{\boldsymbol{c}}_t(\xi_t)\right)^\top \boldsymbol{x}_t(\xi_{[t]})\right]}_{=:\widehat{v}(\boldsymbol{x})},$$

meaning that the two objectives are truly conflicting.

For this reason, if there is no clear preference for one of the objectives, usually the aim is to compute *Pareto-optimal* policies. A policy $\left(\bar{\boldsymbol{x}}_t(\xi_{[t]})\right)_{t \in [T]}$ is Pareto-optimal if it cannot be improved in one objective without getting worse in the other one, *i.e.*, if there exists no other policy $\left(\boldsymbol{x}_t(\xi_{[t]})\right)_{t \in [T]}$ such that $\widetilde{v}(\boldsymbol{x}) \geq \widetilde{v}(\bar{\boldsymbol{x}})$ and $\widehat{v}(\boldsymbol{x}) > \widehat{v}(\bar{\boldsymbol{x}})$ (or the other way around). Pareto-optimal solutions are also called *non-dominated*, and the set of non-dominated objective vectors is called the *Pareto front* [58].

A standard approach to compute Pareto-optimal solutions in optimization is to use some *scalarization* approach in which both conflicting objectives are combined to a weighted sum, which is then optimized in a deterministic single-objective problem. In our case, the DPE (4)-(6) can be adapted to

$$Q_t(x_{t-1}, \xi_t, \lambda) := \begin{cases} \min\limits_{x_t} & \left(\lambda \widetilde{c}_t(\xi_t) + (1-\lambda)\widehat{c}_t(\xi_t)\right)^\top x_t + \mathcal{Q}_{t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases} \tag{78}$$

where

$$\mathcal{Q}_{t+1}(x_t, \lambda) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1}, \lambda)\right] \tag{79}$$

and $\mathcal{Q}_{T+1}(x_T) \equiv 0$. For the first stage, we obtain

$$v^*(\lambda) = \begin{cases} \min\limits_{x_1} & \left(\lambda \widetilde{c}_1 + (1-\lambda)\widehat{c}_1\right)^\top x_1 + Q_2(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{80}$$

SDDP can then be applied to these DPE. In the proposed variant, λ is adapted dynamically. To this end, in each iteration i, after the backward pass, one stage $t \in [T]$ is randomly and independently sampled and the corresponding subproblem is solved again for x_{t-1}^k, ξ_t^k and λ^i. Then, λ^i is updated to λ^{i+1}, where the latter is determined as the closest λ to λ^i such that the optimal basis of the constraint equation system changes.

It is proven that this variant of SDDP converges almost surely to the Pareto front of bi-objective (MSLP) in finitely many iterations. Note that technically speaking not *all* Pareto-optimal policies are guaranteed to be identified by SDDP because for some λ multiple optimal policies may exist. However, all Pareto-optimal policies for which $(\widehat{v}(\boldsymbol{x}), \widetilde{v}(\boldsymbol{x}))$ cannot be represented as a strict convex combination of other non-dominated objective vectors are identified under weak assumptions [58].

13 SDDP with Unknown Distribution [relaxing Assumption 3]

In Sect. 3 we introduced SDDP assuming that the probability distribution $F_{\boldsymbol{\xi}}$ of the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ governing the uncertainty in problem (MSLP) is known, see Assumption 3. This allowed us to sample from this specific distribution in the forward pass of SDDP or, in case of continuous random vectors, to obtain a finite sample average approximation, as described in Sect. 11.

In practical applications, usually, the true distribution $F_{\boldsymbol{\xi}}$ is not known, though. Often, only historical data is available, *i.e.*, some realization of an unknown true distribution. This data is then used to determine a reasonable estimate for the true distribution, from which the required samples are taken. However, using such an estimation imposes the risk of *overfitting* the SDDP policies to this specific distribution, and thus the available data. Philpott et al. [162] identify this problem as particularly noteworthy if the number of possible outcomes q_t per stage is small. For this reason, it may be reasonable to take a more robust approach and factor in the distributional uncertainty. Considering this type of uncertainty in SDDP is a young research area.

13.1 Distributionally Robust SDDP

One way to consider distributional uncertainty in SDDP is by integrating ideas from robust optimization [16, 20] into (multistage) stochastic programming. More precisely, a set of potential distributions is considered, which is called *distributional uncertainty set* or *ambiguity set* and denoted by $\mathcal{P}$. The expected cost is then minimized over the worst-case probability distribution from this set. This is called *Distributionally Robust Optimization* (DRO).

Usually, the outcomes of the random variables $\boldsymbol{\xi}_t$ are fixed to a finite number of realizations observed in the historical data. The ambiguity set $\mathcal{P}_t$ then models a variety of potential probability measures $\mathbb{P}_t \in \mathcal{P}_t$ supported on this finite set Ξ_t.

In the following, we restrict to DRO specifically in the SDDP context. For a general introduction to DRO, we refer to the review [176] and the tutorial [199]. We assume all assumptions from Sect. 3 to hold, except for Assumption 3. Furthermore, we only consider uncertainty in the RHS.

Then, the distributionally robust version of (MSLP) can be written as

$$\begin{aligned}
\min_{x_1,\boldsymbol{x_2},\ldots,\boldsymbol{x_T}} \max_{\mathbb{P}\in\mathcal{P}} \quad & \mathbb{E}\left[\sum_{t\in[T]} \big(\boldsymbol{c}_t(\xi_t)\big)^\top \boldsymbol{x}_t(\xi_{[t]})\right] \\
\text{s.t.} \quad & x_1 \in \mathcal{X}_1 \\
& \boldsymbol{x}_t \in \mathcal{X}_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}),\xi_t) \quad \forall \xi_t \in \Xi_t \ \forall t = 2,\ldots,T.
\end{aligned} \tag{81}$$

Remark 13.1. *Distributionally robust stochastic programming is closely related to risk-averse stochastic programming. In particular, the operator* $\max_{\mathbb{P}\in\mathcal{P}} \mathbb{E}[\cdot]$ *can be interpreted as a multi-period risk measure* $\mathcal{R}[\cdot]$*. This risk measure is coherent [199].*

For SDDP it is required to reformulate problem (81) by means of DPE. This requires that each distribution $\mathbb{P}$ in the ambiguity set $\mathcal{P}$ can be expressed as the cross product of the respective marginal distributions of random vectors $\boldsymbol{\xi}_t$ [199]. Formally,

$$\mathcal{P} := \big\{\mathbb{P} = \mathbb{P}_1 \times \ldots \times \mathbb{P}_T \ : \ \mathbb{P}_t \in \mathcal{P}_t, t \in [T]\big\}.$$

The ambiguity sets $\mathbb{P}_t$ are assumed to be independent of each other. This property is called *rectangularity* of $\mathcal{P}$ and is reminiscent of the stagewise independence assumption for vectors $\boldsymbol{\xi}_t$. Note that $\mathcal{P}_1$ is a singleton containing one distribution with one possible realization.

With the ambiguity sets $\mathcal{P}_t$, then the DPE can be written as

$$Q_{DR,t}(x_{t-1},\xi_t) := \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \mathcal{Q}_{DR,t+1}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1},\xi_t) \end{cases} \tag{82}$$

with

$$\mathcal{Q}_{DR,t+1}(x_t) := \max_{\mathbb{P}_{t+1}\in\mathcal{P}_{t+1}} \mathbb{E}_{\mathbb{P}_{t+1}}\left[Q_{DR,t+1}(x_t,\boldsymbol{\xi}_{t+1})\right], \tag{83}$$

and $\mathcal{Q}_{DR,T+1}(x_T) \equiv 0$. Compared to Sect. 3, here, an inner maximization problem is introduced when defining $\mathcal{Q}_{DR,t+1}(\cdot)$ to obtain the expected cost over the worst-case probability measure in $\mathcal{P}_{t+1}$. The first-stage problem reads

$$v_{DR}^* = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathcal{Q}_{DR,2}(x_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases} \tag{84}$$

How v_{DR}^* and a corresponding optimal policy can be computed algorithmically, heavily depends on the specific choice of the ambiguity sets $\mathcal{P}_t, t = 2,\ldots,T$. Various ambiguity sets are proposed in the literature. Usually, these sets are defined in such a way that they contain all distributions, which are *in some sense* within a given range of some nominal distribution. This nominal distribution, denoted by $\bar{\mathbb{P}}_t$, in turn, is defined by probabilities $\bar{p}_{tj} = \frac{1}{q_t}$ for all $j = 1,\ldots,q_t$, where q_t denotes the number of historical data samples. Based on the measure employed to evaluate the distance between two distributions or probability measures, respectively, different classes of ambiguity sets can be defined.

For SDDP, the following three distance measures have been used so far. In [107], the ℓ_∞ metric with parameter $r > 0$ is used to define the ambiguity set

$$\mathcal{P}_t = \left\{ \mathbb{P}_t \ : \ \sum_{i=1}^{q_t} p_{ti} = 1, \ p_{ti} \geq 0, \ \|p_t - \bar{p}_t\|_\infty \leq r \right\}. \tag{85}$$

A similar metric, but with the ℓ_2-norm, is used in [162] to define the ambiguity set

$$\mathcal{P}_t = \left\{ \mathbb{P}_t \ : \ \sum_{i=1}^{q_t} p_{ti} = 1, \ p_{ti} \geq 0, \ \|p_t - \bar{p}_t\|_2 \leq r \right\}. \tag{86}$$

This is a special case of the class of ϕ-divergence distances, see [12]. Both these distance measures are only applicable to discrete distributions supported on the observed historical data points.

On the contrary, the Wasserstein distance allows to compare general distributions (see for instance [217]). In our case with finite distributions $\mathbb{P}_t$ and $\bar{\mathbb{P}}_t$, the Wasserstein distance can be defined by the minimization problem

$$\begin{aligned} d_W(\bar{\mathbb{P}}_t, \mathbb{P}_t) := \min_z \quad & \sum_{i=1}^{q_t} \sum_{j=1}^{q_t} \|\xi_t^i - \xi_t^j\| z_{ij} \\ \text{s.t.} \quad & \sum_{j=1}^{q_t} z_{ij} = \bar{p}_{ti} \quad \forall i = 1, \dots, q_t \\ & \sum_{i=1}^{q_t} z_{ij} = p_{tj} \quad \forall j = 1, \dots, q_t \\ & z_{ij} \geq 0 \quad \forall i, j = 1, \dots, q_t, \end{aligned}$$

where for the norm different choices are possible. It can be interpreted as the amount of probability mass that has to be moved between the distributions. This distance is used in [65] to define the Wasserstein ambiguity set

$$\mathcal{P}_t = \left\{ \mathbb{P}_t \ : \ \sum_{i=1}^{q} p_{ti} = 1, \ p_{ti} \geq 0, \ d_W(\bar{\mathbb{P}}_t, \mathbb{P}_t) \leq r \right\}. \tag{87}$$

In all three cases, very different strategies are chosen to apply SDDP to the nested min-max structure defined by the DPE (82)-(84).

13.1.1 Reformulation as a Risk-averse Problem

As shown in [107], using the ℓ^∞-ambiguity set (85), the DPE (82)-(84) can be reformulated to those of a risk-averse multistage problem with nested conditional $\text{AVaR}_\alpha[\cdot]$, that is equations (60)-(62) with

$$\lambda_{t+1} = 1 - p_{t+1}^\ell, \quad \alpha_{t+1} = \frac{\lambda_{t+1}}{p_{t+1}^u - p_{t+1}^\ell},$$

where p_{t+1}^ℓ and p_{t+1}^u denote the probabilities associated with the probability measures at the lower and upper bound of ambiguity set (85). Therefore, SDDP can be applied as in this risk-averse setting.

13.1.2 Solving the Inner Maximization Problem Separately

Using the ℓ^2-ambiguity set (86) in the DPE (82)-(84) yields value functions, which can be proven to remain convex, and thus can be approximated by affine cuts [162].

To derive such cuts, Philpott et al. propose to solve the inner maximization problem identifying the worst-case distribution separately. In the backward pass, for some stage t, first the subproblems are solved for all $j = 1, \dots, q_t$ as usual. Then, using the obtained values of $\underline{Q}_t^i(x_{t-1}^{ik}, \xi_{tj})$, the inner maximization problem is solved. This can be done algorithmically and in some cases even analytically, as shown in [162]. The obtained worst-case probability measure $\mathbb{P}^*$ can then be used to compute subgradients and cut coefficients. Even though these coefficients are determined based on cut approximation $\mathfrak{Q}_t^{i+1}(\cdot)$ and on $\mathbb{P}^*$, which does not necessarily coincide with the worst-case probability measure in the true DPE, valid cuts are constructed and convergence is ensured [162].

13.1.3 Using a Dual Representation

If we use the Wasserstein ambiguity set (87) in SDDP, we obtain the inner maximization problem

$$\begin{aligned}
\max_{z_t, p_{t+1}} \quad & \sum_{j=1}^{q_t} p_{t+1,j} Q_{t+1}(x_t, \xi_{t+1,j}) \\
\text{s.t.} \quad & \sum_{i=1}^{q_t} \sum_{j=1}^{q_t} d_{t+1,ij} z_{tij} \le 1 \\
& \sum_{j=1}^{q_t} z_{tij} = \bar{p}_{ti} \quad \forall i = 1, \dots, q_t \\
& \sum_{i=1}^{q_t} z_{tij} = p_{tj} \quad \forall j = 1, \dots, q_t \\
& z_{tij} \ge 0 \quad \forall i, j = 1, \dots, q_t
\end{aligned}$$

with $d_{t+1,ij} = \|\xi_{t+1}^i - \xi_{t+1}^j\|$. Duque and Morton [65] suggest to replace this problem using its dual problem. This way, the value functions can be evaluated by solving the single-level minimization problem

$$Q_{DR,t}(x_{t-1}, \xi_t) :=$$
$$\begin{cases}
\min\limits_{x_t, \gamma_t, \nu_t} & c_t^\top x_t + r\gamma_t + \sum\limits_{i=1}^{q_{t+1}} q_{t+1}^i \nu_t^i \\
\text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \\
& d_{t+1,ij}\gamma_t + \nu_{ti} \ge Q_{DR,t+1}(x_t, \xi_{t+1,j}) \quad \forall i, j = 1, \dots, q_{t+1} \\
& \gamma_t \ge 0
\end{cases}$$

with dual variables γ_t and ν_t.

As proven in [65], these value functions are piecewise linear and convex on $\mathcal{X}_{t-1}$, and therefore can be represented by finitely many linear cuts. However, this approach requires to use multi-cut SDDP, see Sect. 21.2.1, because otherwise bilinear terms occur.

With all these strategies, the forward pass remains basically the same as in standard SDDP. The sampling can be done from the nominal distribution associated with $\bar{\mathbb{P}}_t, t = 2, \dots, T$, or alternatively the current worst-case distribution associated with $\mathbb{P}_t^*$ [65]. If independent sampling is conducted, convergence follows as for standard SDDP. However, challenges to determine valid upper bounds are prevalent for distributionally robust SDDP similarly to the risk-averse case.

Computational results indicate that taking the dual reformulation approach, better approximations are achieved for multi-cut SDDP than solving the inner maximization in a side computation [65]. Furthermore, out-of-sample tests by Philpott et al. [162] imply that distributionally robust SDDP yields policies which are better suited, *e.g.*, induce lower costs, in periods with a substantial risk of high costs.

13.2 Partially Observable Distributions

A different approach to deal with distributional uncertainty is introduced by Dowson et al. in [59], and is referred to as *partially observable* multistage stochastic programming. The idea is to consider a finite number of potential distributions by combining problem (MSLP) with a hidden Markov model. More precisely, in each stage $t \in [T]$, different nodes can be reached, with each node representing one Markov state. Let $\mathcal{N}$ denote the set of all these nodes except for the root node. Each node reflects a different candidate distribution, possibly with identical realizations $\xi_j, j = 1, \ldots, q$, but different associated probabilities.

As a key idea, consider a partition $\mathcal{A}$ of nodes in $\mathcal{N}$ into ambiguity sets $A \in \mathcal{A}$, satisfying $\bigcup_{A \in \mathcal{A}} A = \mathcal{N}$. For example, this partition can be chosen such that there is one ambiguity set A for each stage.

To model the distributional uncertainty, it is now assumed that at any point, only the current ambiguity set is known, while the specific node within it cannot be observed. However, for each node i, a probability b_i is available. In other words, each candidate distribution is considered to be the most accurate representation of the true underlying distribution with a certain probability. These probabilities are stored in a so called *belief state* b. Each time an ambiguity set A is entered and a particular realization $\widetilde{\xi}$ of the random data is observed, the belief state is updated componentwise by applying Bayes' theorem [59].

In contrast to (MSLP) with perfect distribution information (see Assumption 3), the value functions $Q_t(\cdot)$ have to incorporate this belief state. To this end, let $p_{i\ell}$ be the probability of observing $\xi_{i\ell}$ conditional on being in node i with $\ell = 1, \ldots, q^i$. Let $\bar{\mathcal{N}}$ describe all nodes including the root node, ω_{jk} the transition probability from node j to k and $B_k(b, \xi)$ the update rule for the belief state being in (unobservable) node k. Furthermore, let x' denote the current trial solution. Then, the expected value function can be written as

$$\mathcal{Q}_B(x', b) := \sum_{j \in \bar{\mathcal{N}}} b_j \sum_{k \in \mathcal{N}} \omega_{jk} \sum_{\ell=1}^{q^k} p_{k\ell} \; Q_k\big(x', B_k(b, \xi_{k\ell}), \xi_{k\ell}\big). \tag{88}$$

This means that the value functions $Q_k(\cdot, \cdot)$ depend on a node and an updated belief state, and in (88) it is looped over all nodes, weighing the corresponding expected value with the current belief and the transition probabilities between the nodes.

As proven in Theorem 1 in [59], the expected value functions $\mathcal{Q}_B(\cdot)$ are saddle functions, as they are convex in x for fixed b, but concave in b for fixed x. Therefore, to apply SDDP, the cut generation has to be adapted to this property. This can be achieved by using an outer approximation for x and an inner approximation for b [59]. The main difference for the cut computation is that apart from taking expectations over the realizations of ξ, it is looped over all nodes in the current ambiguity set A and the cut components are weighed with the current belief [59].

In the forward pass, for each stage $t = 2, \dots, T$, first, a new node is sampled conditionally on the (unobserved) current node. Then, a realization of ξ is sampled conditionally on the obtained node and the associated candidate distribution. For a more detailed description, see [59].

A different method of combining SDDP with a hidden Markov model is given in [66]. One general drawback of such hidden Markov approaches is that transition probabilities between the nodes have to be properly defined a priori.

14 Stagewise Dependent Uncertainty [relaxing Assumption 2]

As explained in Sect. 2 and 3, stagewise independence (Assumption 2) is a standard assumption in dynamic programming, and thus also for SDDP. It is also crucial for the computational tractability of SDDP compared to NBD because it ensures that there exists only one expected value function $\mathcal{Q}_t(\cdot)$ per stage and that cuts can be shared between scenarios, see Sect. 5.2. However, in many applications, the uncertain data in (MSLP) (*e.g.*, demand, fuel prices, electricity prices, inflows) shows correlations over time and assuming stagewise independence is not appropriate.

If the uncertainty in problem (MSLP) is stagewise dependent, the expected value functions $\mathcal{Q}_t(\cdot)$ for $t = 2, \dots, T$ do not only depend on x_{t-1}, but implicitly also depend on the history $\xi_{[t-1]}$ of the process $(\boldsymbol{\xi}_t)_{t\in[T]}$. In order to apply SDDP, this dependence has to be taken into account, for instance by reformulating the model or adapting the algorithmic steps in SDDP. In this section, we consider different cases of stagewise dependent uncertainty and ways of how SDDP can be applied in these cases.

14.1 Expanding the State Space

As a first case of stagewise dependent uncertainty, let us assume that the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is a simple linear *autoregressive* (AR) process with lag one, defined by appropriately chosen coefficient vectors γ_t, matrices Φ_t and stagewise independent and i.i.d. error terms η_t:

$$\boldsymbol{\xi}_t = \gamma_t + \Phi_t \boldsymbol{\xi}_{t-1} + \boldsymbol{\eta}_t. \tag{89}$$

Remark 14.1. *If we still assume finite randomness (Assumption 5), now for $\boldsymbol{\eta}_t$, then $\boldsymbol{\xi}_t$ can be modeled by a classical scenario tree, see Sect. 5.2.*

The most natural approach to deal with this case, is to reformulate (MSLP) in such a way that it exhibits stagewise independent uncertainty [154]. This can be achieved by including ξ_{t-1} as an additional state variable. Then, as shown in [129],

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[Q_t(x_{t-1}, \boldsymbol{\xi}_t)\right] &= \mathbb{E}_{\boldsymbol{\eta}_t|\xi_{t-1}}\left[Q_t(x_{t-1}, \gamma_t + \Phi_t\xi_{t-1} + \boldsymbol{\eta}_t)\right] \\ &= \mathbb{E}_{\boldsymbol{\eta}_t}\left[Q_t(x_{t-1}, \gamma_t + \Phi_t\xi_{t-1} + \boldsymbol{\eta}_t)\right],\end{aligned}$$

where the second equality holds because η_t and ξ_{t-1} are statistically independent.

By introducing equation (89) as a constraint and defining a new value function

$$\widehat{Q}_t(x_{t-1}, \xi_{t-1}, \eta_t) := Q_t(x_{t-1}, \gamma_t + \Phi_t\xi_{t-1} + \eta_t), \tag{90}$$

and the corresponding expected value function

$$\widehat{\mathcal{Q}}_t(x_{t-1}, \xi_{t-1}) := \mathbb{E}_{\boldsymbol{\eta}_t}\left[\widehat{Q}_t(x_{t-1}, \xi_{t-1}, \boldsymbol{\eta}_t)\right] \tag{91}$$

for all $t = 2, \dots, T$, it follows

$$\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[Q_t(x_{t-1}, \boldsymbol{\xi}_t)\right] = \widehat{\mathcal{Q}}_t(x_{t-1}, \xi_{t-1}).$$

The state variables then consist of the resource state x_{t-1} and the information state ξ_{t-1}, while the stagewise independent uncertainty is modeled by $\boldsymbol{\eta}_t$. Importantly, ξ_t is regarded as a decision variable in the reformulated problem, augmenting the dimension of the decision space.

Remark 14.2. *It is worth emphasizing that this approach is presented in various different ways in the literature. In some cases, as outlined, equation* (89) *is explicitly incorporated into the DPE as an additional constraint [174, 204]. In some cases, each occurrence of ξ_t in the subproblems is simply replaced by the RHS of* (89)*. And in other cases, the dependence on ξ_{t-1} is only expressed by writing $\widehat{Q}_t(\cdot,\cdot,\cdot)$ and $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ as functions of ξ_{t-1}, whereas the explicit relation* (89) *is only considered in the cut generation process [84, 129, 179]. We revisit this observation in the next subsection.*

By the presented procedure, stagewise independence (Assumption 2) is recovered for (MSLP). However, in order to apply SDDP, it also has to be ensured that valid linear cuts for $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ can be derived as functions in both types of state variables. This requires that $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ is convex in both x_{t-1} and ξ_{t-1}. Similarly to Theorem 2.8, it can be shown that under certain assumptions, this property is satisfied.

Theorem 14.3 ([179])**.** *Let $\boldsymbol{\xi}_t$ be described by* (89) *and let ξ_{t-1} be contained in some convex set. Then, under Assumptions 1 and 3 to 9, the expected value function $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ is piecewise linear and*

a) convex in x_{t-1} on $\mathcal{X}_{t-1}$ for fixed ξ_{t-1},

b) convex in $\xi_{t-1} = (T_{t-2}, h_{t-1})$ for fixed $x_{t-1}, W_{t-1}, c_{t-1}$,

c) concave in $\xi_{t-1} = c_{t-1}$ for fixed $x_{t-1}, W_{t-1}, T_{t-2}, h_{t-1}$,

d) convex jointly in x_{t-1} and in $\xi_{t-1} = h_{t-1}$ for fixed $W_{t-1}, T_{t-2}, c_{t-1}$.

Theorem 14.3 shows that convexity in both types of state variables is only guaranteed if the stagewise dependent part of the uncertainty only enters the RHS $\boldsymbol{h}_t(\xi_t)$ of problem (MSLP). Note that this still allows for additional stagewise independent uncertainty in c_t, W_t and T_{t-1}. The result also requires linearity of (MSLP) (Assumption 6) and of the AR process (89) defining the random variable $\boldsymbol{\xi}_t$.

Under certain assumptions, Theorem 14.3 can be generalized to convex problems (MSLP) and stagewise dependence in the RHS defined by a convex function [84]. Moreover, the result is not limited to lag-one processes, but can be enhanced to AR processes with higher lag order [84]. This is important for practical applications, as often several lags are required to explain a time series appropriately. In contrast, for general nonlinear stochastic processes or for uncertainty in W_t, c_t or T_{t-1}, such a generalization seems not possible. In order to cover such cases, different approaches are required. We discuss those in later parts of this section.

For simplicity, assume that $X_t = \{x_t \in \mathbb{R}^{n_t} \ : \ x_t \geq 0\}$ for all $t \in [T]$ and recall the definition of the approximate subproblem (10):

$$\underline{Q}_t(x_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t, \theta_{t+1}} & \big(c_t(\xi_t)\big)^\top x_t + \theta_{t+1} \\ \text{s.t.} & W_t(\xi_t)x_t = h_t(\xi_t) - T_{t-1}(\xi_t)x_{t-1} \\ & x_t \geq 0 \\ & -(\beta^r_{t+1})^\top x_t + \theta_{t+1} \geq \alpha^r_{t+1}, \quad \forall r \in \Gamma_{t+1}, \end{cases} \tag{92}$$

where Γ_{t+1} is the index set of previously generated cuts. Then, the result in Theorem 14.3 can be illustrated by means of the feasible region of the LP dual to (92), which can be written as

$$\begin{aligned} \max_{\pi_t, \rho_t} \quad & \big(h_t(\xi_t) - T_{t-1}(\xi_t)x_{t-1}\big)^\top \pi_t + a_{t+1}^\top \rho_t \\ \text{s.t.} \quad & \big(W_t(\xi_t)\big)^\top \pi_t - B_{t+1}^\top \rho_t \leq c_t(\xi_t) \\ & e^\top \rho_t = 1 \\ & \rho_t \geq 0. \end{aligned} \tag{93}$$

Here, we collect all cut gradients β^r_{t+1} in a matrix B_{t+1} and all cut intercepts α^r_{t+1} in a vector a_t for compact representation. π_t denotes the dual variable to the original constraints, and ρ_t denotes the dual variable to the previously generated cuts.

In the case of linear AR processes in the RHS $\boldsymbol{h}_t(\xi_t)$, the dual feasible region is not affected by the new state variable ξ_{t-1} (and also remains polyhedral). This means that the extreme solutions obtained for one state $\bar{\xi}_{t-1}$ remain valid, although not necessarily optimal, for all other states ξ_{t-1} as well. In contrast, in other cases of stagewise dependence, the dual feasible region and its extreme solutions may change for different states, affecting the properties of $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ [179].

In sum, for affine and convex AR processes occurring in the RHS, expanding the state recovers stagewise independence (Assumption 2), but at the same time convexity of $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ in all state variables is preserved. Therefore, SDDP can be used as introduced in Sect. 3. In this case, the obtained cuts are functions of both state variables and can be formulated with a cut gradient for each of them (compare to (17)), *i.e.*,

$$\phi_t(x_{t-1}, \xi_{t-1}) = \alpha_t + \big(\beta^x_t\big)^\top x_{t-1} + \big(\beta^\xi_t\big)^\top \xi_{t-1}.$$

Unfortunately, depending on the dimension κ_{t-1} of ξ_{t-1}, the state space dimension can increase significantly. This effect is amplified for higher lag orders. As the computational complexity of SDDP grows exponentially in this dimension, see Sect. 4.2, augmenting the state space is detrimental and should be avoided if possible.

14.2 Scenario-Adaptable Cut Formulas

The previously described adverse effect can be alleviated to some degree by a special cut generation approach that was first proposed by Infanger and Morton [109] and later enhanced by de Queiroz and Morton [174] and Guigues [84]. In all these cases, the process model, such as (89), is not explicitly incorporated into the subproblems, see Remark 14.2. Instead, it is merely considered within the cut generation process. The main idea is to derive scenario-adaptable closed-form cut formulas, given AR processes with a specific structure, which allow one to adapt the cut generated for one specific history $\bar{\xi}_{[t-1]}$ to different histories $\xi_{[t-1]}$ of the stochastic process, and thus to different scenarios. This way, the cuts can be *shared* between scenarios (see Sect. 5.2) without the need to incorporate (89) into (MSLP) as a constraint. Importantly, these cut formulas lead to the exact same cuts as the previously described approach.

To illustrate this idea, consider a cut derived using dual problem (93) without paying any particular attention to the stagewise dependence. For convenience, but without loss of generality, we assume T_{t-1} to be deterministic and the RHS uncertainty to be defined by

$$\boldsymbol{h}_t(\xi_t) = \Phi_t h_{t-1}(\xi_{t-1}) + \boldsymbol{\eta}_t \tag{94}$$

with stagewise independent error terms $\boldsymbol{\eta}_t$, similarly to (89). We obtain

$$\begin{aligned} \widehat{\mathcal{Q}}_t(x_{t-1}, \xi_{t-1}) &\geq \mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[-\boldsymbol{\pi}_t^\top T_{t-1}x_{t-1} + \boldsymbol{\pi}_t^\top \boldsymbol{h}_t(\xi_t) + \boldsymbol{\rho}_t^\top \boldsymbol{a}_{t+1}\right] \\ &= \mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[-\boldsymbol{\pi}_t^\top T_{t-1}\right] x_{t-1} + \mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[\boldsymbol{\pi}_t^\top \boldsymbol{h}_t(\xi_t) + \boldsymbol{\rho}_t^\top \boldsymbol{a}_{t+1}\right] \end{aligned} \tag{95}$$

We can make the following observations:

(i) Since the probabilities in $\mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}[\cdot]$ are assumed to not depend on ξ_{t-1} (recall that $\boldsymbol{\eta}_t$ is stagewise independent) and since all scenarios share the same dual feasible region, the cut gradient

$$\beta_t = \mathbb{E}_{\boldsymbol{\xi}_t|\xi_{t-1}}\left[-\boldsymbol{\pi}_t^\top T_{t-1}\right] \tag{96}$$

derived for one specific scenario $\bar{\xi}_{t-1}$, is valid for all other scenarios as well.

(ii) According to (94), the RHS $\boldsymbol{h}_t(\xi_t)$ depends on ξ_{t-1}. Therefore, to evaluate the cut for a specific scenario, this term has to be adapted to this scenario. Otherwise, the cut may become invalid. By (94), this term can be split up into a scenario-dependent part depending on ξ_{t-1} and a scenario-independent part depending on η_t only.

(iii) The last term $\boldsymbol{a}_{t+1}$ in (95) is the cut intercept of the following stage. As we face stagewise dependence, this intercept is not scenario-independent anymore, but should denote $\boldsymbol{a}_{t+1}(\xi_t)$. Moreover, it is defined recursively: The stage-t intercept includes the stage-$(t+1)$ intercept, which includes the stage-$(t+2)$ intercept and so on. This implies that to evaluate $\boldsymbol{a}_{t+1}(\xi_t)$ for a specific scenario, it is basically required to recursively traverse the whole scenario tree starting form stage t. This is computationally intractable.

To address these observations, the main idea by Infanger and Morton [109] is to express the cut intercept $\alpha_t(\xi_{t-1})$ as the sum of a stagewise independent term α_t^{ind} and a stagewise dependent term $\alpha_t^{\text{dep}}(\xi_{t-1})$:

$$\alpha_t(\xi_{t-1}) = \alpha_t^{\text{ind}} + \alpha_t^{\text{dep}}(\xi_{t-1}). \tag{97}$$

Let $\bar{\pi}_t = \mathbb{E}_{\boldsymbol{\eta}_t}[\boldsymbol{\pi}_t]$ and $\bar{\rho}_{tt} = \mathbb{E}_{\boldsymbol{\eta}_t}[\boldsymbol{\rho}_t]$ denote the expected value of the dual variables obtained for realizations of $\boldsymbol{\eta}_t$. As explained, these dual values are valid for any history of the stochastic process due to the structure of the dual feasible set. Let $\bar{\mathscr{P}}_t$ define the $(|\Gamma_t| \times m_t)$-matrix containing the values of $\bar{\pi}_t$ and $\bar{\mathscr{R}}_t$ the $(|\Gamma_t| \times |\Gamma_{t-1}|)$-matrix containing the values of $\bar{\rho}_t$ for the previously determined cuts. Furthermore, let the matrix D_t be defined recursively by

$$D_t = \left[\bar{\mathscr{P}}_{t+1} + \bar{\mathscr{R}}_{t+1} D_{t+1}\right] \Phi_t, \quad D_T = 0. \tag{98}$$

Then, as shown in [109], the stagewise dependent cut intercept is given by

$$\alpha_t^{\text{dep}}(\xi_{t-1}) = [\bar{\pi}_t + \bar{\rho}_t D_t]\, \Phi_t h_{t-1}(\xi_{t-1}). \tag{99}$$

This means that a cut can be constructed by using formula (96) for the gradient and formulas (97), (98) and (99) for the intercept. The stagewise independent term can be either determined by an additional formula or by subtracting (99) from $\alpha_t(\xi_{t-1})$ [109]. In order for a cut to be shared with a different scenario at stage $t-1$, it is only required to adapt the stagewise dependent intercept (99) to this specific scenario. In other words, a given cut can be *corrected* to be valid for a different history of the stochastic process. In particular, it is not required to add (94) as a constraint to the stage-t subproblem or to traverse the whole scenario tree (see Remark 14.1). Instead, only the cut gradient, the stagewise independent part of the intercept and the *cumulative expected dual vector* $\left[\bar{\mathscr{P}}_{t+1} + \bar{\mathscr{R}}_{t+1} D_{t+1}\right] \Phi_t$ have to be stored [109].

Whereas we limited our explanations to a very simple AR process so far, similar cut formulas can be derived for more complex processes [84, 109, 174, 179]. We give an overview on different cases covered in the literature in Table 6. Some of the process formulas in Table 6 are presented in a simplified form for reasons of clarity, *e.g.*, by omitting standardization and the incorporation of seasonal or periodical effects. For example, this is true for the SPAR processes considered in [127] (also see Sect. 9), where spatial dependencies between locations i and i' are taken into account.

Importantly, all processes for which scenario-adaptable closed-form cut formulas can be derived require a specific structure, such as linearity, convexity or separability. As shown by Guigues [84], a generalization to convex AR processes and more complex structures in the RHS is possible. For instance, the RHS h_t does not have to be directly described by the stochastic process (constant $\boldsymbol{h}_t \equiv \boldsymbol{\xi}_t$), but may also be defined as some function $\boldsymbol{h}_t(\cdot)$ of $\boldsymbol{\xi}_t$. Moreover, for the affine case, alternative formulas to the ones provided by Infanger and Morton are presented by Guigues [84]. The main difference is that only a minimal subset of coefficients is used, due to defining the process $(\boldsymbol{\xi}_t)_{t\in[T]}$ componentwise and not in vectorial form compared to (89) or (94). On the other hand, no recursive formula as in (98) is provided to compute the cut coefficients. Finally, Guigues shows that also for feasibility cuts (Sect. 17) scenario-adaptable cut formulas can be derived.

It is important to emphasize that the presented approach only partially mitigates the drawbacks of augmenting the state space. First of all, the history of the stochastic process has to be stored to compute ξ_t, even if such computation is possible outside of the subproblems. Guigues provides a detailed discussion on how state vectors of minimal size can be defined in order to keep the stored information as small as possible [84]. Additionally, due to their dependence on ξ_{t-1}, or $\xi_{[t-1]}$ in general, the expected value functions $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ live in a higher-dimensional space. Therefore, more iterations and cuts may be required to achieve convergence compared to the stagewise independent case, as discussed in Sect. 4.2.

14.3 Sensitivity of SDDP with AR Processes

Let the uncertainty in (MSLP) be modeled by an AR process. Consider the approach of expanding the state, leading to two types of state variables: x_t and $\xi_{[t]}$. Both contain information on future resource availability (*e.g.*, hydro storage volume and hydro inflow history affecting future inflows), but they differ in several aspects [208]. First, whereas

	Autoregressive model for ξ_t				
RHS $h_t(\xi_t)$	Model	Type	Lag	Formula	Source
const.	AR	L	1	$\xi_t = \Phi_t \xi_{t-1} + \eta_t$	[109]
L	AR	L	1	$\xi_t = \Phi_t \xi_{t-1} + \eta_t$	[174]
const.	PAR	L	1	$\xi_t = \varphi_t(\xi_{t-1} - \mu_{t-1}) + \mu_t + \sigma_t \eta_t$	[211]
const.	AR	L	≥ 1	$\xi_t = \sum_{k=1}^{t-1} (\Phi_k^t \xi_k + \Psi_k^t \eta_k) + \eta_t$	[109]
L/C*	AR	L	≥ 1	$\xi_t = \Phi_t \xi_{[t-1]} + \eta_t$	[84]
L/C*	AR	L	≥ 1	$\xi_t = \Phi_t \xi_{[t-1]} + \Psi_t \eta_t + \Theta_t$	[84]
const.	SPAR	L	≥ 1	$\xi_{ti} = \sum_{i'} \sum_{k=1}^{t-1} \Phi_{ii'k}^t \xi_{ti'} + \eta_{ti}$	[127]
const.	AR	NL	1	$\xi_t = \Phi_t(f_t(v_{t-1}) + \xi_{t-1}) + \eta_t$	[109]
const.	AR	NL	≥ 1	$\xi_t = \sum_{k=1}^{t-1} (\Phi_k^t \xi_k + f_k^t(v_k)) + \eta_t$	[109]
C	AR	C	≥ 1	$\xi_t = f_t(\xi_{[t-1]}, \eta_t)$	[84]

L = affine/linear function, C = convex function, NL = general nonlinear function
* only in case of inequality constraints

Table 6: RHS and uncertainty models considered in the literature on SDDP with stage-wise dependence to derive scenario-adaptable closed-form cut formulas.

the information provided by the state x_{t-1} is certain, the information provided by $\xi_{[t-1]}$ enters an AR model predicting future realizations, which still involves uncertainty. Second, the parameters of this AR model are estimated from data, and thus can be subject to estimation errors. Third, in practice it can often be observed that the values in $(\xi_t)_{t \in [T]}$ show higher variability over short time than the values of $(x_t)_{t \in [T]}$. This uncertainty and variability raises the question on how much the solutions obtained in SDDP react to changes in $\xi_{[t-1]}$. This can be examined in a *sensitivity analysis*.

A general approach for sensitivity analysis in SDDP is presented in [97] and applied to an inventory problem with AR demand. Also the sensitivity with respect to AR model parameters Φ_t or γ_t is discussed.

For a hydrothermal problem, in [208], it is shown that the solutions obtained in SDDP are more sensitive to changes in the initial information state ξ_1 than to changes in the initial resource state x_0. Based on the previous observations this leads to the unfavorable side effect of expanding the state space that solutions of SDDP exhibit larger variability. This may have severe consequences in economic applications, such as increasing risk, unpredictability of prices or distorted investment signals.

To address this issue, Soares et al. present different mitigation heuristics [208], such as regularizing changes in x_t over time, or using the accurate AR model in the forward pass of SDDP, but predefined unconditional samples in the backward pass in order to avoid the dependence of cuts on $\xi_{[t-1]}$. While they report positive computational results, the authors provide no theoretical results on reasonable parameter choice, cut validity and convergence for their heuristics.

14.4 Markov Chain SDDP

Assume that the data process $(\boldsymbol{\xi}_t)_{t \in [T]}$ is Markovian, *i.e.*, as in (89), $\boldsymbol{\xi}_t$ only depends on ξ_{t-1} for all $t = 2, \dots, T$ instead of the whole history $\xi_{[t-1]}$. Then, instead of expanding the state space also an alternative approach can be used to apply SDDP.

In this case, the data process can be represented, or at least approximated (if the random variables $\boldsymbol{\xi}_t$ are continuous), by a discrete Markov chain. This approximation can be obtained by lattice quantization techniques [29, 129]. As it contains only finitely many states per stage $t = 2, \ldots, T$, this Markov chain can be illustrated as a recombining scenario tree or *scenario lattice* [129], just as in the case of stagewise independence Assumption 2, see Sect. 2. The difference is that in the Markov chain case the probabilities of transitions to stage-t nodes may differ between different stage-$(t-1)$ nodes. This also includes the possibility that some stage-t nodes may not be reached from certain stage-$(t-1)$ nodes.

Due to this difference, the (expected) value functions $\mathcal{Q}_{t\ell}(\cdot)$ depend on the states $\zeta_\ell, \ell = 1, \ldots, L$, of the Markov chain. In other words, for each such state (*i.e.*, each node in the recombining tree), a different expected value function and a different set of value functions exist. In SDDP, then cuts are derived for each of these functions separately. This idea is called Markov chain SDDP (MC-SDDP) [129] or *approximate dual dynamic programming* (ADDP) [130, 131], whereas for distinction the approach of expanding the state space is referred to as time series SDDP (TS-SDDP).

For problems with moderate state space dimension, expanding the state may be computationally favorable as only one expected value function has to be approximated per stage. On the other hand, a computational advantage of MC-SDDP is that the computational effort grows linearly with the number of Markov states only [201]. In contrast, expanding the state leads to a state space dimension increase in which the complexity of SDDP grows exponentially. Moreover, MC-SDDP requires no linearity and is not limited to stagewise dependent uncertainty only appearing in the RHS of (MSLP). As long as the Markov property is satisfied, it allows for stagewise dependent uncertainty in all data c_t, T_{t-1}, W_t and h_t of (MSLP).

The main drawback of MC-SDDP lies in the relation to the true problem $(\widetilde{P})$ in case of a continuous data process $(\boldsymbol{\xi}_t)_{t \in [T]}$, see also Sect. 11. For SDDP with AR processes and expanding the state space, many results exist that allow for inference of the SAA solution with respect to the true problem, see Sect. 11. One key property in this regard is that ξ_{t-1} is treated as a possibly continuous state variable in SDDP, such that the derived cuts are also valid at states which are not reached by the scenarios $\xi^s \in \mathcal{S}$ that are considered in SDDP. Similar results are not available for MC-SDDP. In particular, the obtained policy and lower bounds are not necessarily valid for the true problem [129].

In spite of this theoretical downside, Löhndorf and Shapiro report tighter lower bounds and better policies even for the true process based on computational experiments [129]. They conjecture that this is due to a differing exploration of the state space. Expanding the state space introduces additional state variables, which are not under control of the optimal policy (their trajectory is not chosen based on solving the approximate subproblems in the forward pass, but selected randomly in the forward pass). This may lead to selection of states, which do not provide the highest information gain. With MC-SDDP this is partially mitigated by choosing sufficiently different states in advance when constructing the Markov chain.

14.5 SDDP with Integrated Markov Chain

By Theorem 14.3, a natural extension of SDDP to stagewise dependent uncertainty using expanding the state space is only possible for linear (or at least convex) AR processes appearing in the RHS of problem (MSLP). In all other cases, expanding the state space destroys the convexity of the expected value functions $\widehat{\mathcal{Q}}_t(\cdot, \cdot)$. Therefore, in such cases, different approaches are required. One such approach is to integrate a discrete Markov chain into the uncertainty modeling. This approach is quite established in the literature on and in practical application of SDDP. Importantly, this approach does not necessarily coincide with the previous case where the process $(\boldsymbol{\xi}_t)_{t\in[T]}$ itself is assumed to be Markovian and approximated by a Markov chain. Instead, the process is not assumed Markovian, but its realizations ξ_t are assumed to depend on the state of an underlying Markov chain.

Modeling. Consider a Markov chain with finitely many possible states $\zeta_\ell, \ell = 1,\dots,L$, with $L \in \mathbb{N}$. At each stage $t \in [T]$, we denote the current state of the Markov chain as ψ_t (again, we assume that ψ_1 is deterministic). The transition probabilities between state $\psi_{t-1} = \zeta_\ell$ at stage $t-1$ and $\psi_t = \zeta_{\ell'}$ at stage t are then denoted by $\omega_{\ell\ell'}$ for $\ell, \ell' \in \{1,\dots,L\}$. For simplicity, we assume the Markov chain to be time-homogeneous, such that $\omega_{\ell\ell'}$ does not depend on t, even though this is not required.

We now assume that the distribution of random variable $\boldsymbol{\xi}_t$ at stage $t \in [T]$ may depend on the state ψ_t of the Markov chain. In other words, for each possible state $\zeta_\ell, \ell = 1,\dots,L$, the distribution of $\boldsymbol{\xi}_t$ may differ. We emphasize this by writing $\boldsymbol{\xi}_t^\ell$.

The value functions $Q_t(\cdot,\cdot)$ for (MSLP) then do not only depend on x_{t-1} and the realization ξ_t of $\boldsymbol{\xi}_t$, but also on the current Markov state ψ_t. As this state can only take finitely many values, we denote this by $Q_{t\ell}(x_{t-1},\xi_t)$, where index ℓ indicates conditioning on $\psi_t = \zeta_\ell$. Based on this definition, the expected value functions can be expressed as

$$\mathcal{Q}_{t\ell}(x_{t-1}) := \sum_{\ell'=1}^{L} \omega_{\ell\ell'} \mathbb{E}_{\boldsymbol{\xi}_t|\ell'} \left[Q_{t\ell'}(x_{t-1}, \boldsymbol{\xi}_t^{\ell'}) \right]. \tag{100}$$

The index ℓ of the expected value function refers to the previous Markov state $\psi_{t-1} = \zeta_\ell$. Compared to standard SDDP, the expectation is not only taken over the realizations of $\boldsymbol{\xi}_t^{\ell'}$, but also the state transitions from ψ_{t-1} to ψ_t are taken into account. Using this definition, the DPE for stages $t = 2,\dots,T$ can be written as

$$Q_{t\ell}(x_{t-1},\xi_t^\ell) := \begin{cases} \min\limits_{x_t} & \zeta_\ell^\top x_t + \mathcal{Q}_{t+1,\ell}(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(\xi_t^\ell). \end{cases} \tag{101}$$

Note that the dependence on ζ_ℓ in (100) resembles the expanding-the-state approach from Sect. 14.1. However, there are important differences. ψ_{t-1} does not enter the subproblems and it can only take a finite number of different values, whereas $\xi_{[t-1]}$, even if discrete, is treated like a continuous state variable when expanding the state. Furthermore, as the transition probabilities $\omega_{\ell\ell'}$ may differ for each ζ_ℓ, the cut components are weighted differently and cuts cannot be shared between different Markov states. Consequently, it is required to store separate expected value functions $\mathcal{Q}_{t\ell}(\cdot)$ for each $\ell = 1,\dots,L$. In return, the non-convexity of these functions is circumvented, since each $\mathcal{Q}_{t\ell}(\cdot)$ remains convex and is approximated on its own, see also the discussion in Sect. 14.4.

As an example, consider a problem with $L = 2$ Markov states and $q^\ell = 2$ realizations for $\boldsymbol{\xi}_t^\ell$ for each of them, which is borrowed from [160]. The corresponding scenario tree with underlying Markov chain is illustrated in Figure 13. For the transition probabilities let $\omega_{11} = q, \omega_{12} = 1-q, \omega_{21} = 1-p$ and $\omega_{22} = p$. For all t and $\ell \in \{1,2\}$, the distribution of ξ_t^ℓ is given by $p_{tj} = \frac{1}{2}$ for $j \in \{1,2\}$.

As an alternative to the scenario tree in Figure 13, the stochastic process with underlying Markov chain can be represented by a Markovian policy graph with finitely

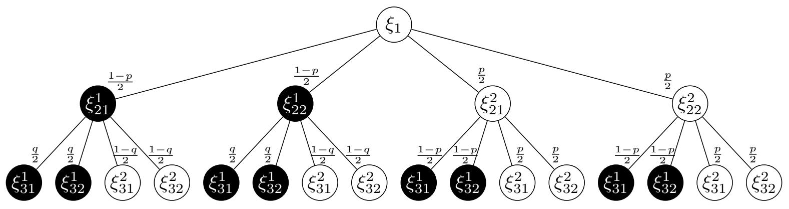


Figure 13: Scenario tree with underlying Markov chain (state 1 printed in black, state 2 printed in white). Replication from [160].

many nodes per stage [56]. This approach is also included in the software package `SDDP.jl`, see Sect. 10.

SDDP. Let us now address how SDDP works in this case. In the forward pass, different approaches are used in the literature. The most natural one is for each stage t and each sample path $k \in \mathcal{K}$, to sample first from the Markov states and then conditionally from $\boldsymbol{\xi}_t^\ell$ [161]. Sometimes it is also proposed to use historical values here, *e.g.*, true inflow spot-price combinations [79]. In such a case, it is possible that a spot price is drawn which is not a valid state of the Markov chain. Then, a strategy is to use the *in some sense* closest state from the Markov chain [79]. Another one is to use a linear interpolation between the hyperplanes of neighbouring states [81, 229].

For stages $t = 2, \dots, T$, states $\ell = 1, \dots, L$ and samples $k \in \mathcal{K}$, based on (101), the approximate subproblems solved in the forward pass of SDDP have the form

$$\underline{Q}_{t\ell}^i(x_{t-1}^{ik}, \xi_t^{\ell k}) := \begin{cases} \min\limits_{x_t} & \left(c_t(\xi_t^\ell)\right)^\top x_t + \mathfrak{Q}_{t+1\ell}^i(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^{ik}, \xi_t^\ell). \end{cases} \tag{102}$$

Importantly, each function $\mathcal{Q}_{t\ell}(\cdot), \ell = 1, \dots, L$, is approximated by an individual cut approximation $\mathfrak{Q}_{t\ell}(\cdot)$.

In the backward pass of some iteration i, the stages are traversed in backward direction as usual to improve the cut approximations. At each stage t, the subproblems (102) updated with $\mathfrak{Q}_{t\ell}^{i+1}(\cdot)$ are solved for each trial state x_{t-1}^{ik}, $k \in \mathcal{K}$, each stage-t Markov state $\psi_t = \zeta_\ell, \ell = 1, \dots, L$, and all realizations $\xi_{tj}^\ell, j = 1, \dots, q_t^\ell$.

Then, for each x_{t-1}^{ik} and $\psi_{t-1} = \zeta_\ell, \ell = 1, \dots, L$, a valid cut can be derived for $\mathcal{Q}_{t\ell}(\cdot)$. Let $\beta_{t\ell kj}^i$ denote a subgradient for $\underline{Q}_{t\ell}^i(\cdot, \cdot)$ at x_{t-1}^{ik}. In accordance with (16), but also taking into account the Markov chain transition probabilities, we can then define cut coefficients

$$\beta_{t\ell k}^i := \sum_{\ell'=1}^L \omega_{\ell\ell'} \left(\sum_{j=1}^{q_{t\ell}} p_{t\ell j} \left(\underline{Q}_{t\ell}^{i+1}(x_{t-1}^{ik}, \xi_t^{\ell k}) - (\beta_{t\ell kj}^i)^\top x_{t-1}^{ik} \right) \right),$$

$$\alpha_{t\ell} := \sum_{\ell'=1}^L \omega_{\ell\ell'} \left(\sum_{j=1}^{q_{t\ell}} p_{t\ell j} \beta_{t\ell kj}^i \right),$$

where $q_{t\ell}$ and $p_{t\ell j}$ denote the number of realizations and probabilities of $\boldsymbol{\xi}_t^\ell$.

A cut (17) for $\mathcal{Q}_{t\ell}(\cdot)$ is then given by function

$$\phi_{t\ell k}^i(x_{t-1}) := \alpha_{t\ell k}^i + (\beta_{t\ell k}^i)^\top x_{t-1}$$

and can be used to update $\mathfrak{Q}_{t\ell}^{i}(\cdot)$. Philpott et al. derive similar formulas for the multi-cut and risk-averse case [161].

Use Cases. There exist different use cases for modeling the uncertainty in (MSLP) with an integrated Markov chain.

- The data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ can be modeled as a nonlinear AR process or a nonlinear transformation of a linear AR process (see Sect. 9), which, if handled by expanding the state space, destroys the convexity of $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$. Sometimes such a nonlinear process can be approximated by assuming that the realizations ξ_t depend on an underlying system state which follows a Markov process [161], thus not capturing the nonlinearity explicitly in a formula. As the value functions are also not convex in this, possibly continuous, Markov state, the Markov process is approximated using a discrete Markov chain.
- Instead of a single AR process, sometimes the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ may be best modeled by a finite set of different AR processes, which are valid representations, and thus active, under different circumstances (*e.g.*, macroeconomic, political or ecological situations). A discrete Markov chain can then be used to model these overall system states, and AR models can be used to describe realizations of the uncertain data conditioned on these states. Such *regime-switching* models are very common in wind forecasting [233].
- *Hybrid SDP/SDDP.* Different parts of the data in (MSLP) exhibit stagewise dependent uncertainty. While some of them, namely uncertainty in the RHS h_t, can be treated by expanding the state space, for others, *e.g.*, stagewise dependent uncertainty in the objective coefficients c_t, it would destroy the convexity of $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$. Therefore, this part of the uncertainty may be modeled by a discrete Markov chain instead. Since one part of the uncertainty is treated as in standard SDDP (allows for cut-sharing between scenarios), while another one is treated by enumerating separate expected value functions for each $\ell = 1,\dots,L$ (cuts cannot be shared between Markov states), this is often referred to as a hybrid SDP/SDDP method [79].

  For instance, this setting often occurs in medium-term hydrothermal scheduling problems (see Sect. 9) when inflow uncertainty in the RHS as well as spot-price uncertainty in the objective function are taken into account. The idea to address this by using a Markov chain goes back to Gjelsvik et al. who modeled this kind of scheduling problem for the Norwegian power system [79, 81, 82]. Since then, this approach has been employed in several applications, for example, hydrothermal scheduling including balancing market bids [100, 101], risk management [108, 116, 142] and fuel contracts [37]. It is also applied to model fuel price uncertainty [151].

  In contrast to the presented general approach, in this case it is usually assumed that the uncertainty in the RHS and in the objective are independent of each other. Therefore, for each state $\zeta_\ell, \ell = 1,\dots,L$, the distribution of $\boldsymbol{\xi}_t$ is the same, and marginal distributions can be used in the expectation in (100). Moreover, note that in this specific case the Markov chain states are not underlying the distribution of $\boldsymbol{\xi}_t$, but instead entering the subproblems explicitly, *e.g.*, as objective coefficients. Still SDDP can be applied using the same ideas as above.

The described approach allows for the incorporation of even nonlinear stagewise dependent uncertainty into SDDP, but also gives rise to some challenges. Among those is the assumption of the Markov property, which may not always be appropriate. Moreover, it is required to define useful values $\zeta_\ell, \ell = 1, \dots, L$, and transition probabilities $\omega_{\ell\ell'}$ for the Markov states [81, 143]. Most importantly, cuts cannot be shared between, but only within Markov states, so that separate expected value functions have to be considered for each $\ell = 1, \dots, L$. Therefore, the number of Markov states should be rather small to preserve computational tractability.

14.6 Hybrid NBD/SDDP

In the previous section, we presented a hybrid SDP/SDDP method as a tool to model different stagewise dependent uncertain data in (MSLP) by different approaches. Instead of modeling the "complicating" part of the uncertainty by a discrete Markov chain, also a scenario tree can be used. Instead of a hybrid SDP/SDDP method, this yields a hybrid NBD/SDDP method [179], see also Sect. 5.2.

Assume that the random vector $\boldsymbol{\xi}_t$ modeling the uncertainty in c_t, W_t, T_{t-1} and h_t can be separated into two separate and independent parts, $\boldsymbol{\xi}_t^S$ and $\boldsymbol{\xi}_t^T$. The first vector $\boldsymbol{\xi}_t^S$ can either be stagewise independent or exhibit some linear dependency if it occurs in the RHS. In the latter case, it can be handled by expanding the state space. Within SDDP, in each iteration samples of $\boldsymbol{\xi}_t^S$ are considered. The second vector $\boldsymbol{\xi}_t^T$, on the other hand, may lead to non-convexities in the value functions if it is approached by expanding the state space. Therefore, it is modeled by a scenario tree, which is treated exactly in SDDP. This means that for this particular part of the uncertainty, no samples are drawn, but all scenarios are considered in each iteration of SDDP, as in NBD, see Sect. 5.2. This approach is similar to hybrid SDP/SDDP in the sense that the expected value functions $\mathcal{Q}_t(\cdot)$ depend on the scenarios from $\boldsymbol{\xi}_t^S$ and that cuts can only be shared within, but not between such scenarios.

By only treating the crucial part $\boldsymbol{\xi}^T$ of $\boldsymbol{\xi}$ as a scenario tree and the remainder $\boldsymbol{\xi}^S$ still by sampling, complex uncertainty processes can be considered, while at the same time the increase of computational complexity is kept as small as possible [179]. To take advantage of this, the scenario tree associated with $\boldsymbol{\xi}^S$ should not be too large.

Compared to hybrid SDP/SDDP, in specific applications the one or the other approach may be favorable. For instance, the Markov chain approaches allow for dependencies between different uncertainty processes. Moreover, in the case that each realization of $\boldsymbol{\xi}_t$ is assigned to one specific Markov state $\zeta_\ell, \ell = 1, \dots, L$, the number of LPs to be solved per iteration can be kept equal to standard SDDP. The scenario tree approach, by contrast, requires independence of $\boldsymbol{\xi}^S$ and $\boldsymbol{\xi}^T$. By design, it considers all combinations of scenarios of $\boldsymbol{\xi}^T$ and $\boldsymbol{\xi}^S$, so no assignment of realizations of $\boldsymbol{\xi}^S$ to scenarios of $\boldsymbol{\xi}^T$ is required. However, the number of LPs to be solved grow exponentially in the number of stages [179]. On the other hand, a scenario tree may be more appropriate to model very complex processes, *e.g.*, referring to macroeconomical, political or structural decisions [179], for which the Markov property is not appropriate.

14.7 Saddle Cuts

We consider the special case of stagewise dependent objective coefficients $\boldsymbol{c}_t(\xi_t)$ in (MSLP), as they appear for uncertain prices models by AR processes. So far, we introduced SDDP with integrated Markov chain as a suitable solution approach in this case. Now, we discuss as second one.

As discussed in Sect. 14.1, by expanding the state space, stagewise independence

(Assumption 2) can be recovered, but in return the expected value functions $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ are no longer convex. In Theorem 14.3 it is shown that $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ is in fact convex in x_{t-1}, but concave in c_{t-1}, which yields a saddle shape. Therefore, linear cuts are not sufficient to approximate them. As a resort, exploiting the saddle shape, special *saddle cuts* can be used.

To derive this formally, in the vein of [55], we assume the objective coefficients to be described by $\big(y_t(\xi_t)\big)^\top C_t$ instead of $c_t(\xi_t)$. While the matrix C_t is considered deterministic, $y_t(\xi_t)$ is defined by the following AR process

$$y_t(\xi_t) = B_t(\xi_t) y_{t-1}(\xi_{t-1}) + b_t(\xi_t) \tag{103}$$

for all stages $t = 2, \dots, T$. Here, the matrix B_t and the vector b_t are uncertain and depend on the realization of $\boldsymbol{\xi}_t$. Thus, the sequence $\big(y_t(\xi_t)\big)_{t=1}^T$ is scenario-dependent.

Inserting relation (103) into the objective function and considering y_{t-1} as an additional state variable, for $t = 2, \dots, T$, we obtain the subproblems

$$\begin{aligned} &\widehat{Q}_t(x_{t-1}, y_{t-1}, \xi_t) \\ &= \begin{cases} \min\limits_{x_t} & \big(B_t(\xi_t) y_{t-1} + b_t(\xi_t)\big)^\top C_t x_t + \widehat{\mathcal{Q}}_{t+1}\big(x_t, B_t(\xi_t) y_{t-1} + b_t(\xi_t)\big) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}, \xi_t) \end{cases} \end{aligned}$$

where

$$\widehat{\mathcal{Q}}_{t+1}(x_t, y_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\Big[\widehat{Q}_{t+1}(x_t, y_t, \boldsymbol{\xi}_{t+1}))\Big]$$

and $\widehat{\mathcal{Q}}_{T+1}(x_T, y_T) \equiv 0$. For the first stage, we obtain

$$v^* = \begin{cases} \min\limits_{x_1} & b_1 C_1 x_1 + \widehat{\mathcal{Q}}_2(x_1, y_1) \\ \text{s.t.} & x_1 \in \mathcal{X}_1. \end{cases}$$

The additional state y_{t-1} is referred to as an *objective state*. This state is not allowed to appear in the constraints [55]. As stated before, $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ is piecewise linear and convex in x_{t-1}, but piecewise linear concave in y_{t-1} and as such, a piecewise bilinear saddle function.

The concept of approximating saddle functions with saddle cuts goes back to Baucke et al., who propose a deterministic algorithm to solve stochastic minimax dynamic programs [11]. A related approach is used in *robust dual dynamic programming* (RDDP), which uses an SDDP-like framework to solve multistage robust programs [76]. The main idea is to compute lower and upper bounding saddle functions, which combine the ideas of an outer approximation by cutting-planes and an inner approximation by convex combinations of function values, the latter of which we discuss thoroughly in Sect. 8. For stagewise dependent objective coefficients, it is sufficient to only use the lower bounding saddle functions, so-called *saddle cuts*, from [11] to approximate the expected value functions in SDDP.

Let (16) define β_t and α_t as in standard SDDP. Then, the r-th saddle cut for $\widehat{\mathcal{Q}}_{t+1}(\cdot,\cdot)$

is defined as the solution to the optimization problem

$$\begin{aligned} \min_{\mu_t,\ \theta_{t+1}} \quad & y_t^\top \mu_t + \theta_{t+1} \\ \text{s.t.} \quad & (y_t^r)^\top \mu_t + \theta_{t+1} \geq \alpha_{t+1}^r + (\beta_{t+1}^r)^\top x_t \\ & \|\mu_t\|_\infty \leq \nu \end{aligned} \tag{104}$$

where $y_t^r = y_t^{ik}$ denotes the current objective state in iteration i and for scenario $k \in \mathcal{K}$. Importantly, this problem has x_t and y_t as parameters. Hence, a saddle cut gives a valid lower approximation for $\widehat{\mathcal{Q}}_{t+1}(\cdot,\cdot)$ for all x_t and y_t and can be shared between scenarios. Moreover, the saddle cuts are tight at the trial state given by x_t^{ik} and y_t^{ik}, at which they are created.

A crucial part of applying this approach is to bound the decision variable μ_t in (104) by an appropriate constant ν. To this end, the expected value functions $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ are required to be Lipschitz continuous with respect to y_{t-1}. As shown in [11], to ensure validity of the saddle cuts, the parameter ν has to be chosen at least as large as the Lipschitz constant of $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ with respect to y_{t-1} under the dual norm $\|\cdot\|_1$ of $\|\cdot\|_\infty$. If it is chosen smaller, this may result in invalid cuts and suboptimal solutions. If it is chosen too large, the cuts may become very weak [55].

Incorporating the saddle cuts, for each stage $t = 2,\dots,T$, iteration i and scenario $k \in \mathcal{K}$, the SDDP subproblems can be formulated as

$$\begin{aligned} &\underline{\widehat{Q}}_t^i(x_{t-1}^{ik}, y_{t-1}^{ik}, \xi_{tj}) \\ &= \begin{cases} \min\limits_{x_t,\mu_t,\theta_{t+1}} & (y_t^{ik})^\top C_t x_t + (y_t^{ik})^\top \mu_t + \theta_{t+1} \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^{ik}, \xi_{tj}) \\ & (y_t^r)^\top \mu_t + \theta_{t+1} - (\beta_{t+1}^r)^\top x_t \geq \alpha_{t+1}^r, \quad r \in \Gamma_{t+1} \\ & \|\mu_t\|_\infty \leq \nu, \end{cases} \end{aligned}$$

where $y_t^{ik} = B_t(\xi_t^k) y_{t-1}^{ik} + b_t(\xi_t^k)$.

It can be shown that only finitely many different saddle cuts can be constructed. As a consequence, the convergence results are the same as for standard SDDP [55].

14.8 Applying Dual SDDP

A third alternative that is tailored to stagewise dependent objective coefficients $\boldsymbol{c}_t(\xi_t)$ in (MSLP) is to apply dual SDDP [97], as presented in Sect. 8. Recall the value functions derived from the dual problem of (MSLP):

$$\widetilde{D}_t(\pi_{t-1}) := \begin{cases} \max\limits_{\pi_t} & \sum\limits_{j=1}^{q_t} p_{tj} \Big(- h_{tj}^\top \pi_{tj} + \widetilde{D}_{t+1}(\pi_{tj}) \Big) \\ \text{s.t.} & \sum\limits_{j=1}^{q_t} p_{tj} \Big(T_{t-1,j}^\top \pi_{tj} \Big) + W_{t-1}^\top \pi_{t-1} \leq c_{t-1}. \end{cases} \tag{105}$$

These value functions are concave in π_{t-1}. Crucially, here the objective coefficients c_{t-1} appear in the RHS. If $(\boldsymbol{c}_t)_{t\in[T]}$ is described as a linear AR process, we can expand the state space as for the primal subproblems in Sect. 14.1, and the new state variable $c_{[t-2]}$ appears in the RHS. Therefore, the obtained value functions are also concave in $c_{[t-2]}$ and can be approximated from above by linear cuts. This can be done by applying dual SDDP [97], see Sect. 8.

14.9 Conditional Cuts

The previously discussed approaches all have in common that they require to expand the state space or to set up a scenario tree or a discrete Markov chain from the true (continuous) data process (or from existing historical data). van Ackooij and Warin propose an alternative approach that works without these requirements [227]. The approach is based on established methods in mathematical finance and optimal stopping theory. A crucial assumption is that the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ is Markovian.

Assume that a finite set $\mathcal{S}$ of scenarios ξ^s, $s \in \mathcal{S}$, is given, *e.g.*, historical observations of the data. This set is chosen in advance and not changed within SDDP. The first key ingredient of the proposed variant of SDDP is to partition the set of possible values of $\boldsymbol{\xi}_t$ for each stage $t \in [T]$ into a finite number $|L_t|$ of hypercubes $D_{t\ell}$, $\ell = 1, \dots, |L_t|$, also called *meshes*. This partitioning is done in such a way that approximately a uniform distribution of the samples is achieved [227].

As we explain below, the main idea now is to compute cuts conditioned on specific meshes, *i.e.*, for each mesh a different set of cuts is considered.

In the forward pass of SDDP, a subset $L_t \subseteq \mathcal{S}_t$ of scenarios are sampled for each stage. This is done with the aim to obtain a trial solution x_t^ℓ for each mesh in expectation for all $t = 2, \dots, T$. Each of these trial solutions is then used in the backward pass to derive cuts.

In the backward pass, for any sequence $(x_t^{i\ell})_{t\in[T]}$ of trial solutions, let $\big(d(t)^{i\ell}\big)_{t\in[T]}$ denote the sequence of corresponding meshes, *i.e.*, $x_t^{i\ell}$ has been determined in the forward pass for $\xi_t^\ell \in D_{t,d(t)^{i\ell}}$. At each stage $t = T, \dots, 2$, the SDDP subproblems are now solved for all scenarios ξ_t^s for which $\xi_{t-1}^s \in D_{t-1,d(t-1)^{i\ell}}$. This means that for each trial solution, all scenarios are considered which share the same mesh with the scenario used to obtain the trial solution.

After solving these subproblems, the obtained solutions are used to construct cuts. In contrast to standard SDDP, however, the cut coefficients are determined as estimates of the corresponding conditional expectations [227]:

$$\alpha_{t\ell}^i(\xi_{t-1}) = \hat{\mathbb{E}}_{|\xi_{t-1}}^S \Big[(\boldsymbol{\pi}_t^{i\ell s})^\top \boldsymbol{h}_t(\xi_t) + \sum_{r\in\Gamma_{t+1}} \boldsymbol{\rho}_t^{i\ell sr} \alpha_{t+1}^r \Big]$$

and

$$\beta_{t\ell}^i(\xi_{t-1}) = -\hat{\mathbb{E}}_{|\xi_{t-1}}^S \Big[(\boldsymbol{\pi}_t^{i\ell s})^\top T_{t-1} \Big].$$

These estimates are computed by linearly regressing the terms for each considered scenario ξ_t^s on a finite number of local base functions, *e.g.*, monomials in $\mathbb{R}^{p_t}$, with support on the considered mesh. Importantly, they are zero outside of this mesh. The idea is that this way a cut of form

$$\mathcal{Q}_t(x_{t-1}, \xi_{t-1}) \geq \phi_{t\ell}^i(x_{t-1}, \xi_{t-1}) = \big(\beta_{t\ell}^i(\xi_{t-1})\big)^\top x_{t-1} + \alpha_{t\ell}^i(\xi_{t-1}), \tag{106}$$

can be constructed, which provides a local update of the cut approximation in the current mesh $D_{t-1,d(t-1)}$ and is zero otherwise. Hence, the cut is associated with this specific mesh and stored in a corresponding index set. In following iterations of SDDP, for each subproblem then only the set of cuts is taken into account which is associated with the currently explored mesh [227]. Therefore, these cuts are called *conditional cuts*.

The main drawback of this approach is that the described cuts are not guaranteed to be valid underestimators, so the inequality in (106) is not guaranteed to be satisfied, because their formula relies on *estimators* that may deviate from the true conditional expectations.

Still, for problems with a low-dimensional vector $\boldsymbol{\xi}_t$ and Markovian dependency, the policies obtained using conditional cuts are reported to be competitive with those obtained by expanding the state space, but without an increase of the state dimension and without the need to set up a scenario tree [227].

15 Extension to Convex Programs [relaxing Assumption 6]

A natural extension of SDDP can be achieved by relaxing the assumption of linearity, *i.e.*, Assumption 6, but assuming a multistage stochastic convex problem (MSCP). In the same vein as problem (3), this problem can be formulated in the general form

$$v_C^* := \begin{cases} \min\limits_{x_1,\boldsymbol{x_2},\dots,\boldsymbol{x_T}} & \mathbb{E}\left[\sum\limits_{t\in[T]} \boldsymbol{f}_t(\boldsymbol{x}_t(\xi_{[t]}),\xi_t)\right] \\ \text{s.t.} & g_1(x_1) \le 0 \\ & g_t(\boldsymbol{x}_{t-1}(\xi_{[t-1]}),\boldsymbol{x}_t(\xi_{[t]}),\xi_t) \le 0 \quad \forall \xi_{[t]}\ \forall t = 2,\dots,T \\ & x_t \in X_t \quad \forall t \in [T] \\ & \boldsymbol{x}_t(\cdot)\ \mathscr{F}_t\text{-measurable} \quad \forall t \in [T], \end{cases} \tag{107}$$

with $\boldsymbol{f}_t(\cdot)$ and $\boldsymbol{g}_t(\cdot,\cdot)$ some $\mathscr{F}_t$-measurable functions with respect to $\boldsymbol{\xi}$.

We take the following assumptions [78, 85].

Assumption 10. *For fixed $\xi_t \in \Xi_t$, let $f_t(\cdot,\xi_t)$ and $g_t(\cdot,\cdot,\xi_t)$ (componentwise) be proper, convex, lower semicontinuous and differentiable functions and X_t nonempty convex compact sets for all $t \in [T]$.*

Under stagewise independence (Assumption 2), finite randomness (Assumption 5) and Assumption 10, (MSCP) in (107) can be expressed using its DPE in the following form. For $t = 2,\dots,T$ they read

$$Q_{t,C}(x_{t-1},\xi_t) := \begin{cases} \min\limits_{x_t} & f_t(x_t,\xi_t) + \mathcal{Q}_{t+1,C}(x_t) \\ \text{s.t.} & g_t(x_{t-1},x_t,\xi_t) \le 0 \\ & x_t \in X_t, \end{cases} \tag{108}$$

with expected value functions defined as usual by

$$\mathcal{Q}_{t+1,C}(x_t) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}}\left[Q_{t+1,C}(x_t,\boldsymbol{\xi}_{t+1})\right] \tag{109}$$

and $\mathcal{Q}_{T+1,C}(x_T) \equiv 0$. For the first stage, this yields

$$v_C^* = \begin{cases} \min\limits_{x_1} & f_1(x_1) + \mathcal{Q}_{2,C}(x_1) \\ \text{s.t.} & g_1(x_1) = 0 \\ & x_1 \in X_1. \end{cases} \tag{110}$$

Applying SDDP to (MSCP) with convergence guarantees requires a more strict recourse assumption compared to Assumption 9.

Assumption 11. *(Extended relatively complete recourse [78]) Let aff$(\mathcal{X}_t)$ be the affine hull of the reachable set $\mathcal{X}_t$ and $B_t(\delta_t) = \{y \in$ aff$(\mathcal{X}_t) \; : \; \|y\| < \delta_t\}$ for some $\delta_t > 0$ and some norm $\|\cdot\|$.*

For all $t \in t = 2, \dots, T$, all $x_{t-1} \in \mathcal{X}_{t-1} + B_t(\delta_t)$ and all $\xi_{tj}, j = 1, \dots, q_t$, the feasible set of subproblems (108) *is non-empty.*

Intuitively, Assumption 11 demands that feasibility of the subproblems is also ensured for x_{t-1} slightly outside of $\mathcal{X}_t$. This is required in order to guarantee Lipschitz continuity of all value functions $Q_{t,C}(\cdot,\cdot)$ and expected value functions $\mathcal{Q}_{t,C}(\cdot)$ [78]. Additionally, all value functions are convex, and thus can be approximated by linear cuts. Such cuts can be generated using Lagrangian duality. More precisely, for all $t = 2, \dots, T$, $x_{t-1} \in \mathcal{X}_{t-1}$ and some multipliers $\pi_t \in \mathbb{R}^{m_t}$ (with m_t the dimension of $g_t(\cdot,\cdot)$), we introduce the Lagrangian function

$$L_{t,C}(\pi_t; x_{t-1}, x_t, \xi_t) = f_t(x_t, \xi_t) + \pi_t^\top g_t(x_{t-1}, x_t, \xi_t), \tag{111}$$

the corresponding dual function

$$\mathcal{L}_{t,C}(\pi_t; x_{t-1}, \xi_t) = \min_{x_t \in X_t} L_t(\pi_t; x_{t-1}, x_t, \xi_t) \tag{112}$$

and the corresponding Lagrangian dual problem

$$\max_{\pi_t \geq 0} \mathcal{L}_t(\pi_t; x_{t-1}, \xi_t). \tag{113}$$

Further, we make the following assumption which ensures no duality gap between the primal subproblems (108) and their dual problems (113) [85]. Here, ri(S) denotes the relative interior of some set S.

Assumption 12. *(Slater condition [85]) For all $x_{t-1} \in \mathcal{X}_{t-1}$ and all $\xi_{tj}, j = 1, \dots, q_t$, there exists $x_t \in ri(X_t)$ such that $g_t(x_{t-1}, x_t, \xi_{tj}) < 0$.*

Then, exploiting differentiability, a subgradient of $\mathcal{Q}_{t,C}(\cdot)$ at $\bar{x}_{t-1}$ is given by

$$\bar{\beta}_t = \partial \mathcal{Q}_t(\bar{x}_{t-1}) = \sum_{j=1}^{q_t} p_{tj} \nabla_{x_{t-1}} L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj}),$$

where $\bar{x}_{tj}$ is an optimal solution to the primal problem (108) and $\bar{\pi}_{tj}$ is an optimal solution to the dual problem (113) given ξ_{tj}. Moreover, $\nabla_x h(\cdot)$ denotes the gradient of some function $h(\cdot)$ with respect to x. Using this subgradient, a cut for $\mathcal{Q}_t(\cdot)$ is given by [85]

$$\mathcal{Q}_t(x_{t-1}) \geq \mathcal{Q}_t(\bar{x}_{t-1}) + \bar{\beta}_t^\top (x_{t-1} - \bar{x}_{t-1}).$$

Under Assumption 11, the norm of the obtained subgradients can be shown to be bounded [85].

This cut derivation can be generalized to DPE including $\mathfrak{Q}_t(\cdot)$ instead of $\mathcal{Q}_t(\cdot)$. The results can also be generalized to cost functions $f_t(x_{t-1}, x_t, \xi_t)$ depending on the state x_{t-1}, see [85] for details.

Contrary to the linear case, however, the expected value functions $\mathcal{Q}_{t,C}(\cdot)$ are no longer polyhedral. As a consequence, they cannot be represented exactly by a finite number of cuts. However, it can be shown that given the above assumptions and Assumptions 1 to 8 almost sure asymptotic convergence of SDDP is ensured. In [78] this is proven for the case that x_{t-1} only enters the subproblems (108) in linear constraints, that is, $g_t(\cdot)$ being a linear function. In [85] the convergence proof is extended to the more general setting presented above. For both convergence proofs also the differentiability requirement can be dropped. As shown in [71], almost sure *finite* convergence can be achieved for ε-optimal policies, for some predefined $\varepsilon > 0$.

In [92], Guigues and Monteiro propose a slightly different algorithmic approach, called StoDCuP (Stochastic Dynamic Cutting Plane), in which not only $\mathcal{Q}_t(\cdot), t = 2, \ldots, T$, but also some or all nonlinear functions $f_t(\cdot)$ and $g_t(\cdot)$ are iteratively approximated by affine functions at the trial points visited in the forward pass.

Another variant of SDDP is DASC (decomposition algorithm for multistage stochastic programs with strongly convex cost functions), which is introduced in [86]. It can be applied when the (expected) value functions in (MSCP) are *strongly convex.* For this type of problems, it is proposed to approximate them using functions $\mathfrak{Q}_t(\cdot)$ which are defined as the pointwise maximum of quadratic cuts instead of affine cuts. In contrast to standard SDDP, this means that the subproblems to be solved in SDDP become nonlinear, but in return good approximations of the expected value functions are obtained much quicker, and thus less iterations are expected [86]

While most research on SDDP deals with problems (MSLP), some of the extensions presented previously and in the following sections have also been enhanced to the convex case, *e.g.*, risk-aversion [85], inexact cuts [88], regularization [90] or exact upper bounding procedures [10, 119]. [85] contains an extension of the convergence proof from [78] to the risk-averse case. Furthermore, the idea to use inexact cuts is generalized to convex non-differentiable problems [91], see Sect. 21.

16 Extensions to Mixed-integer and Non-convex Problems [relaxing Assumption 6]

In many practical applications, multistage stochastic problems do involve integer decision variables or nonlinear, but non-convex terms in the objective function or constraints, see Sect. 9. In general, such programs can be formulated in the same way as in the convex case, but with the functions $f_t(\cdot)$ and $g_t(\cdot)$ possibly being non-convex. Moreover, in this case, X_t is the intersection of a convex compact set, *e.g.*, representing box constraints, with possible integer constraints, *i.e.*, $X_t \subset \mathbb{R}_t^{n_{t1}} \times \mathbb{Z}_+^{n_{t2}}$ with $n_t = n_{t1} + n_{t2}$. We denote the optimal value by v_{NC}^*.

Under stagewise independence (Assumption 2), the DPE can be written as (108)-(110), but for distinction we denote the value functions by $Q_{t,NC}(x_{t-1}, \xi_t)$ and the expected value functions by $\mathcal{Q}_{t,NC}(x_{t-1})$ for all $t = 2, \ldots, T$. Both, integer variables and non-convex functions make this a non-convex multistage stochastic programming problem (MSNCP). Importantly, $Q_{t,NC}(\cdot, \cdot)$ and $\mathcal{Q}_{t,NC}(\cdot)$ are no longer ensured to be convex, but become non-convex functions in x_{t-1}. They are also not guaranteed to be (Lipschitz) continuous. This poses significant challenges on approximation algorithms such as SDDP, as linear cuts are not sufficient to approximate $\mathcal{Q}_{t,NC}(\cdot)$.

To approach (MSNCP) by SDDP, different strategies can be used. As nonlinear or mixed-integer stochastic programming are large research areas on their own, we give a brief overview here and for methodological details refer to the cited literature.

16.1 Convexification

A standard approach in practice is to solve a static convex relaxation ($\widehat{P}_{NC}$) of (MSNCP), which is associated with convex expected value functions $\widehat{\mathcal{Q}}_t(\cdot)$ for all $t \in [T]$. Such relaxation can be achieved by relaxing the integrality constraints and replacing non-convex functions with convex relaxations, such as McCormick envelopes [140]. In this case, the Benders cuts determined by SDDP can be very loose, though. Therefore, only some rough under-approximation $\widehat{v}^*_{NC}$ of the optimal value v^*_{NC} may be determined. However, sometimes this is considered sufficient to obtain reasonable policies for practical implementation. Also note that even if convex relaxations are considered when running SDDP to compute a policy, the simulation of this policy afterwards can be executed including integrality constraints and non-convex functions.

A second strategy is to keep the subproblems in SDDP non-convex, but to convexify the expected value functions $\mathcal{Q}_{t,NC}(\cdot)$ *in some sense*. Often, in this case, the nonlinearities in (MSNCP) are first relaxed by piecewise linear approximations, such that all subproblems are MILPs [36, 219]. In the backward pass, given some incumbent x^{ik}_{t-1}, for all $t = T, \dots, 2$ and all $\xi_{tj}, j = 1, \dots, q_t$, instead of solving an LP relaxation of the subproblems (10) (or its LP dual), a Lagrangian relaxation is solved where the coupling constraints $g_t(x_{t-1}, x_t, \xi_{tj}) \leq 0$ are relaxed. For any trial point x^{ik}_{t-1} and any multiplier $\pi_t \in \mathbb{R}^{m_t}$, this relaxation can be written as

$$\begin{aligned} \mathfrak{L}^{i+1}_t(\pi_t; x^{ik}_{t-1}, \xi_{tj}) := \min_{x_t} \quad & f_t(x_t, \xi_{tj}) + \mathfrak{Q}_{t+1}(x_t) + \pi_t^\top g_t(x^{ik}_{t-1}, x_t, \xi_{tj}) \\ \text{s.t.} \quad & x_t \in \mathcal{X}_t. \end{aligned}$$

In the Lagrangian dual, this dual function is maximized over all multipliers π_t:

$$v^{i+1}_{t,LD}(x^{ik}_{t-1}, \xi_{tj}) := \max_{\pi_t \geq 0} \; \mathfrak{L}^{i+1}_t(\pi_t; x^{ik}_{t-1}, \xi_{tj}). \tag{114}$$

It is known from the theory on Lagrangian relaxation that the optimal value $v^{i+1}_{t,LD}(x^{ik}_{t-1}, \xi_{tj})$ coincides with the lower convex envelope of $\underline{Q}^{i+1}_{t,NC}(\cdot, \xi_{tj})$ at x^{ik}_{t-1} [75]. Therefore, cuts obtained based on (114) are associated with a convexification of the value function. In order to derive utilizable cut formulas from (114) some specific conditions have to be satisfied by the constraints. Suppose the constraints $g_t(x_{t-1}, x_t, \xi_t) \leq 0$ can be rewritten as

$$\widehat{g}_t(x_{t-1}) - \bar{g}_t(x_t, \xi_t) \leq 0, \quad \tilde{g}_t(x_t, \xi_t) \leq 0,$$

i.e., the nonlinear function being separable with respect to x_{t-1}, and let π^{ikj}_t denote optimal multipliers in (114). Then, in line with Sect. 3.3, *Lagrangian cuts* can be derived as [213]

$$\mathcal{Q}_{t,NC}(x_{t-1}) \geq \alpha^i_{tk} + (\beta^i_{tk})^\top \widehat{g}_t(x_{t-1}),$$

with

$$\alpha_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \Big(\mathfrak{L}_t(\pi_t^{ikj}; x_{t-1}^{ik}, \xi_{tj}) - (\pi_t^{ikj})^\top \widehat{g}_t(x_{t-1}^{ik}) \Big),$$

$$\beta_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \pi_t^{ikj}.$$

For linear functions $\widehat{g}_t(\cdot)$ and $\bar{g}_t(\cdot,\cdot)$, a similar result is derived in [36].

The obtained Lagrangian cuts provably dominate standard Benders cuts, which can be obtained by solving LP relaxations [213]. However, convergence of SDDP is not guaranteed, since there may still be some duality gap between $v_{t,LD}^{i+1}(x_{t-1}^{ik}, \xi_{tj})$ and $\underline{Q}_{t,NC}^{i+1}(x_{t-1}^{ik}, \xi_{tj})$.

Moreover, generating Lagrangian cuts can be computationally costly. Various methods have been proposed to solve the Lagrangian dual (114), such as cutting-plane methods [111], subgradient methods [69, 167] or bundle methods [122], but all of them may take considerable time compared to solving an LP relaxation. Advantageously, even suboptimal Lagrangian multipliers π_t yield valid cuts for $\mathcal{Q}_{t,NC}(\cdot)$.

Instead of a static convexification approach [36], Steeger and Rebennack [211, 213], also apply the above principle in a dynamic fashion by considering DPE for the Lagrangian relaxations in the backward pass.

16.2 Exact Methods

Recently, there has been more research on directly applying the SDDP idea to problems (MSNCP) to avoid the requirement of convexification and to close the optimality gap.

Step Functions. Given that the value functions $Q_{t,NC}(\cdot)$ are monotonically increasing or decreasing, they can be approximated by special step functions instead of affine functions. This idea is incorporated into the SDDP framework in the so-called *mixed-integer dynamic approximation scheme* (MIDAS) [163]. To determine the step functions, mixed-integer linear subproblems have to be solved exactly at each stage and in each iteration. In contrast to the previous approaches, convergence of MIDAS to an approximately optimal policy for (MSNCP) is guaranteed.

SDDiP. For the mixed-integer *linear* case, the *stochastic dual dynamic integer programming* (SDDiP) approach by Zou, Ahmed and Sun [234] allows for the computation of optimal policies for (MSNCP) as long as all state variables x_t are binary (or bounded integer).

Consider the subproblems (10), but with binary state variables $x_t \in \{0,1\}^{n_t}$. Similarly to the approaches in [36, 213, 219], Lagrangian dual problems are solved in the backward pass to derive valid cuts. However, in SDDiP a new class of Lagrangian cuts is proposed. The crucial idea is to introduce local copies z_t of the state variables x_{t-1} and to relax the corresponding copy constraints in the Lagrangian relaxation:

$$\begin{aligned} \mathcal{L}_t^{i+1}(\pi_t; x_{t-1}^{ik}, \xi_{tj}) := \min_{x_t, z_t} \quad & \big(c_t(\xi_{tj})\big)^\top x_t + \mathfrak{Q}_{t+1}(x_t) + \pi_t^\top (x_{t-1}^{ik} - z_t) \\ \text{s.t.} \quad & x_t \in \mathcal{X}_t(z_t, \xi_t) \\ & z_t \in [0,1]^{d_{a(n)}}. \end{aligned}$$

In the Lagrangian dual, this dual function is maximized over all multipliers π_t:

$$\tilde{v}_{t,LD}^{i+1}(x_{t-1}^{ik}, \xi_{tj}) := \max_{\pi_t} \ \mathcal{L}_t^{i+1}(\pi_t; x_{t-1}^{ik}, \xi_{tj}).$$

Then, Lagrangian cuts can be determined as

$$\mathcal{Q}_{t,NC}(x_{t-1}) \geq \alpha_{tk}^i + (\beta_{tk}^i)^\top x_{t-1}, \tag{115}$$

with

$$\alpha_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \Big(\mathcal{L}_t(\pi_t^{ikj}; x_{t-1}^{ik}, \xi_{tj}) - (\pi_t^{ikj})^\top x_{t-1}^{ik} \Big),$$
$$\beta_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \pi_t^{ikj}.$$

These cuts can be proven to be valid and, in particular, tight, as defined in Lemma 3.3. The key aspect behind this tightness property is that for $x_{t-1} \in \{0,1\}^{n_t}$ the value functions $Q_{t,NC}(\cdot)$ coincide with their lower convex envelopes at all x_{t-1}. Therefore, Lagrangian cuts recovering the latter are also tight for the former.

Moreover, if only dual basic solutions are considered, the cuts (115) are also finite in the sense of Lemma 3.3. Therefore, almost sure finite convergence of SDDiP to an optimal policy of (MSNCP) is guaranteed [234].

If the state variables x_t are bounded and general integer or even continuous, they can be componentwise approximated by a (weighted) sum of binary variables in order to apply SDDiP [234]:

$$x_{tj} \approx \sum_{k=1}^{K_{tj}} 2^{k-1} \beta_{tj} \lambda_{tkj},$$

with discretization precision β_{ti} (for integer x_t, $\beta_t = 1$), binary variables $\lambda_{tkj}, k = 1, \ldots, K_{tj}$, and $K_{tj} \in \mathbb{N}$ for all $j = 1, \ldots, n_t$. Under some recourse assumptions, it can be proven that for a sufficiently fine binary expansion, an approximately optimal policy for (MSNCP) is computed. However, it may be challenging to choose an appropriate precision in advance in practice.

SDDiP is applied in the case studies [103], [175] and [234]. In the latter, additional non-convex functions occur in (MSNCP), which are linearized using a Big-M reformulation.

Non-convex Lipschitz cuts. As long as the value functions are assured to be Lipschitz continuous (*e.g.* because the *complete continuous recourse* [234] property is satisfied), the requirement of binary state variables can be dropped. This is exploited in the *stochastic Lipschitz dynamic programming* (SLDP) method proposed by Ahmed et al. in [1], which enhances SDDiP to general MILPs. In contrast to the Lagrangian cuts (115), here, two types of non-convex, but Lipschitz continuous cuts are derived to approximate $\mathcal{Q}_{t,NC}(\cdot)$: Reverse-norm cuts, which are constructed by using Lipschitz constants, and augmented Lagrangian cuts, which are based on (115), but contain an additional penalization term $-\mu \| x_{t-1} - x_{t-1}^i \|$, where μ denotes some user-controlled parameter and $\|\cdot\|$ some arbitrary norm.

This idea is further refined by Zhang and Sun in [231] who propose a new framework to solve multistage non-convex stochastic MINLPs as part of their complexity analysis of SDDP-like algorithms, see Sect. 4. The first key ingredient of their framework is to consider Lipschitz regularizations of the value functions, see Sect. 17.2. This ensures that the considered value functions are Lipschitz continuous without the requirement of restricting recourse assumptions for (MSNCP). The second idea is to construct non-linear *generalized conjugacy cuts* by solving conjugate dual problems, similar to the approach in SLDP. Whereas of theoretical interest, this method has not been applied in computational experiments yet. In particular, it is not clear how to solve the conjugate dual problems efficiently in general. Moreover, the framework requires the costly solution of MINLP subproblems in each iteration.

Based on concepts from SDDiP and [231], Füllner and Rebennack present a new framework to solve multistage (stochastic) non-convex MINLPs [73]. Here, the original MINLP is outer approximated by MILPs using piecewise linear relaxations, which are iteratively improved in an outer loop. In an inner loop, those MILPs are solved by an SDDP- and NBD-like decomposition scheme, which combines the Lipschitz regularization approach from [231] with binary approximation to generate non-convex cuts. In contrast to SDDiP, the binary approximation is applied only temporarily to derive linear cuts in the lifted binary space, which are then projected back to the original state space. The pointwise maximum of this projection yields a Lipschitz continuous non-convex cut for the value functions. The projection is computationally important, as it allows to construct cuts which are guaranteed to be valid also for the outer loop MINLPs. The binary approximation is dynamically refined within the algorithm, instead of a static choice in advance. Another key difference compared to the approach from [231] is that it is not required to solve MINLPs in each iteration to derive cuts. The cut projection closure for a non-convex and discontinuous value function is illustrated in Figure 14.

Similar to SLDP [1], however, it is required to introduce a potentially large number of auxiliary variables and constraints to express the non-convex approximations by mixed-integer linear constraints. While the framework in [73] is presented for deterministic problems, the inner loop decomposition method can be enhanced to the stochastic case. Therefore, by appropriate modifications of the refinement and stopping criteria, also the larger framework may be enhanced to stochastic problems.

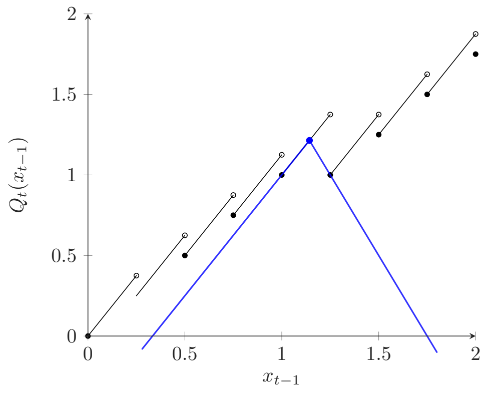


Figure 14: Non-convex and discontinuous value function with tight non-convex cut.

17 Infeasible Subproblems [relaxing Assumption 9]

Under relatively complete recourse (see Assumption 9), it is guaranteed that any subproblem occurring in the DPE (4)-(6) and their approximations (10) has a feasible solution. As we also assume boundedness, for each of these subproblems there exists some optimal point with finite optimal value. Moreover, all value functions are finite-valued.

In some practical applications, Assumption 9 may not be satisfied. For instance, variable bounds may prevent equality constraints from being satisfied for all x_{t-1} and all realizations of ξ_t, as is illustrated by a toy example in [84]. In such a case, the primal subproblems become infeasible and the corresponding dual problems become unbounded. Different measures can be taken to cope with infeasibilities.

17.1 Feasibility Cuts

One approach is to approximate the effective domains $\operatorname{dom}(\mathcal{Q}_t)$ of $\mathcal{Q}_t(\cdot)$ by cutting away states $x_{t-1}^{ik} \in \mathcal{X}_t$ leading to infeasible subproblems on stage t. This can be achieved by generating so called *feasibility cuts* in addition to the *optimality cuts* derived in Sect. 3. These cuts have the form $(\beta_t^f)^\top x_{t-1} \leq \alpha_t^f$, with cut gradient β_t^f, cut intercept α_t^f and the superscript f signifying the cut as a feasibility cut. They can be derived as follows [84].

Consider some stage-t subproblem

$$\underline{Q}_t^i(x_{t-1}^{ik}, \xi_t^k) = \begin{cases} \min\limits_{x_t} & \big(c_t(\xi_t^k)\big)^\top x_t + \mathfrak{Q}_{t+1}^i(x_t) \\ \text{s.t.} & x_t \in \mathcal{X}_t(x_{t-1}^{ik}, \xi_t^k) \\ & (\beta_{t+1}^{fr})^\top x_t \leq \alpha_{t+1}^{fr}, \quad r \in \Gamma_{t+1}^f \end{cases} \tag{116}$$

in the forward pass of SDDP. This problem may already contain some feasibility cuts, which are indexed by $r \in \Gamma_{t+1}^f$. To assess feasiblity of problem (116) and construct a feasibility cut if required, we consider the auxiliary feasibility problem

$$v_t^f(x_{t-1}^{ik}, \xi_t^k) := \begin{cases} \min\limits_{x_t, y_t^+, y_t^-, z_t} & e^\top y_t^+ + e^\top y_t^- + e^\top z_t \\ \text{s.t.} & W_t(\xi_t^k)x_t + Iy_t^+ - Iy_t^- = h_t(\xi_t^k) - T_{t-1}(\xi_t^k)x_{t-1}^{ik} \quad (\sigma_t) \\ & (\beta_{t+1}^{fr})^\top x_t + Iz_t \leq \alpha_{t+1}^{fr}, \quad r \in \Gamma_{t+1}^f \quad (\omega_t^r) \\ & x_t \geq 0 \\ & y_t^+, y_t^-, z_t \geq 0. \end{cases}$$

Here, slack variables y_t^+, y_t^- and z_t are introduced to (116) to ensure feasibility. The symbol I denotes the identity matrix and e denotes a vector of ones. If we have $v_t^f(x_{t-1}^{ik}, \xi_t^k) = 0$, the subproblem (116) is feasible, otherwise, it is infeasible.

By strong duality of linear programs, $v_t^f(x_{t-1}^{ik}, \xi_t^k)$ can be expressed as

$$v_t^f(x_{t-1}^{ik}, \xi_t^k) = \big(h_t(\xi_t^k) - T_{t-1}(\xi_t^k)x_{t-1}^{ik}\big)^\top \sigma_t + \sum_{r \in R_{t+1}^f} (\alpha_{t+1}^{fr})^\top \omega_t^r \tag{117}$$

with optimal dual vectors σ_t^{ik} and $\omega_t^{ikr}, r \in R_{t+1}^f$. Then, in case of infeasibility it follows that the term in (117) is larger than 0.

To avoid the observed infeasibility on stage t in future iterations, the stage-$(t-1)$ trial point x_{t-1}^{ik} should removed from the feasible set on stage $t-1$. This can be achieved by adding the feasibility cut

$$-(\sigma_t^{ik})^\top T_{t-1}(\xi_t^k)x_{t-1} + (\sigma_t^{ik})^\top h_t(\xi_t^k) + \sum_{r \in R_{t+1}^f} (\omega_t^{ikr})^\top \alpha_{t+1}^{fr} \leq 0 \tag{118}$$

to stage $t-1$. By defining

$$\alpha_{t-1}^f := -(\sigma_t^{ik})^\top h_t(\xi_t^k) - \sum_{r \in R_{t+1}^f} (\omega_t^{ikr})^\top \alpha_{t+1}^{fr}$$

and

$$\beta_{t-1}^f := -(\sigma_t^{ik})^\top T_{t-1}(\xi_t^k),$$

the cut (118) can be expressed in the previously stated form.

An important question when using feasibility cuts in SDDP is how to proceed, once an infeasible subproblem has been detected and a new feasibility cut (118) has been generated. For example, it is possible to stop the forward pass and traverse the stages in backward direction until the root node of the scenario tree is reached. Alternatively, the current subproblem can be resolved to obtain a new trial point x_{t-1}^{ik} and the forward pass can be continued. For SDDP, no assessment and comparison of these strategies has been conducted so far.

Another drawback is that feasibility cuts do not necessarily prevent infeasibilities when the obtained policy is simulated outside of SDDP [84]. For this reason, most commonly, the construction of feasibility cuts is circumvented in SDDP.

17.2 Penalization

Another common approach is to artificially enforce relatively complete recourse for a problem at hand, even if it is not satisfied initially. This can be achieved by using *soft-constraints*, that is, introducing slack variables to relax certain constraints and then penalizing their violation in the objective function. In some applications, this may even be practically justifiable, *e.g.*, in load balance equations in power optimization slack variables can be used to model load shedding or curtailment. However, a reasonable choice of the penalty parameters is not trivial and may distort the expected value functions [84].

Lipschitz Regularization. A specific penalization approach is to consider *Lipschitz regularizations*, also called *Pasch-Hausdorff envelopes* of the value functions. More precisely, let $\|\cdot\|$ denote some norm, $\sigma_t > 0$ some constant and z_t a local stage-t copy of x_{t-1}. Then, by allowing z_t to deviate from the incumbent x_{t-1}^{ik} and penalizing such deviations in the objective, for all $t = 2, \dots, T$ and the approximate value functions (10) we obtain the approximate Lipschitz-regularized value functions

$$\underline{Q}_t^{R;i+1}(x_{t-1}^{ik}, \xi_t; \|\cdot\|) := \min_{z_t \geq 0} \Big\{ \underline{Q}_t^{i+1}(z_t, \xi_t) + \sigma_t \|z_t - x_{t-1}^{ik}\| \Big\}.$$

These functions are proven to be Lipschitz continuous on $\mathbb{R}^{d_{a(n)}}$ with Lipschitz constant σ_t. Moreover, for sufficiently large σ_t for all $t \in [T]$, it can be shown that by considering the regularized problems still the original (MSLP) is solved to optimality

[68, 231]. However, choosing σ_t in a sufficient way is an open challenge in practice. If σ_t is chosen too small, it may even happen that the Lipschitz-regularized value function will be constant $-\infty$, given that the subproblem associated with $\underline{Q}_t^{i+1}(\cdot, \xi_t)$ for some fixed ξ_t is unbounded for any z_t.

18 No Block-diagonal Structure [relaxing Assumption 7]

A key element of dynamic programming methods is that in the multistage decision process only subsequent stages are linked in the constraints, as it allows one to express (MSLP) using the DPE (4)-(6). In the single-problem formulation (3) of (MSLP), this coincides with a block-diagonal structure, see Assumption 7.

In some cases, it may be relevant to include constraints spanning multiple stages instead. One example is the incorporation of emission quotas that are not allowed to be exceeded for a given time horizon in energy optimization problems [14, 178, 180].

In order to apply SDDP, the considered (MSLP) has to be reformulated to a problem satisfying Assumption 7. This can be achieved by aggregating stages [54], even though this changes the structure, solution and interpretability of (MSLP). An alternative approach is augmenting the state space. For emission quotas, for instance, instead of summing emissions over several stages and comparing them with the upper bound, at a given stage the remaining emission allowances can be considered as an additional state variable [178], see Sect. 9.

19 Infinite Horizon [relaxing Assumption 1]

So far, we considered problems (MSLP) with a finite time horizon $T < \infty$ (Assumption 1). In some practical applications, however, repeated decisions have to be modeled without a clear bound on the horizon. Considering such infinite-horizon problems is for instance common for Markov decision processes [19]. In such a case, to ensure that v^* is finite, a geometric discount factor $\delta < 1$ is introduced for the cost at each stage.

Since SDDP performs a forward and a backward pass through all stages in each iteration, it is not directly applicable to such problems, as no iteration would ever be completed. Therefore, often different solution methods are utilized in such a setting, see for example [9]. Still, recently there has been some focus on enhancing the SDDP idea to problems with infinite time horizon.

One approach, called Benders squared or B^2, is based on limiting each iteration of SDDP to a finite horizon of τ stages, but to dynamically increase τ per iteration, *e.g.*, by 1, until convergence is reached [145]. By presuming that the uncertainty occurs in the RHS and is not only stagewise independent, but also i.i.d. for all stages $t \in [T]$, almost sure convergence to an approximately optimal policy is assured. The reason is that under this special assumption, $\mathcal{Q}_t(\cdot)$ are the same for all stages, so cuts computed at stage t cannot only be incorporated at stage $t-1$, but at *all* stages [145].

A different option to adopt SDDP to infinite horizon problems exists if such problems possess some kind of periodical behavior. This idea is put forward by Shapiro and Ding [203]. Assume that for some period $m \in \mathbb{N}$, the distributions of ξ_t as well as the data c_t, W_t, h_t and T_{t-1} are the same for $t = \tau$ and $t = \tau + m$ for all $\tau = 2, \dots$ Then, under Assumption 9, the functions $\mathcal{Q}_t(\cdot)$ and $\mathcal{Q}_{t+m}(\cdot)$ are equivalent as well. This means that it is sufficient to derive cuts for $\mathcal{Q}_{t+m}(\cdot)$ at stages $t = 2, \dots, m+1$ in order to obtain valid cuts for all stages.

In the forward pass of SDDP, it is proposed to only consider a finite number of T stages starting from stage 1, with $T \geq m+1$ in order to determine at least one trial point for each of the differing expected value functions. In the case of $T > m+1$, multiple candidate trial points exist, at which cuts can be constructed in the backward pass. Before starting the backward pass, the used trial points can be chosen from such a candidate set randomly or by some heuristic.

For both approaches, B^2 and periodic SDDP, for discount factors δ close to 1, the influence of late stages on v^* may be substantial, and thus policy evaluation and upper bound determination may become very challenging and computationally costly. Still, Shapiro and Ding [203] propose some proxies based on some finite, but sufficiently large T. However, they do not provide a convergence proof.

A big advantage of SDDP for periodical problems is that it can also be applied to increase the performance for problems with a finite, but very large number of stages, given that they satisfy some notion of periodicity. The authors present an example where for a 60-month horizon, exploiting the periodical structure of the problem, instead of a 60-stage problem only a 13-stage problem has to be solved [203]. This can make even large problems amenable to SDDP and computationally tractable. It is also considered to mitigate the so-called *end-of-horizon effect*, which we discuss in Sect. 9.

On a different note, the policy graph approach introduced by Dowson [56] to model (MSLP) provides a natural extension to infinite-horizon problems, as it allows for cyclic graphs. Solving such problems, similarly to [145], relies on a discount factor and a truncation after a finite number of nodes in the graph. Then, approximate convergence can be proven.

20 Random Horizon [relaxing Assumption 1]

Another way to relax Assumption 1 is to assume that the horizon T is random. For simplicity, we discuss this aspect for the linear case only, even though it is presented in [89] for the convex case.

Consider (MSLP) from Sect. 2.3, satisfying Assumptions 2 to 8, but with T not being fixed. Instead, we take the following assumption:

Assumption 13. *The time horizon T is a discrete random variable taking values in $\{2,\dots,\overline{T}\}$ with known $\overline{T} \in \mathbb{N}$.*

Then, the horizon T induces the Bernoulli process $(\boldsymbol{D}_t)_{t\in[\overline{T}]}$ with realizations

$$D_t = \begin{cases} 0, & \text{if the optimization period ended at } t \text{ or before} \\ 1, & \text{otherwise,} \end{cases}$$

and therefore T can be written as

$$T = \min\left\{t \in [1,\overline{T}] \ : \ D_t = 0\right\}.$$

Under stagewise independence (Assumption 2), the decisions $\boldsymbol{x}_t(\cdot)$ are functions of $\boldsymbol{\xi}_t$, $\boldsymbol{D}_t$ and $\boldsymbol{D}_{t-1}$. In other words, $\boldsymbol{x}_t$ is $\overline{\mathscr{F}}_t$-measurable with $\overline{\mathscr{F}}_t$ the sigma-algebra $\sigma(\boldsymbol{\xi}_t, \boldsymbol{D}_j, j \leq t)$ [89].

As shown in [89], for (MSLP) with this type of random horizon, the following DPE equations can be derived. Importantly, the state space is augmented by D_{t-1}:

$$Q_t(x_{t-1}, D_t, D_{t-1}, \xi_t) = \min_{x_t \in \mathcal{X}_t(x_{t-1},\xi_t)} D_{t-1}\big(c_t(\xi_t)\big)^\top x_t + \mathcal{Q}_{t+1}(x_t, D_t),$$

where

$$\mathcal{Q}_{t+1}(x_t, D_t) = \mathbb{E}_{\boldsymbol{\xi}_{t+1}, \boldsymbol{D}_t | D_{t-1}} \left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1}) \right]$$

and $\mathcal{Q}_{\overline{T}+1}(x_{\overline{T}}, D_{\overline{T}}) \equiv 0$. For the first stage, we obtain

$$v^* = \min_{x_1 \in \mathcal{X}_1(x_0,\xi_1)} c_1^\top x_1 + \mathcal{Q}_2(x_1, D_1).$$

These DPE are the same as those that would be obtained for a problem with a fixed number of stages $\overline{T} \in \mathbb{N}$, but an objective function including the stagewise dependent stochastic process $(\boldsymbol{D}_t)_{t\in[\overline{T}]}$. As $(\boldsymbol{D}_t)_{t\in[\overline{T}]}$ can be modeled by an inhomogeneous Markov chain with two states, SDDP for processes with Markov chains can be applied [89], see Sect. 14.5.

21 Performance Improvements

Apart from extensions to different problem classes, a lot of research on SDDP has focused on improving its computational performance, because standard SDDP may suffer from various performance issues.

As shown in Sect. 4.2, its worst-case iteration complexity is exponential in the number of stages T and the dimension n_t of the state space, the latter being a well-known drawback of cutting-plane methods in general. Whereas SDDP is successfully applied to various large-scale problems in practice, see Sect. 9, with the optimality gap closed in reasonable time, especially for problems with a large state space it may empirically fail to converge. For instance, Ávila et al. [6] report instances for which the lower bounds $\underline{v}^i$ already start to stall at a gap of about 22%.

In addition to the high number of iterations required, also the computational effort in each iteration can become substantial, even if the number of subproblems solved per iteration has linear complexity, see Sect. 4.2. The reason is that with each iteration of SDDP, the subproblems (10) become larger, as additional cuts are included. This can increase the computational effort per iteration significantly, especially for problems (MSLP) which require many iterations to converge, and thus many cuts to be generated.

In this section, we give an overview on modifications of SDDP to address these issues and improve its performance.

21.1 Speeding up SDDP Iterations

We start with techniques attempting to speed up the SDDP iterations by reducing the computational effort.

21.1.1 Cut Elimination and Selection

As mentioned before, with each added cut, the subproblems (10) become larger, and thus potentially harder to solve. However, computational results indicate that SDDP tends to generate a large number of similar or redundant cutting planes, which do not contribute much to the approximation quality in later iterations [6, 205]. Therefore, the computational burden of SDDP may be reduced if only a subset of all cuts is taken into account. However, this requires careful elimination of cuts which are dominated and do not contribute to the solution process, or careful selection of decisive cuts, as otherwise the performance of SDDP may even become worse.

Cut Elimination. One way to reduce the number of cuts is to eliminate some cuts permanently. This can be done by repeatedly solving an auxiliary problem after a specified number of iterations, checking feasibility of the system

$$\begin{cases} \theta_{t+1} \leq \alpha_{t+1}^{\tilde{r}} + (\beta_{t+1}^{\tilde{r}})^\top x_t \\ \theta_{t+1} \geq \alpha_{t+1}^{r} + (\beta_{t+1}^{r})^\top x_t, \quad r \in \Gamma_{t+1} \setminus \{\tilde{r}\} \\ \quad x_t \in X_t \end{cases}$$

for each $\tilde{r} \in \Gamma_{t+1}$, where X_t is assumed to be a compact set [205].

If this system is infeasible, then the cut $\theta_{t+1} \geq \alpha_{t+1}^{\tilde{r}} + (\beta_{t+1}^{\tilde{r}})^\top x_t$ is redundant and can be eliminated. The drawback of this method is that the auxiliary problem has to be solved for all cuts in the system.

A different approach is to permanently store all cuts for each stage t, but only select a subset of those cuts to be considered in each iteration i. Selection techniques based on this approach are introduced in [8, 49].

Selecting Last Cuts. In this naive strategy, only the $\Gamma \in \mathbb{N}$ most recently added cuts are selected. Although on average, late cuts may provide a better approximation of $\mathcal{Q}_t(\cdot)$ than early ones, this strategy does not guarantee that all important cuts are considered.

Level of Dominance. This strategy is a heuristic in order to consider only non-dominated cuts, but avoid the computational effort of the above cut elimination approach. Using the most basic approach, only cuts are selected, which yield the highest function value at one of the trial solutions considered so far within the algorithm. This is called *Level 1 Dominance* [49]. A similar approach is proposed in [156], but there cuts are permanently removed if they are dominated.

Let x_t^ℓ be the trial solution corresponding to the ℓ-th cut, $\ell \in \Gamma_{t+1}$, and $\phi^r(x_t^\ell)$ the corresponding function value of cut r. Then, the values $v(\ell) := \max_{r \in \Gamma_{t+1}} \left\{\phi^r(x_t^\ell)\right\}$ and $r(\ell) := \arg\max_{r \in \Gamma_{t+1}} \left\{\phi^r(x_t^\ell)\right\}$ can be saved in a list and be updated every time a new cut is constructed. Similarly, a *Level H Dominance* strategy can be used, selecting the $H \in \mathbb{N}$ highest cuts for all trial solutions. Using this strategy, only previous trial points are taken into consideration, though. Therefore, cuts may be excluded which provide a significant benefit at not yet visited feasible states.

Another challenge is that this strategy draws a lot of resources to store all the required cut information – especially, since the number of visited trial points increases significantly in the course of SDDP. Memory requirements can even be relevant for Level 1, especially if the maximum function value at the trial solutions is attained by several cuts. As a resort, in [87], the *Limited Memory Level 1* strategy is introduced, selecting only the oldest of such cuts. In [8] a more general cut selection strategy is applied to SDDP and almost sure convergence is proven.

Dynamic Cut Selection. A dynamic, but also computationally more expensive strategy is to select cuts dynamically within the SDDP framework. In [49] it is proposed to remove all cuts at the beginning of each iteration. Then, for each stage t, each scenario k, and each function $\phi^r(\cdot), r \in \Gamma_{t+1}$, the forward pass subproblem (10) is solved. If the current cut yields the highest value at the obtained trial solution, it is added to the subproblem, and the next cut is considered. This way, only those cuts are selected that contribute to the optimal solution in the current iteration. On the other hand, the additional loop may slow down the convergence speed. The computational effort can be reduced by some additional heuristics [49].

A similar approach is considered in [31]. Here, cuts are iteratively added as long as they induce a substantial change in the current optimal value and up to a predefined maximum number of cuts. Instead of iterating over all cuts, in each step, the cut with the highest value at the current trial point is chosen as a candidate for selection.

Numerical results for sampling about 5,000 scenarios and computing 10,000 cuts in SDDP indicate that all cut selection techniques can significantly speed-up the classical SDDP method [49]. For example, the Level 1 strategy is reported to be ten times faster than SDDP without cut selection. For dynamic cut selection, the reported speed-up is much smaller. It is also shown that the cut selection strategies do not have a significant impact on the quality of the determined policies and bounds. In [8], Limited Memory Level 1 is identified as more efficient than pure Level 1.

21.1.2 Sampling Schemes

SDDP allows to use a variety of different sampling schemes which affect its computational performance.

Number of Forward Samples per Iteration. In standard SDDP, see Sect. 3, $|\mathcal{K}|$ scenarios are sampled in each iteration, with $|\mathcal{K}| \ll |\mathcal{S}|$ and $\mathcal{K} \subset \mathcal{S}$. Philpott and Guan even propose a method with only $|\mathcal{K}| = 1$ for all iterations [164]. This strategy may be particularly efficient in earlier iterations in order to obtain a rough approximation of $\mathcal{Q}_t(\cdot)$ fast without wasting too much effort in regions which are likely to be far from optimal. On the other hand, if the current policy is already reasonably good, it should be beneficial to generate more than one new cut per stage and iteration [49]. Moreover, if $|\mathcal{K}| = 1$, then it is not possible to apply a statistical stopping criterion, see Sect. 7.

Therefore, instead of fixing $|\mathcal{K}|$, a *scenario incrementation* strategy in which $|\mathcal{K}|$ is gradually increased is a promising approach [204]. It is tested in [49].

Subsampling Trial Points. In the *reduced sampling method* (ReSa) [102] the forward pass follows the same principle as in SDDP by sampling scenarios $\xi_t^k, k \in \mathcal{K}$, for $\mathcal{K} \subset \mathcal{S}$. In the backward pass, however, to reduce the number of subproblems to be solved, only a subsample $\tilde{\mathcal{K}} \subset \mathcal{K}$ is considered and only $|\tilde{\mathcal{K}}|$ curs are generated. Considering more samples in the forward pass without additional effort in the backward pass is helpful to compute accurate statistical upper bounds in an efficient way.

A similar approach is applied in the *abridged nested decomposition* (AND) method [53]. It is claimed that SDDP is not well-designed for bushier scenario trees with large values q_t because solving $|\mathcal{K}|q_t$ subproblems per stage may quickly become computationally costly, especially for large $|\mathcal{K}|$. On the other hand, a large $|\mathcal{K}|$ may be required to get reliable statistical upper bounds and to incorporate information on sufficiently many scenarios in the trajectories $(x_t^{ik})_{k \in \mathcal{K}}$. As a remedy, an alternative sampling scheme is proposed. In the forward pass, on each stage $t \in [T]$ a set $\mathcal{K}_t$ of realizations is sampled and trial points x_t^{ik} are computed. On stage $t+1$, however, only a few *branching values* are used as parameters (in the forward and backward pass), which can either be sampled from or be a convex combination of all $x_t^{ik}, k \in \mathcal{K}_t$. The latter idea allows to compute trial points which contain information on a large set of scenarios, while keeping the computational effort in the backward pass at a minimum. The main drawback of AND is that the special structure of the forward pass allows no direct estimate of an upper bound [102].

Sampling in the Cut Generation Process. The computational effort in the backward pass can be reduced if the subproblems (10) are not solved for all noise terms ξ_{tj}, $j = 1, \dots, q_t$, in each iteration, but only for a subset. The remaining elements that are required to compute a valid cut can then be used from previous iterations where the corresponding noise ξ_{tj} was sampled.

Even more, if the uncertainty is restricted to the RHS h_t of (MSLP), then the dual feasible set does not depend on it. Therefore, optimal dual multipliers which correspond to dual extreme points can be re-used between different realizations $j = 1, \dots, q_t$. This allows for the following procedure: Assume that in each iteration i, for each stage $t \in [T]$ only one noise term $\hat{\xi}_t^i$ is sampled and used to compute optimal dual multipliers $\hat{\pi}_t^i$ and (scenario-specific) cut intercepts $\hat{\alpha}_t^i$ as in (21.2.1). For each stage $t = 2, \dots, T$, all dual multipliers and intercepts obtained up to iteration i are then stored in a set $\mathcal{D}_t^i$ defined by $\mathcal{D}_t^i = \mathcal{D}_t^{i-1} \cup \left\{ \left(\hat{\pi}_t^i, \hat{\alpha}_t^i, \hat{\xi}_t^i \right) \right\}$.

For any $\xi_{tj}, j = 1, \dots, q_t$, and a given incumbent x_{t-1}^i, the dual multipliers used to compute a new cut can then be determined as

$$\left(\hat{\pi}_t^{ij}, \hat{\alpha}_t^{ij}, \hat{\xi}_t^{ij} \right) = \underset{(\hat{\pi}_t, \hat{\alpha}_t, \hat{\xi}_t) \in \mathcal{D}_t^i}{\arg\max} \left\{ \hat{\alpha}_t - \hat{\pi}_t^\top T_{t-1} x_{t-1}^i + \hat{\pi}_t^\top \left(h_t(\xi_{tj}) - h_t(\hat{\xi}_t) \right) \right\}.$$

Hence, not necessarily optimal dual multipliers for ξ_{tj} are used, but the ones in $\mathcal{D}_t^i$ providing the best approximation for realization ξ_{tj} at x_{t-1}^i.

Let $\pi_{tj}^i = \hat{\pi}_t^{ij}$ and $\alpha_{tj}^i = \hat{\alpha}_t^{ij} + (\hat{\pi}_t^{ij})^\top \left(h_t(\xi_{tj}) - h_t(\hat{\xi}_t^j) \right)$ for all $j = 1, \dots, q_t$. Then, a cut can be defined by using subgradient formula (19) and taking expectations as in formula (16). Note that our description slightly differs from the presentation in the literature, as we adapted it to our cut formulas in Sect. 3.3.

This idea for the cut generation process is used in two algorithms related, which mainly differ by *when* cuts are constructed. The CUPPS (*convergent cutting-plane and partial-sampling*) method [38] only contains a forward pass, in which both trial points are computed and cuts are generated using some sample $\xi_t^{k'}$. It has the drawback that the obtained cuts are not necessarily tight. First, the dual multipliers obtained from formula (21.1.2) are not necessarily optimal for all $j = 1, \dots, q_t$. Second, no backward pass is used, and thus new information in form of cuts for stage $t+1$ are not taken into account when deriving a new cut for stage t.

The *dynamic outer approximation sampling algorithm* (DOASA) [164] contains a forward pass and a backward pass. In the former, a trajectory of trial points $(x_t^{ik})_{k \in \mathcal{K}}$ is computed as in SDDP (note that in [164] $|\mathcal{K}| = 1$ is chosen, but this is not mandatory). In the backward pass, cuts are constructed using a backward sample $\xi_t^{k'}$ and formula (21.1.2). It is proven that this generalization of SDDP also exhibits almost sure finite convergence [164].

21.1.3 Inexact SDDP

Recall Lemma 3.3 (b), stating that the cuts generated in the backward pass of SDDP are *tight* for $\underline{\mathcal{Q}}_t^{i+1}(\cdot)$ at the incumbent x_{t-1}^{ik}. This result is premised on using optimal dual multipliers in the cut formula, *i.e.*, solving the LP subproblem or its dual to global optimality (*exact* solution). Whereas such an exact solution is the standard assumption in the literature on SDDP, computationally it may be more efficient to solve subproblems only approximately, especially early in the solution process when the cut approximations are suboptimal anyway [88].

We first introduce the notion of inexact cuts.

Definition 21.1 (ε-inexact cut)**.** *For any $t = 2, \dots, T$, $\varepsilon > 0$ and a trial point x_{t-1}^{ik}, let $\phi_t : \mathbb{R}^{d_{a(n)}} \to \mathbb{R}$ be an affine function satisfying*

$$\mathcal{Q}_t(x_{t-1}) \geq \underline{\mathcal{Q}}_t^{i+1}(x_{t-1}) \geq \phi_t(x_{t-1}) \quad \textit{(validity)}$$

for all $x_{t-1} \in \mathcal{X}_{t-1}$ and

$$\underline{\mathcal{Q}}_t^{i+1}(x_{t-1}^{ik}) - \phi_t(x_{t-1}^{ik}) \leq \varepsilon \quad \textit{(ε-tightness).}$$

Then, $\phi_t(\cdot)$ defines an ε-inexact cut *at x_{t-1}^{ik} [88].*

Importantly, inexact cuts still yield valid lower approximations of $\mathcal{Q}_t(\cdot)$ for all $t = 2, \dots, T$. We now address how inexact cuts can be determined.

Linear Problems. For any iteration i in SDDP, any $t = 2, \dots, T$ and any trial point x_{t-1}^{ik}, consider the linear subproblem (10). For simplicity, we assume that $X_t = \{x_t \in \mathbb{R}^{n_t} \ : \ x_t \geq 0\}$.

For some $\varepsilon > 0$, let π_{tjk}^i be an ε-optimal feasible solution for the dual problem of (10) given ξ_{tj} and let θ_{tjk}^i be the corresponding dual objective value for $j = 1, \dots, q_t$. Then, analogously to Sect. 3.3, an ε-inexact cut can be defined by [88]

$$\mathcal{Q}_t(x_{t-1}) \geq \phi_{tk}^i(x_{t-1}) := \alpha_{tk}^i + (\beta_{tk}^i)^\top x_{t-1},$$

with intercept and subgradient defined by

$$\alpha_{tk}^i = \sum_{j=1}^{q_t} p_{tj}\big(\theta_{tjk}^i - (\beta_{tkj}^i)^\top x_{t-1}^{ik}\big),$$
$$\beta_{tk}^i = -\sum_{j=1}^{q_t} p_{tj}(\pi_{tkj}^i)^\top T_{t-1,j}.$$

Nonlinear Differentiable Problems. Consider a multistage stochastic convex program (MSCP) as introduced in Sect. 15, that is, satisfying Assumptions 10 to 12. Moreover, recall the definitions of the Lagrangian function (111), the dual function (112) and the Lagrangian dual problem (113).

Then, an ε-inexact cut can be derived using a pair of approximate primal-dual solutions as follows [88]. Let $\bar{x}_{tj}$ be an ε-optimal feasible primal solution for problem (108) given some noise realization $\xi_{tj}, j = 1, \dots, q_t$, and some trial point $\bar{x}_{t-1}$, and let $\bar{\pi}_{tj}$ be an ε-optimal feasible solution for the corresponding Lagrangian dual (113).

We define

$$\eta(\varepsilon) := \ell(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}\xi_{tj}) := \max_{x_t \in X_t} \nabla_{x_t} L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj})^\top (\bar{x}_{tj} - x_t). \tag{119}$$

Assume that $f_t(x_t, \xi_{tj})$ takes finite values for all $x_t \in X_t$ and that the term in (119) is finite. Then, an ε-inexact cut can be defined by

$$\mathcal{Q}_t(x_{t-1}) \geq \phi_{tk}^i(x_{t-1}) := \alpha_{tk}^i + (\beta_{tk}^i)^\top x_{t-1},$$

with intercept and subgradient defined by

$$\alpha_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \big(L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj}) - \eta(\varepsilon) - (\beta_{tkj}^i)^\top x_{t-1}^{ik} \big),$$
$$\beta_{tk}^i = \sum_{j=1}^{q_t} p_{tj} \nabla_{x_{t-1}} L_{t,C}(\bar{\pi}_{tj}; \bar{x}_{t-1}, \bar{x}_{tj}, \xi_{tj}).$$

We refer to [88] for a convergence analysis of SDDP using inexact cuts, both for the linear and the nonlinear convex case. In particular, it is shown that the obtained dual solutions are almost surely bounded and that the error terms $\eta(\varepsilon_t^i)$ vanish as i approaches $+\infty$.

Non-differentiable Problems. Using SDDP with inexact cuts is generalized to non-differentiable problems in [91]. In this paper, inexact cuts are derived using two different approaches. In the first approach, it is assumed that the objective and constraint functions have saddle-point representations. The second approach is more general, but requires the introduction of additional variables and constraints.

More precisely, consider a multistage stochastic convex program (MSCP) as introduced in Sect. 15 and assume that it is satisfying Assumptions 10 and 11 except for the differentiability properties. Using a local copy z_t of the state variable x_{t-1}, the approximate value functions can be reformulated as

$$Q_{t,C}(x_{t-1}, \xi_t) := \begin{cases} \min\limits_{x_t, z_t} & f_t(x_t, \xi_t) + \mathcal{Q}_{t+1,C}(x_t) \\ \text{s.t.} & g_t(z_t, x_t, \xi_t) \leq 0 \\ & x_t \in X_t \\ & x_{t-1} = z_t. \end{cases} \tag{120}$$

Assume that this modified subproblem satisfies a slater condition analogous to Assumption 12. Additionally, consider the Lagrangian dual problem

$$\max_{\pi_t} \mathcal{L}_t(\pi_t; x_{t-1}, \xi_t). \tag{121}$$

with dual function

$$\mathcal{L}_{t,C}(\pi_t; x_{t-1}, \xi_t) = \begin{cases} \min\limits_{x_t \in X_t} & f_t(x_t, \xi_t) + \mathcal{Q}_{t+1,C}(x_t) + \pi_t^\top (x_{t-1} - z_t) \\ \text{s.t.} & g_t(z_t, x_t, \xi_t) \leq 0 \\ & x_t \in X_t, \end{cases}$$

which is obtained by relaxing the copy constraint.

Given a trial point $\bar{x}_{t-1}$ and a noise realization $\xi_{tj}, j = 1, \ldots, q_t$, let $\bar{x}_{tj}$ denote an ε_P-optimal feasible solution of problem (120) and let $\bar{\pi}_{tj}$ be an ε_D-optimal feasible solution of problem (121). Then, an $(\varepsilon_P + \varepsilon_D)$-inexact cut is defined by function

$$\phi_{tk}^i(x_{t-1}) := \sum_{j=1}^{q_t} p_{tj} \Big(f_t(\bar{x}_{tj}, \xi_{tj}) - (\varepsilon_P + \varepsilon_D) + \bar{\pi}_{tj}^\top (x_{t-1} - \bar{x}_{t-1}) \Big).$$

For more details and a convergence analysis we refer to [91].

21.2 Reducing the Number of SDDP Iterations

We now consider techniques with the attempt to reduce the required number of iterations of SDDP until convergence is reached.

21.2.1 Multi-cut SDDP

In the backward pass of SDDP, for any $t \in [T]$ and any $x_{t-1}^{ik}, k \in \mathcal{K}$, subproblems (10) are solved for all noise realizations ξ_{tj}, $j = 1, \dots, q_t$. By taking expected values, a cut (17) is derived. Such cuts are then incorporated into the stage-$(t-1)$ subproblem using a single variable $\theta_t \in \mathbb{R}$ by

$$\phi_{tk}^i(x_{t-1}) = (\beta_{tk}^i)^\top x_{t-1} + \alpha_{tk}^i \leq \theta_t,$$

see Sect. 3.3. This is referred to as a *single-cut* approach.

A different approach, called *multi-cut*, that is well-studied for (nested) Benders decomposition [27, 74, 144], is to not aggregate the dual information, but to generate a separate cut for each noise realization $\xi_{tj}, j = 1, \dots, q_t$. This requires to introduce variables $\theta_{t,\ell}$ and cut approximations $\mathfrak{Q}_{t+1,\ell}^{i+1}(\cdot)$ for all $\ell = 1, \dots, q_t$ in the stage-t subproblem. In this case, we obtain cuts

$$\phi_{tkj}^i(x_{t-1}) := (\beta_{tkj}^i)^\top x_{t-1} + \alpha_{tkj}^i \leq Q_t(x_{t-1}, \xi_{tj}), \qquad j = 1, \dots, q_t,$$

where, analogously to the derivation in Sect. 3.3, β_{tkj}^i denotes a subgradient of $\underline{Q}_t^{i+1}(\cdot, \xi_{tj})$ at x_{t-1}^{ik} for $k \in \mathcal{K}, j = 1, \dots, q_t$, and α_{tkj}^i is defined by

$$\alpha_{tkj}^i := \underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_t) - (\beta_{tkj}^i)^\top x_{t-1}^{ik}.$$

The expectation is then taken in the objective function instead of the cut formula:

$$\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj}) = \min_{x_t \in \mathcal{X}_t(x_{t-1}, \xi_t)} \big(c_t(\xi_{tj})\big)^\top x_t + \sum_{\ell=1}^{q_{t+1}} p_{t+1,\ell} \mathfrak{Q}_{t+1,\ell}^{i+1}(x_t).$$

This way, more specific information about the value functions is incorporated into the subproblems, hopefully leading to fewer iterations. Moreover, it can be shown that multi-cut SDDP has the same convergence properties as SDDP [8]. On the downside, the number of decision variables and cuts grows significantly compared to the single-cut approach, especially if q_t is large, which increases the computational effort for each iteration. Therefore, so far multi-cut SDDP has rarely been considered in the literature. It should be most promising when q_t is only of moderate size. For the two-stage case, a rule of thumb is that a single-cut approach should be preferred if the number of realizations is considerably larger than the number of first-stage constraints [26]. Note that in principle also a trade-off between single-cut and multi-cut is possible by partially aggregating cuts [23, 28]. Another approach to reduce the computational burden of multi-cut SDDP is to combine it with cut selection strategies, see Sect. 21.1.1, as proposed in [8].

We return to Example 3.4 to illustrate the multi-cut approach.

Example 21.2. *(Continuation of Example 3.4) Using multi-cut SDDP, at stage 3, instead of* $\mathcal{Q}_3(\cdot)$*, the functions* $Q_3(\cdot, \xi_3)$ *are separately approximated by cuts for* $\xi_3 \in \{1, 2, 4\}$*. These value functions are displayed in Figure 15. Each of them consists of* *only two linear pieces, so two cuts are required to represent them exactly. In contrast, $\mathcal{Q}_3(\cdot)$ consists of four linear segments. Therefore, multi-cut SDDP should need less iterations than single-cut SDDP to achieve convergence.*

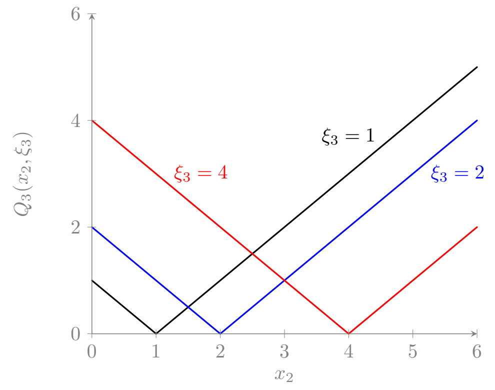


Figure 15: Stage-3 value functions for Example 3.4.

21.2.2 Batch Learning and Experience Replay

While SDDP is used in stochastic programming, dynamic programming or optimal control, its methodology also shares some characteristics with Q-learning algorithms, which are studied in reinforcement learning, see Remark 3.1. This can be exploited by translating established performance enhancing techniques from reinforcement learning to SDDP.

As one such technique, Ávila et al. [6] propose to use a *batch learning* technique called *experience replay* in SDDP. The motivation of this is the following: In SDDP, the cut approximations $\mathfrak{Q}_t(\cdot)$ of the expected value functions $\mathcal{Q}_t(\cdot)$ are generated recursively in a backward pass through the stages $t = T, \dots, 2$. This means that approximation errors at later stages are propagated to earlier stages by means of the cut approximations $\mathfrak{Q}_t(\cdot)$, which then leads to loose cuts at these earlier stages and so on. However, this implies that errors are accumulated at early stages. The authors identify this as a driver for the slow convergence of SDDP, as it favors over-exploring of suboptimal regions and the generation of redundant cuts throughout the iterations.

Experience replay addresses this issue by revisiting previous trial points x_t^i and updating the cut approximations $\mathfrak{Q}_t(\cdot)$ at these points. This seems counterintuitive at first glance because cuts are generated at already visited points instead of improving the approximation of $\mathcal{Q}_t(\cdot)$ at regions of $\mathcal{X}_t$ that have not been visited yet. However, by taking into account all the information currently available to update $\mathfrak{Q}_t(\cdot)$ at x_t^i, it avoids that on earlier stages $\tau < t$ unnecessarily poor approximations of $\mathcal{Q}_t(\cdot)$ at x_t^i are used for several more iterations.

More precisely, the proposed SDDP method works as follows. A predefined number of iterations of standard SDDP are executed and the corresponding trial points x_t^i are stored in a replay memory M_t for all $t \in [T-1]$. When the sizes of the replay memories reach a predefined cardinality Z, then the experience replay step is initiated. This step performs a backward pass through the stages $t = T-1, \dots, 2$. For each stage t, first, a batch $B_t \subseteq M_t$ of trial points is selected from the replay memory (also a full batch

$B_t = M_t$ is possible). For each trial point $\tilde{x}_t^\ell$ from this batch, with $\ell = 1, \dots, |B_t|$, the previously generated cut is removed from $\mathfrak{Q}_{t+1}^{i+1}(\cdot)$ and a new cut is constructed by solving the associated subproblems (10) (including the experience replay updates from following stages) for $\tilde{x}_t^\ell$. With these cuts, $\mathfrak{Q}_{t+1}^{i}(\cdot)$ is updated and then, the previous stage is explored.

It is shown that experience replay manages to improve the convergence behavior of SDDP, and also the out-of-sample performance of the obtained policies [6]. However, experience replay comes at an increased computational effort, as every Z iterations an additional backward pass solving $q_t|B_t|$ subproblems for each stage $t = T, \dots, 2$ has to be performed. For full batches, this adds up to $q_t|\mathcal{K}|Z$ LPs per stage. For this reason, the authors suggest to parallelize both standard SDDP iterations as well as the experience replay. They report computational results which indicate that batch learning is better exploiting parallelism than standard SDDP.

21.2.3 Regularization

As Kelley's cutting-plane method [111, 146], SDDP exhibits an iteration complexity which is exponential in the dimension n_t of the state variables, see Sect. 4.2. An unfavorable characteristic of cutting-plane methods, and also of SDDP, in this regard is *zig-zagging* behavior. This means that trial points x_t^i and x_t^{i+1} computed in subsequent iterations can be located far away from each other in different regions of the state space, and that with each new cut the minimum of the subproblems (10) is again attained in the respective other region. In particular, this implies that these regions of $\mathcal{X}_t$ experience very tight, but almost redundant approximations $\mathfrak{Q}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$, while other regions are not properly explored and thus the approximation quality at the true optimum improves very slowly.

In convex and nonsmooth optimization, regularization techniques called *bundle methods* are shown to entail faster convergence than classical cutting-plane methods [122], as they mitigate zig-zagging by stabilizing subsequent trial points around a *stability center* (also called incumbent). Hence, it looks promising to translate these regularization techniques to SDDP.

A common regularization approach, which is predominantly used in two-stage stochastic programming [191, 194], is convex quadratic regularization. Here, some quadratic deviation of x_t from a stability center $\widehat{x}_t$ is penalized in the objective function for stabilization. An application of quadratic regularization to SDDP is not straightforward because using a separate stability center for each scenario $s \in \mathcal{S}$ is computationally infeasible due to the exponential growth of $|\mathcal{S}|$ in T [5].

Therefore, Asamov and Powell [5] propose a regularization technique for linear problems, in which stability centers are considered part of the state variable, and thus are the same for all realizations of $\xi_{tj}, j = 1, \dots, q_t$. Then, in the forward pass the objective function is modified to

$$c_t^\top x_t + \mathfrak{Q}_{t+1}^i(x_t) + \frac{\gamma^i}{2}(x_t - \widehat{x}_t^{i-1})^\top H_t (x_t - \widehat{x}_t^{i-1}), \tag{122}$$

with a positive semidefinite matrix H_t and some sequence $(\gamma^i)_{i \in \mathbb{N}}$ satisfying $\gamma^i \geq 0$ for all i and $\lim_{i \to \infty} \gamma^i = 0$. The stability centers $\widehat{x}_t^{i-1}$ are chosen as the previous forward pass solution, *i.e.*, the solution is stabilized around a "known" region of the domain of $\mathcal{Q}_t(\cdot)$. This idea is generalized to nonlinear problems and improved in [90] by considering weighted averages of several previous forward pass solutions.

Using objective (122), a convex, continuous and linearly constrained quadratic programming problem has to be solved in each forward pass step of SDDP, hopefully, reducing the required number of iterations. Importantly, only the forward pass of SDDP is changed, while the backward pass remains the same. In particular, only LPs have to be solved in the backward pass. As the cuts are still finite (see Lemma 3.3), almost sure finite convergence is assured. In computational tests, it is shown that this method exhibits faster convergence than SDDP, in particular for a high state dimension n_t [5]. This speed-up is especially important for regularized DDP, see the numerical experiments in [90]. DDP (Dual Dynamic Programming) is the corresponding deterministic counterpart of SDDP (when ξ_t is deterministic for all $t \in [T]$).

Whereas the above approach stabilizes the solution around a "known" region of the domain of $\mathcal{Q}_t(\cdot)$, in a sampling setting, it is not clear whether this is always beneficial. For the current sample ξ_t^k a region may be identified and used for stabilization, which is no appropriate indicator for all $\xi_{tj}, j = 1, \ldots, q_t$. Additionally, as pointed out in [226], the condition $\lim_{i\to\infty} \gamma^i = 0$ may evoke that the regularization is diminished and the proposed method in [5] reduces to standard SDDP *before* convergence is obtained, although regularization may be particularly important close to the optimal solution. Therefore, this is claimed to be detrimental to convergence speed [226].

Van Ackooij et al. [226] also address that convergence of proximal bundle methods usually requires the stability centers to be feasible, which is not guaranteed for SDDP subproblems where the feasible set changes with x_{t-1}^i. Therefore, they propose to combine SDDP with a level bundle method, which does not face this requirement.

For stage t and scenario ξ_t^k, trial solutions x_t^{ik} are obtained by solving

$$\begin{cases} \min\limits_{x_t} & \psi_t(x_t) \\ \text{s.t.} & x_t \in \mathbb{X}_t(x_{t-1}^{ik}; \ell_t) \end{cases} \tag{123}$$

with $\psi_t(x_t) : \mathbb{R}^{n_t} \to \mathbb{R}$ a given convex function, *e.g.*, $\psi_t(x_t) := x_t^\top x_t$, and

$$\mathbb{X}_t(x_{t-1}^{ik}; \ell_t) := \begin{cases} \operatorname*{arg\,min}\limits_{x_t \geq 0} & \max\left\{c_t^\top x_t + \mathfrak{Q}_{t+1}^i(x_t), \ell_t\right\} \\ \quad\text{s.t.} & W_t x_t = h_t - T_{t-1} x_{t-1}^{ik}. \end{cases} \tag{124}$$

If the maximum in (124) is attained by the first term, then x_t^{ik} obtained by solving (123) is an ordinary SDDP trial point, referred to as a *normal iterate*. Otherwise, problem (123) reduces to a typical level bundle method subproblem, yielding a regularized *level iterate* x_t^{ik}.

The determination of a good level ℓ_t and of an efficient regularization for SDDP are still open questions, and heuristics are proposed in [226] to choose ℓ_t.

An alternative stabilization approach is proposed in [15] based on the concept of Chebyshev centers of polyhedrons. Here, in the forward pass of SDDP, the subproblems (10) are modified such that the computed trial states are defined as Chebyshev centers of the polyhedrons given by previously constructed cuts and an appropriate upper bound. It can be shown that this approach is equivalent to modifying the cut formula to

$$-(\beta_{t+1}^r)^\top x_t + \theta_{t+1} \geq \alpha_{t+1}^r + \bar{\sigma}_t \|(1, c_t + \beta_{t+1}^r)\|, \quad r \in \Gamma_{t+1}. \tag{125}$$

The authors use the Euclidean norm $\|\cdot\|_2$ in (125), however, different choices are possible as well.

Geometrically, the additional term in (125) changes the cut intercept, thus lifting the cut. For $\bar{\sigma}_t = 0$, the usual SDDP trial point x_t^i is determined, whereas for $\bar{\sigma}_t > 0$ an offset in the objective compared to the standard SDDP subproblem is considered, yielding a different iterate. To actually improve the performance of SDDP, choosing $\bar{\sigma}_t$ appropriately is crucial, yet not trivial. Adversely, if $\bar{\sigma}_t$ is chosen too large, basically any feasible point can become the new trial solution. Moreover, to ensure convergence, it has to be ensured that $\bar{\sigma}_t$ converges to zero in the course of the algorithm. In [15] heuristics are used to determine $\bar{\sigma}_t$, but it is not clear whether they guarantee performance gains for SDDP.

21.3 Parallelization

The performance of SDDP cannot only be improved by modifications of the algorithm itself, but also by its implementation and computational execution. Since several computational steps in SDDP are independent of each other, a performance improvement can be achieved by parallelization.

Different parallelization strategies have been proposed for SDDP. They can be classified with respect to how the workload is distributed among different processors and how the processors are synchronized. Based on this observation, Ávila et al. [6] present a taxonomy of parallelization strategies, which we follow in this section.

Parallelization by Scenario. This is the predominant parallelization strategy for SDDP in the literature. Mostly, a *synchronized* version is proposed. In the forward pass, for all $t \in [T]$, the subproblems (10) are solved for $|\mathcal{K}|$ different scenarios, which are sampled independently. The uncertain data ξ_t^k and the trial solutions x_{t-1}^k in each of those problems do only depend on scenario k. Therefore, the different scenarios $\xi^k, k \in \mathcal{K}$, can be assigned to different processors. Assuming P different processors, each processor is assigned $\frac{P}{|\mathcal{K}|}$ scenarios and solves all corresponding subproblems. A master process is then used to aggregate the objective values and compute the upper bound estimate (21). This means that there is a synchronization point for all processors at the end of the forward pass.

In the backward pass, a similar approach is followed. The subproblems are again distributed among the processors by scenarios, in such way that for a specific stage $t = T, \ldots, 2$ and a scenario-based trial point x_{t-1}^k, the subproblems for all noise realizations $\xi_{tj}, j = 1, \ldots, q_t$, are solved by the same processor. Evenly distributing the problems between processors, this way each processor solves $\frac{P}{|\mathcal{K}|} q_t$ subproblems. However, it is also possible to let the master process assign new scenarios to processes once they become idle instead of using a fixed assignment scheme [166].

After solving all associated subproblems, each processor then generates a cut for $\mathcal{Q}_t(\cdot)$ and sends it to the master process. When cut generation is finished for all $k \in \mathcal{K}$, the processors are synchronized so that all of them can proceed with the same set of cuts on stage $t-1$. As stated in [99], this synchronization can be partially relaxed to avoid waiting for single slow processors. Instead, the master process can assign stage-$(t-1)$ subproblems to available processors even if not all cuts have been generated for stage t yet. Numerical results show that such partial relaxation can improve the computational performance of SDDP. However, the number of cuts to wait for to achieve an optimal trade-off between faster iterations and better approximation of $\mathcal{Q}_t(\cdot)$ is problem-dependent.

Even more, an *asynchronous* approach can be used where processors immediately get back to stage $t-1$ after generating their cuts at stage t, using all cuts currently available without waiting for other processes to finish [57].

A major shortcoming of parallelization by scenario is that using more processors becomes more beneficial the more scenarios $|\mathcal{K}|$ are sampled in the forward pass. However, as discussed in Sect. 21.1.2, it is often favorable to only consider one or a few scenarios per iteration, especially in earlier iterations. Choosing large $|\mathcal{K}|$ may lead to the accumulation of similar trial points and the generation of redundant cuts [6]. Therefore, exploiting the potential performance gains of additional processors may wrongly incentivize to sample more scenarios than reasonable, thus not accelerating but slowing down the solution process. Additionally, Ávila et al. [6] report computational results indicating that (synchronized) parallelization by scenario scales poorly when increasing the number of samples $|\mathcal{K}|$ due to the combination of long waiting times between processors and low quality cuts.

Parallelization by Node. Using parallelization by node, the strategy is to draw only one or a few samples in the forward pass, as this is often computationally preferable. Then, the forward pass is not necessarily parallelized. In the backward pass, the work is distributed among the processors by nodes of the recombining tree (cf. Sect. 2.1). That means that even for the same $k \in \mathcal{K}$ and the associated trial point x_{t-1}^k, the subproblems (10) for different realizations $\xi_{tj}, j = 1, \dots, q_t$, may be solved by different processors. The processors are synchronized at each stage to generate aggregated cuts (given that a single-cut approach is used).

In [6], the authors report clear computational benefits using parallelization by node compared to parallelization by scenario, and also better scaling properties. However, these results require that the processors can access a shared memory, otherwise the computational overhead is too large. Another drawback is that distributing subproblems for different ξ_{tj}, but the same x_{t-1}^{ik} among different scenarios prevents the exploitation of warm starting techniques.

Parallelization by node can also be used in an asynchronous way, as proposed by Machado et al. [136] in their *asynchronous SDDP* method. In this method, the subproblems of all stages $t = 1, \dots, T$ are solved simultaneously. More precisely, in each *step*, for all stages $t = 1, \dots, T$ and scenarios $k \in \mathcal{K}$, the subproblems for all realizations $\xi_{tj}, j = 1, \dots, q_t$, are solved. Once a processor is finished, it constructs a new cut for $\mathcal{Q}_t(\cdot)$ using all available information. If a required processor has not finished yet, multipliers π_{tkj} from previous steps are re-used. The generated cut can then be incorporated in stage $t-1$ in the next step. Additionally, each processor generates a new trial point which can be used at stage t in the next step. In contrast to SDDP iterations, this approach requires several steps to propagate information through all stages. Therefore, an ordinary forward pass can only be observed implicitly over several stages. This has to be considered in the computation of upper bounds.

Independent of the applied strategy, parallelizing SDDP in practice comes with considerable challenges, such as communication overhead, problem-dependent performance and lack of reproducibility of results. Therefore, its potential to speed up SDDP in general is naturally limited [6].

21.4 Aggregation Techniques

Aggregating information in (MSLP) is another tool with potential to speed up the SDDP solution process.

One approach is to aggregate the variables and constraints of several time periods in a single stage, thus solving a problem with a smaller horizon T. This is straightforward for NBD [54], where each node of the aggregated problem is a subtree of the original scenario tree, even though only few time periods can be aggregated to keep the subproblems tractable. However, it cannot be directly generalized to the sampling and stagewise independent setting in SDDP. The main issue is that it is difficult to model the uncertainty appropriately, without violating non-anticipativity [54].

An alternative approach is to aggregate realizations of $\boldsymbol{\xi}_t$ on each stage (or a subset of stages) [207]. To this end, for some stage t, the support Ξ_t is partitioned into clusters $C_t^\ell, \ell = 1, \ldots, L_t$, with $L_t \in \mathbb{N}$. Instead of solving subproblems associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \xi_{tj})$ for all $j = 1, \ldots, q_t$ in the backward pass of SDDP, subproblems associated with $\underline{Q}_t^{i+1}(x_{t-1}^{ik}, \bar{\xi}_t^\ell)$ are solved for clusters $\ell = 1, \ldots, L_t$, with $\bar{\xi}_t^\ell := \sum_{j \in C_t^\ell} \frac{p_{tj}}{\bar{p}_t^\ell} \xi_{tj}$, and $\bar{p}_t^\ell$ the probability of cluster C_t^ℓ. This should be beneficial in early iterations where policies are still far away from optimal and a fine information structure unnecessarily slows down the solution process.

Using subgradients and intercepts associated with clusters $C_t^\ell, \ell = 1, \ldots, L_t$, *coarse cuts* can be generated for $\mathcal{Q}_t(\cdot)$. Given that W_t and c_t are deterministic, these cuts are valid underestimators for $\mathcal{Q}_t(\cdot)$ by Jensen's inequality [207]. They are not guaranteed to be tight, though.

The authors in [207] discuss several different refinement strategies, such as refinements within the SDDP backward pass (the partition at stage t is refined as soon as a coarse cut does not improve the approximation of $\mathcal{Q}_t(\cdot)$ at the trial point x_{t-1}^{ik}) or refinements outside of SDDP. In the latter case, SDDP is performed on a coarse recombining tree, which is iteratively refined once the algorithm has stopped. Computational results show that this latter approach performs significantly better than the first one due to less computational overhead. However, identifying *when* SDDP should be best stopped to perform a refinement remains a challenging task.

22 Outlook

In this tutorial-type review, we give an overview on the motivation, theory, strengths and weaknesses, extensions and applications of SDDP.

While many proposals have been made in the last 30 years on how to extend SDDP and on how to improve its performance, there still remain open research questions, leaving room for future improvement. Among the most crucial topics are the following.

1. *Stopping.* To this date, in many applications SDDP is stopped heuristically, *e.g.*, based on a fixed number of iterations or stabilization of lower bounds, which leaves the task to define a reasonable stopping criterion to the user. Recently, there has been some pioneering work on developing deterministic upper bounding techniques and stopping criterions, but these are still limited, as they require significant computational effort.

2. *Upper bounds in risk-averse SDDP.* Developing efficient upper bounding techniques is especially relevant to risk-averse variants of SDDP, where the commonly used nested risk measures do not allow for employment of their pendants from risk-neutral SDDP. Lately, different risk measures have been proposed, which avoid this issue. However, such risk measures usually hamper interpretability. Therefore, it can still be regarded an open question how risk should be optimally measured in SDDP in order to obtain a computationally tractable problem and at the same time to properly reflect the true risk preferences of a decision-maker.

3. *Distributionally robust SDDP.* Recently, the consideration of distributional uncertainty in SDDP has gained more interest. However, while distributionally robust optimization is a flourishing research area, incorporating it into SDDP is still in its early stages, with potential for further improvements.

4. *Non-convex extensions.* In many applications, nonlinear functions or integer variables are required to appropriately model the problem at hand. As the (expected) value functions become non-convex in this case, traditional cutting-plane techniques fail to approximate them correctly. Starting with SDDiP, recently, there has been a trend to extend the NBD and SDDP frameworks to non-convex problems. Lagrangian-type cuts, which are possibly non-convex, show theoretical potential in approximating non-convex functions. However, their construction is computationally costly and subject to rather strong technical assumptions, such that especially large-scale non-convex problems remain computationally intractable. Consequently, in the future, the trade-off between computationally efficient cut generation techniques and best possible approximations of the value functions needs to be further explored.

5. *Regularization.* As a descendant of Kelley's cutting-plane method, SDDP has a computational complexity which grows exponentially in the dimension of the state variables. Therefore, it can become computationally intractable for problems with high-dimensional state space. This is aggravated by common reformulations, *e.g.*, in case of stagewise dependent uncertainty, that artificially augment the state space. For Kelley's method, regularization methods have proven helpful in accelerating the solution process. Whereas some first attempts have been made to regularize SDDP, an efficient regularization remains an open challenge.

6. *Reinforcement learning techniques.* As the case of batch learning shows, SDDP can benefit from acceleration techniques that are well-known and established in reinforcement learning, but have not been translated to SDDP setting yet. By exploiting its affinity to Q-learning, there should be a lot of potential to improve the computational performance of SDDP in practice.

7. *Decision-dependent uncertainty.* The only standard assumption for SDDP that has not been relaxed in the literature yet, is to allow for stagewise-dependent stochastic processes modeling the uncertainty in (MSLP). This topic has still to be studied.

Acknowledgments

The authors thank the anonymous referees and the editor for their detailed and insightful feedback which helped to improve earlier versions of this manuscript substantially.

References

[1] S. Ahmed, F. G. Cabral, and B. Freitas Paulo da Costa. Stochastic Lipschitz dynamic programming. *Mathematical Programming*, 191:755–793, 2022.

[2] D. Arjoon, Y. Mohamed, Q. Goor, and A. Tilmant. Hydro-economic risk assessment in the eastern Nile River basin. *Water Resources and Economics*, 8:16–31, 2014.

[3] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.

[4] N. V. Arvanitidis and J. Rosing. Composite representation of a multireservoir hydroelectric power systems. *IEEE Transactions on Power Apparatus and Systems*, 89(2):319–326, 1970.

[5] T. Asamov and W. B. Powell. Regularized decomposition of high-dimensional multistage stochastic programs with markov uncertainty. *SIAM Journal on Optimization*, 28(1):575–595, 2018.

[6] D. Ávila, A. Papavasiliou, and N. Löhndorf. Batch learning SDDP for long-term hydrothermal planning. *IEEE Transactions on Power Systems*, 39(1):614–627, 2024.

[7] H. Bakker, F. Dunke, and S. Nickel. A structuring review on multi-stage optimization under uncertainty: Aligning concepts from theory and practice. *Omega*, 96, 2020.

[8] M. Bandarra and V. Guigues. Single cut and multicut stochastic dual dynamic programming with cut selection for multistage stochastic linear programs: convergence proof and numerical experiments. *Computational Management Science*, 18:125–141, 2021.

[9] R. Baucke. An algorithm for solving infinite horizon markov dynamic programmes. Preprint, available online at `http://www.optimization-online.org/DB_FILE/2018/04/6565.pdf`, 2018.

[10] R. Baucke, A. Downward, and G. Zakeri. A deterministic algorithm for solving multistage stochastic programming problems. Preprint, available online at `http://www.optimization-online.org/DB_FILE/2017/07/6138.pdf`, 2017.

[11] R. Baucke, A. Downward, and G. Zakeri. A deterministic algorithm for stochastic minimax dynamic programmes. Preprint, available online at `http://www.optimization-online.org/DB_FILE/2018/02/6449.pdf`, 2018.

[12] G. Bayraksan and D. K. Love. Data-driven stochastic programming using phi-divergences. In *Tutorials in Operations Research: The Operations Research Revolution*, pages 1–19. INFORMS, 2015.

[13] R. E. Bellman. *Dynamic programming.* Princeton University Press, Princeton, New Jersey, 1957.

[14] M.M. Belsnes, A. Haugstad, B. Mo, and P. Markussen. Quota modeling in hydrothermal systems. In *IEEE Bologna Power Tech Conference Proceedings*, 2003.

[15] F. Beltrán, E. C. Finardi, G. M. Fredo, and W. de Oliveira. Improving the performance of the stochastic dual dynamic programming algorithm using Chebyshev centers. *Optimization and Engineering*, 2020.

[16] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization.* Princeton University Press, 2009.

[17] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[18] J. M. Velásquez Bermúdez. GDDP: generalized dual dynamic programming theory. *Annals of Operations Research*, 117(1-4):21–31, 2002.

[19] D. P. Bertsekas. *Dynamic Programming and Optimal Control, Volume II.* Athena Scientific, 4th edition edition, 2017.

[20] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

[21] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: a fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[22] A. Bhattacharya, J. P. Kharoufeh, and B. Zeng. Managing energy storage in microgrids: a multistage stochastic programming approach. *IEEE Transactions on Smart Grid*, 9(1):483–496, 2018.

[23] M. Biel and M. Johansson. Dynamic cut aggregation in L-shaped algorithms. Preprint, available online at `https://arxiv.org/abs/1910.13752`, 2019.

[24] J. R. Birge. *Solution methods for stochastic dynamic linear programs.* Phd dissertation, Systems Optimization Laboratory, Stanford University, 1980.

[25] J. R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, 33(5):989–1007, 1985.

[26] J. R. Birge and F. Louveaux. *Introduction to stochastic programming.* Springer Series in Operations Research and Financial Engineering. Springer Science & Business Media, 2nd edition, 2011.

[27] J. R. Birge and F. V. Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34(3):384–392, 1988.

[28] X. Blanchot, F. Clautiaux, B. Detienne, A. Froger, and M. Ruiz. The Benders by batch algorithm: Design and stabilization of an enhanced algorithm to solve multicut Benders reformulation of two-stage stochastic programs. *European Journal of Operational Research*, 309(1):202–216, 2023.

[29] J. F. Bonnans, Z. Cen, and T. Christel. Energy contracts management by stochatic programming techniques. *Annals of Operations Research*, 200:199–222, 2012.

[30] R. B. S. Brandi, A. L. M. Marcato, B. H. Dias, T. P. Ramos, and I. C. da Silva Junior. A convergence criterion for stochastic dual dynamic programming: application to the long-term operation planning problem. *IEEE Transactions on Power Systems*, 33(4):3678–3690, 2018.

[31] R. B. S. Brandi, T. P. Ramos, B. H. Dias, A. L. M. Marcato, and I. Chaves da Silva Junior. Improving stochastic dynamic programming on hydrothermal systems through an iterative process. *Electric Power Systems Research*, 123:147–153, 2015.

[32] A. Brigatto, A. Street, and D. M. Valladão. Assessing the cost of time-inconsistent operation policies in hydrothermal power systems. *IEEE Transactions on Power Systems*, 32(6):4541–4550, 2017.

[33] S. Bruno, S. Ahmed, A. Shapiro, and A. Street. Risk neutral and risk averse approaches to multistage renewable investment planning under uncertainty. *European Journal on Operational Research*, 250(3):979–989, 2016.

[34] L. Cambier and D. Scieur. FAST (Finally An SDDP Toolbox). Accessed April 27, 2021, `https://stanford.edu/~lcambier/cgi-bin/fast/index.php`.

[35] P. Carpentier, J-.P. Chancelier, G. Cohen, M. De Lara, and P. Girardeau. Dynamic consistency for stochastic optimal control problems. *Annals of Operations Research*, 200:247–263, 2012.

[36] S. Cerisola, J. M. Latorre, and A. Ramos. Stochastic dual dynamic programming applied to nonconvex hydrothermal models. *European Journal of Operational Research*, 218(3):687–697, 2012.

[37] R. M. Chabar, M. V. F. Pereira, S. Granville, L. A. Barroso, and N. A. Iliadis. Optimization of fuel contracts management and maintenance scheduling for thermal plants under price uncertainty. In *IEEE Power Systems Conference and Exposition*, pages 923–930. IEEE, 2006.

[38] Z.-L. Chen and W. B. Powell. A convergent cutting-plane and partial-sampling algorithm for multistage stochastic linear programs with recourse. *Journal of Optimization Theory and Applications*, 102(3):497–524, 1999.

[39] A. Chiralaksanakul and D. P. Morton. Assessing policy quality in multi-stage stochastic programming. *Stochastic Programming E-Print Series*, 12, 2004.

[40] B. F. P. da Costa and V. Leclère. Dual SDDP for risk-averse multistage stochastic programs. *Operations Research Letters*, 51(3):332–337, 2023.

[41] L. C. da Costa Jr., F. S. Thomé, J. Dias Garcia, and M. V. F. Pereira. Reliability-constrained power system expansion planning: a stochastic risk-averse optimization approach. *IEEE Transactions on Power Systems*, 36(1):97–106, 2021.

[42] E. L. da Silva and E. C. Finardi. Parallel processing applied to the planning of hydrothermal systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(8):721–729, 2003.

[43] G. B. Dantzig and G. Infanger. Multi-stage stochastic linear programs for portfolio optimization. *Annals of Operations Research*, 45:59–76, 1993.

[44] G.B. Dantzig. Linear programming under uncertainty. *Management Science*, 1(3–4):197–206, 1955.

[45] C. M. B. de Castro, A. L. M. Marcato, R. Castro Souza, I. Chaves Silva Junior, F. L. Cyrino Oliveira, and T. Pulinho. The generation of synthetic inflows via bootstrap to increase the energy efficiency of long-term hydrothermal dispatches. *Electric Power Systems Research*, 124:33–46, 2015.

[46] M. de Lara and V. Leclère. Building up time-consistency for risk-measures and dynamic optimization. *European Journal of Operational Research*, 249(1):177–187, 2016.

[47] V. L. de Matos and E. C. Finardi. A computational study of a stochastic optimization model for long term hydrothermal scheduling. *International Journal of Electrical Power & Energy Systems*, 43(1):1443–1452, 2012.

[48] V. L. de Matos, D. P. Morton, and E. C. Finardi. Assessing policy quality in a multistage stochastic program for long-term hydrothermal scheduling. *Annals of Operations Research*, 253:713–731, 2017.

[49] V. L. de Matos, A. B. Philpott, and E. C. Finardi. Improving the performance of Stochastic Dual Dynamic Programming. *Journal of Computational and Applied Mathematics*, 290:196–208, 2015.

[50] L. Ding, S. Ahmed, and A. Shapiro. A Python package for multi-stage stochastic programming. Preprint, available online at `http://www.optimization-online.org/DB_FILE/2019/05/7199.pdf`, 2019.

[51] T. Ding, X. Zhang, R. Lu, M. Qu, M. Shahidehpour, Y. He, and T. Chen. Multi-stage distributionally robust stochastic dual dynamic programming to multi-period economic dispatch with virtual energy storage. *IEEE Transactions on Sustainable Energy*, 13(1):1–13, 2022.

[52] A. L. Diniz, M. E. P. Maceira, C. L. V. Vasconcellos, and D. D. J. Penna. A combined SDDP/Benders decomposition approach with a risk-averse surface concept for reservoir operation in long term power generation planning. *Annals of Operations Research*, 292:649–681, 2020.

[53] C. J. Donohue and J. R. Birge. The abridged nested decomposition method for multistage stochastic linear programs with relatively complete recourse. *Algorithmic Operations Research*, 1(1):20–30, 2006.

[54] T. N. dos Santos and A. L. Diniz. A new multiperiod stage definition for the multistage Benders decomposition approach applied to hydrothermal scheduling. *IEEE Transactions on Power Systems*, 24(3):1383–1392, 2009.

[55] A. Downward, O. Dowson, and R. Baucke. Stochastic dual dynamic programming with stagewise-dependent objective uncertainty. *Operations Research Letters*, 48(1):33–39, 2020.

[56] O. Dowson. The policy graph decomposition of multistage stochastic programming problems. *Networks*, 76(1):3–23, 2020.

[57] O. Dowson and L. Kapelevich. SDDP.jl: a Julia package for stochastic dual dynamic programming. *INFORMS Journal on Computing*, 33(1):27–33, 2020.

[58] O. Dowson, D. P. Morton, and A. Downward. Bi-objective multistage stochastic linear programming. *Mathematical Programming*, 196:907–933, 2022.

[59] O. Dowson, D. P. Morton, and B. K. Pagnoncelli. Partially observable multistage stochastic programming. *Operations Research Letters*, 48(4):505–512, 2020.

[60] O. Dowson, D. P. Morton, and B. K. Pagnoncelli. Incorporating convex risk measures into multistage stochastic programming algorithms. *Annals of Operations Research*, 2022.

[61] O. Dowson, A. Philpott, A. Mason, and A. Downward. A multi-stage stochastic optimization model of a pastoral dairy farm. *European Journal of Operations Research*, 274(3):1077–1089, 2019.

[62] I. Dunning, J. Huchette, and M. Lubin. JuMP: a modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[63] J. Dupačová and V. Kozmík. SDDP for multistage stochastic programs: preprocessing via scenario reduction. *Computational Management Science*, 14(1):67–80, 2017.

[64] J. Dupačová and V. Kozmík. Structure of risk-averse multistage stochastic programs. *OR Spectrum*, 37:559–582, 2015.

[65] D. Duque and D. P. Morton. Distributionally robust stochastic dual dynamic programming. *SIAM Journal on Optimization*, 30(4):2841–2865, 2020.

[66] J. L. Durante, J. Nascimento, and W. B. Powell. Risk directed importance sampling in stochastic dual dynamic programming with hidden Markov models for grid level energy storage. Preprint, available online at `https://arxiv.org/pdf/2001.06026.pdf`, 2020.

[67] A. Eichhorn and W. Römisch. Polyhedral risk measures in stochastic programming. *SIAM Journal on Optimization*, 16(1):69–95, 2005.

[68] M.J. Feizollahi, S. Ahmed, and X. A. Sun. Exact augmented Lagrangian duality for mixed integer linear programming. *Mathematical Programming*, 161(1-2):365–387, 2017.

[69] M. L. Fisher. An applications oriented guide to Lagrangian relaxation. *Interfaces*, 15(2):10–21, 1985.

[70] B. C. Flach, L. A. Barroso, and M. V. F. Pereira. Long-term optimal allocation of hydro generation for a price-maker company in a competitive market: latest developments and a stochastic dual dynamic programming approach. *IET Generation, Transmission & Distribution*, 4(2):299–314, 2010.

[71] M. Forcier and V. Leclère. Trajectory following dynamic programming algorithms without finite support assumptions. *Journal of Convex Analysis*, 30(3):951–999, 2023.

[72] S.M. Frank and S. Rebennack. An introduction to optimal power flow: theory, formulation, and examples. *IIE Transactions*, 48(12):1172–1197, 2016.

[73] C. Füllner and S. Rebennack. Non-convex nested Benders decomposition. *Mathematical Programming*, 196:987–1024, 2022.

[74] H. I. Gassmann. MSLiP: A computer code for the multistage stochastic linear programming problem. *Mathematical Programming*, 47:407–423, 1990.

[75] A. M. Geoffrion. Lagrangean relaxation for integer programming. In M. L. Balinski, editor, *Approaches to integer programming*, pages 82–114. Springer Berlin Heidelberg, Berlin, Heidelberg, 1974.

[76] A. Georghiou, A. Tsoukalas, and W. Wiesemann. Robust Dual Dynamic Programming. *Operations Research*, 67(3):813–830, 2019.

[77] H. Gevret, N. Langrené, J. Lelong, R. Lobato, T. Ouillon, X. Warin, and A. Maheshwari. STochastic OPTimization library in C++. Technical report, EDF Lab, 2018.

[78] P. Girardeau, V. Leclère, and A. B. Philpott. On the convergence of decomposition methods for multistage stochastic convex programs. *Mathematics of Operations Research*, 40(1):130–145, 2015.

[79] A. Gjelsvik, M. M. Belsnes, and A. Haugstad. An algorithm for stochastic medium-term hydrothermal scheduling under spot price uncertainty. In *Proceedings of 13th power systems computation conference*, 1999.

[80] A. Gjelsvik, M. M. Belsnes, and M. Håland. A case of hydro scheduling with a stochastic price model. In E. Broch, D.K. Lysne, N. Flatabø, and E. Helland-Hansen, editors, *Procedings of the 3rd international conference on hydropower*, pages 211–218. A.A. Balkema, 1997.

[81] A. Gjelsvik, B. Mo, and A. Haugstad. Long- and medium-term operations planning and stochastic modelling in hydro-dominated power systems based on stochastic dual dynamic programming. In S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors, *Handbook of power systems*, Energy Systems. Springer-Verlag Berlin Heidelberg, 2010.

[82] A. Gjelsvik and S. W. Wallace. Methods for stochastic medium-term scheduling in hydro-dominated power systems. Technical report, Norwegian Electric Power Research Institute, Trondheim, 1996.

[83] Z. Guan and A. B. Philpott. A multistage stochastic programming model for the New Zealand dairy industry. *International Journal of Production Economics*, 134(2):289–299, 2011.

[84] V. Guigues. SDDP for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57:167–203, 2014.

[85] V. Guigues. Convergence analysis of sampling-based decomposition methods for risk-averse multistage stochastic convex programs. *SIAM Journal on Optimization*, 26(4):2468–2494, 2016.

[86] V. Guigues. DASC: a Decomposition Algorithm for multistage stochastic programs with Strongly Convex cost functions. Preprint, available online at `https://arxiv.org/pdf/1711.04650.pdf`, 2017.

[87] V. Guigues. Dual dynamic programing with cut selection: convergence proof and numerical results. *European Journal of Operational Research*, 258(1):47–57, 2017.

[88] V. Guigues. Inexact cuts in stochastic dual dynamic programming. *SIAM Journal on Optimization*, 30(1):407–438, 2020.

[89] V. Guigues. Multistage stochastic programs with a random number of stages: dynamic programming equations, solution methods, and application to portfolio selection. *Optimization Methods and Software*, 36(1):211–236, 2021.

[90] V. Guigues, M. A. Lejeune, and W. Tekaya. Regularized stochastic dual dynamic programming for convex nonlinear optimization problems. *Optimization and Engineering*, 21:1133–1165, 2020.

[91] V. Guigues, R. Monteiro, and B. Svaiter. Inexact cuts in SDDP applied to multistage stochastic nondifferentiable problems. *SIAM Journal on Optimization*, 31(3):2084–2110, 2021.

[92] V. Guigues and R. D. C. Monteiro. Stochastic dynamic cutting plane for multistage stochastic convex programs. *Journal of Optimization Theory and Applications*, 189:513–559, 2021.

[93] V. Guigues and W. Römisch. Sampling-based decomposition methods for multistage stochastic programs based on extended polyhedral risk measures. *SIAM Journal on Optimization*, 22(2):286–312, 2012.

[94] V. Guigues and W. Römisch. SDDP for multistage stochastic linear programs based on spectral risk measures. *Operations Research Letters*, 40(12):313–318, 2012.

[95] V. Guigues and C. Sagastizábal. The value of rolling-horizon policies for risk-averse hydro-thermal planning. *European Journal of Operational Research*, 217(1):129–140, 2012.

[96] V. Guigues and C. Sagastizábal. Risk-averse feasible policies for large-scale multistage stochastic linear programs. *Mathematical Programming*, 138:167–198, 2013.

[97] V. Guigues, A. Shapiro, and Y. Cheng. Duality and sensitivity analysis of multistage linear stochastic programs. *European Journal of Operational Research*, 308(2):752–767, 2023.

[98] V. Guigues, A. Shapiro, and Y. Cheng. Risk-averse stochastic optimal control: an efficiently computable statistical upper bound. *Operations Research Letters*, 51:393–400, 2023.

[99] A. Helseth and H. Braaten. Efficient parallelization of the stochastic dual dynamic programming algorithm applied to hydropower scheduling. *Energies*, 8(12):14287–14297, 2015.

[100] A. Helseth, M. Fodstad, and B. Mo. Optimal medium-term hydropower scheduling considering energy and reserve capacity markets. *IEEE Transactions on Sustainable Energy*, 7(3):934–942, 2016.

[101] A. Helseth, B. Mo, M. Fodstad, and M. N. Hjelmeland. Co-optimizing sales of energy and capacity in a hydropower scheduling model. In *IEEE Power Tech*, pages 1–6, Eindhoven, The Netherlands, 2015.

[102] M. Hindsberger. ReSa: A method for solving multistage stochastic linear programs. *Journal of Applied Operational Research*, 6(1):2–15, 2014.

[103] M. N. Hjelmeland, J. Zou, A. Helseth, and S. Ahmed. Nonconvex medium-term hydropower scheduling by stochastic dual dynamic integer programming. *IEEE Transactions on Sustainable Energy*, 10(1), 2019.

[104] T. Homem-de-Mello and G. Bayraksan. Monte Carlo sampling-based methods for stochastic optimization. *Surveys in Operations Research and Management Science*, 19(1):56–85, 2014.

[105] T. Homem-de-Mello, V. L. De Matos, and E. C. Finardi. Sampling strategies and stopping criteria for stochastic dual dynamic programming: a case study in long-term hydrothermal scheduling. *Energy Systems*, 2(1):1–31, 2011.

[106] T. Homem-de-Mello and B. K. Pagnoncelli. Risk aversion in multistage stochastic programming: A modeling and algorithmic perspective. *European Journal of Operational Research*, 249(1):188–199, 2016.

[107] J. Huang, K. Zhou, and Y. Guan. A study of distributionally robust multistage stochastic optimization. Preprint, available online at `https://arxiv.org/abs/1708.07930`, 2017.

[108] N.A. Iliadis, V.F. Perira, S. Granville, M. Finger, P.A. Haldi, and L.A. Barroso. Benchmarking of hydroelectric stochastic risk management models using financial indicators. In *2006 IEEE power engineering society general meeting*, 2006.

[109] G. Infanger and D. P. Morton. Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75:241–256, 1996.

[110] J. Kallrath, P. M. Pardalos, S. Rebennack, and M. Scheidt, editors. *Optimization in the energy industry.* Springer, 2009.

[111] J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[112] A. Kiszka and D. Wozabal. Stochastic dual dynamic programming for optimal power flow problems under uncertainty. Preprint, available online at `https://optimization-online.org/wp-content/uploads/2021/11/8689.pdf`, 2021.

[113] V. Kozmík. On variance reduction of mean-CVaR Monte Carlo estimators. *Computational Management Science*, 12(2):221–242, 2015.

[114] V. Kozmík and D. P. Morton. Evaluating policies in risk-averse multi-stage stochastic programming. *Mathematical Programming*, 152:275–300, 2015.

[115] V. Krasko and S. Rebennack. Two-stage stochastic mixed-integer nonlinear programming model for post-wildfire debris flow hazard management: mitigation and emergency evacuation. *European Journal of Operational Research*, 263(1):265–282, 2017.

[116] T. Kristiansen. Hydropower scheduling and financial risk management. *Optimal Control Applications and Methods*, 27(1):1–18, 2006.

[117] G. Lan. Complexity of stochastic dual dynamic programming. *Mathematical Programming*, 191:717–754, 2022.

[118] Y. Lan, Q. Zhai, X. Liu, and X. Guan. Fast stochastic dual dynamic programming for economic dispatch in distribution systems. *IEEE Transactions on Power Systems*, 38(4):3828–3840, 2022.

[119] V. Leclère, P. Carpentier, J.-P. Chancelier, A. Lenoir, and F. Pacaud. Exact converging bounds for stochastic dual dynamic programming via Fenchel duality. *SIAM Journal on Optimization*, 30(2):1223–1250, 2020.

[120] V. Leclère, F. Pacaud, T. Rigaut, and H. Gerard. StochDynamicProgramming.jl. Code released on GitHub `https://github.com/JuliaStochOpt/StochDynamicProgramming.jl`.

[121] B. Legat. StructDualDynProg.jl. Code released on GitHub `https://github.com/JuliaStochOpt/StructDualDynProg.jl`.

[122] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical Programming*, 69(1-3):111–147, 1995.

[123] C. M. Leocadio, C. S. G. Richa, M. Z. Fortes, V. H. Ferreira, and B. H. Dias. Economic evaluation of a CHP biomass plant using stochastic dual dynamic programming. *Electrical Engineering*, 102:2605–2615, 2020.

[124] L. M. M. Lima, E. Popova, and P. Damien. Modeling and forecasting of Brazilian reservoir inflows via dynamic linear models. *International Journal of Forecasting*, 30(3):464–476, 2014.

[125] K. Linowsky and A. B. Philpott. On the convergence of sampling-based decomposition algorithms for multistage stochastic programs. *Journal of Optimization Theory and Applications*, 125(2):349–366, 2005.

[126] R. P. Liu and A. Shapiro. Risk neutral reformulation approach to risk averse stochastic programming. *European Journal of Operational Research*, 286(1):21–31, 2020.

[127] T. Lohmann, A. S. Hering, and S. Rebennack. Spatio-temporal hydro forecasting of multireservoir inflows for hydro-thermal scheduling. *European Journal of Operational Research*, 255(1):243–258, 2016.

[128] N. Löhndorf. An empirical analysis of scenario generation methods for stochastic optimization. *European Journal of Operational Research*, 255(1):121–132, 2016.

[129] N. Löhndorf and A. Shapiro. Modeling time-dependent randomness in stochastic dual dynamic programming. *European Journal of Operational Research*, 273(2):650–661, 2019.

[130] N. Löhndorf and D. Wozabal. Gas storage valuation in incomplete markets. *European Journal of Operational Research*, 288(1):318–330, 2021.

[131] N. Löhndorf, D. Wozabal, and S. Minner. Optimizing trading decisions for hydro storage systems using approximate dual dynamic programming. *Operations Research*, 61(4):810–823, 2013.

[132] R. Lu, T. Ding, B. Qin, J. Ma, X. Fang, and Z. Dong. Multi-stage stochastic programming to joint economic dispatch for energy and reserve with uncertain renewable energy. *IEEE Transactions on Sustainable Energy*, 11(3):1140–1151, 2020.

[133] M. E. P. Maceira and J. M. Damázio. Use of the PAR (p) model in the stochastic dual dynamic programming optimization scheme used in the operation planning of the Brazilian hydropower system. *Probability in the Engineering and Informational Sciences*, 20(1):143–156, 2006.

[134] M. E. P. Maceira, V. S. Duarte, D. D. J. Penna, L. A. M. Moraes, and A. C. G. Melo. Ten years of application of stochastic dual dynamic programming in official and agent studies in Brazil - description of the NEWAVE program. In *16th PSCC, Glasgow, Scotland*, 2008.

[135] M. E. P. Maceira, D. D. J. Penna, A. L. Diniz, R. J. Pinto, A. C. G. Melo, C. V. Vasconcellos, and C. B. Cruz. Twenty years of application of stochastic dual dynamic programming in official and agent studies in Brazil: main features and improvements on the NEWAVE model. In *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018.

[136] F. D. R. Machado, A. L. Diniz, C. L. T. Borges, and L. C. Brandão. Asynchronous parallel stochastic dual dynamic programming applied to hydrothermal generation planning. *Electric Power Systems Research*, 191:1–14, 2021.

[137] H. Macian-Sorribes, A. Tilmant, and M. Pulibo-Velazquez. Improving operating policies of large-scale surface-groundwater systems through stochastic programming. *Water Resources Research*, 53:1407–1423, 2017.

[138] G. F. Marques and A. Tilmant. The economic value of coordination in large-scale multireservoir systems: The Parana River case. *Water Resources Research*, 49:7546–7557, 2013.

[139] Y. Mbeutcha, M. Gendreau, and G. Emiel. Benefit of PARMA modeling for long-term hydroeletric scheduling using stochastic dual dynamic programming. *Journal of Water Resources Planning and Management*, 147(3):1–12, 2021.

[140] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: part I – convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[141] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

[142] B. Mo, A. Gjelsvik, and A. Grundt. Integrated risk management of hydro power scheduling and contract management. *IEEE Transactions on Power Systems*, 16(2):216–221, 2001.

[143] B. Mo, A. Gjelsvik, A. Grundt, and K. Karesen. Optimisation of hydropower operation in a liberalised market with focus on price modelling. In *PowerTech Proceedings*, 2001.

[144] D. P. Morton. An enhanced decomposition algorithm for multistage stochastic hydroelectric scheduling. *Annals of Operations Research*, 64(1):211–235, 1996.

[145] G. Nannicini, E. Traversi, and R. Wolfler Calvo. A benders squared (b^2) framework for infinite-horizon stochastic linear programs. *Mathematical Programming Computation*, 13:645–681, 2021.

[146] Y. Nesterov. *Introductory lectures on convex optimization: a basic course.* Springer, Boston, 2004.

[147] N. Newham. *Power system investment planning using stochastic dual dynamic programming.* PhD thesis, University of Canterbury, 2008.

[148] A. Papavasiliou, Y. Mou, L. Cambier, and D. Scieur. Application of stochastic dual dynamic programming to the real-time dispatch of storage under renewable supply uncertainty. *IEEE Transactions on Sustainable Energy*, 9(2):547–558, 2018.

[149] P. M. Pardalos, S. Rebennack, M. V. F. Pereira, N. A. Iliadis, and V. Pappu, editors. *Handbook of wind power systems.* Springer, 2013.

[150] P. Parpas, B. Ustun, M.Webster, and Q. K. Tran. Importance sampling in stochastic programming: a Markov Chain Monte Carlo approach. *INFORMS Journal on Computing*, 27(2):358–377, 2015.

[151] M. V. F. Pereira, N. M. Campodónico, and R. Kelman. Application of stochastic dual DP and extensions to hydrothermal scheduling. Technical report, PSRI, 1999.

[152] M. V. F. Pereira and L. M. V. G. Pinto. Stochastic optimization of a multireservoir hydroelectric system: a decomposition approach. *Water Resources Research*, 21(6):779–792, 1985.

[153] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375, 1991.

[154] M.V.F. Pereira. Optimal scheduling of hydrothermal systems - an overview. In *IFAC Symposium on Planning and Operation of Electric Energy Systems., Rio de Janeiro, Brazil, 22-25 July*, pages 1–6, 1985.

[155] S. J. Pereira-Cardenal, B. Mo, A. Gjelsvik, N. D. Riegels, K. Arnbjerg-Nielsen, and P. Bauer-Gottwein. Joint optimization of regional water-power systems. *Advances in Water Resources*, 92:200–207, 2016.

[156] L. Pfeiffer, R. Apparigliato, and S. Auchapt. Two methods of pruning Benders' cuts and their application to the management of a gas portfolio. Preprint, available online at `http://www.optimization-online.org/DB_FILE/2012/11/3683.pdf`, 2012.

[157] G. C. Pflug. *Probabilistic Constrained Optimization: Methodology and Applications*, chapter Some Remarks on the Value-at-Risk and the Conditional Value-at-Risk., pages 272–281. Kluwer Academic Publishers, 2000.

[158] G. C. Pflug and A. Pichler. Time-consistent decisions and temportal decomposition of coherent risk functionals. *Mathematics of Operations Research*, 41(2):682–699, 2016.

[159] G. C. Pflug and A. Ruszczyński. Measuring risk for income streams. *Computational Optimization and Applications*, 32(1):161–178, 2005.

[160] A. B. Philpott and V. L. de Matos. Dynamic sampling algorithms for multi-stage stochastic programs with risk aversion. *European Journal of Operational Research*, 218(2):470–483, 2012.

[161] A. B. Philpott, V. L. de Matos, and E. Finardi. On solving multistage stochastic programs with coherent risk measures. *Operations Research*, 61(4):957–970, 2013.

[162] A. B. Philpott, V. L. de Matos, and L. Kapelevich. Distributionally robust SDDP. *Computational Management Science*, 15:431–454, 2018.

[163] A. B. Philpott, F. Wahid, and F. Bonnans. MIDAS: A mixed integer dynamic approximation scheme. *Mathematical Programming*, 181:19–50, 2020.

[164] A.B. Philpott and Z. Guan. On the convergence of stochastic dual dynamic programming and related methods. *Operations Research Letters*, 36(4):450–455, 2008.

[165] J. Pina, A. Tilmant, and P. Côté. Optimizing multireservoir system operating policies using exogenous hydrologic variables. *Water Resources Research*, 53(11):9845–9859, 2017.

[166] R. J. Pinto, C. L. T. Borges, and M. E. P. Maceira. An efficient parallel algorithm for large scale hydrothermal system operation planning. *IEEE Transactions on Power Systems*, 28(4):4888–4896, 2013.

[167] B. T. Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9(3):14–29, 1969.

[168] H. Poorsepahy-Samian, V. Espanmanesh, and B. Zahraie. Improved inflow modeling in stochastic dual dynamic programming. *Journal of Water Resources Planning and Management*, 142(12):1–10, 2016.

[169] W. B. Powell. *Approximate dynamic programming: solving the curses of dimensionality.* Wiley-Interscience, 2007.

[170] W. B. Powell. *Reinforcement Learning and Stochastic Optimization – A Unified Framework for Sequential Decisions.* John Wiley & Sons, 2022.

[171] G. Pritchard. Stochastic inflow modeling for hydropower scheduling problems. *European Journal of Operational Research*, 246(2):496–504, 2015.

[172] PSR. SDDP - Stochastic hydrothermal dispatch with network restrictions. Accessed April 27, 2021, `https://www.psr-inc.com/softwares-en/`.

[173] Quantego. QUASAR. Accessed April 27, 2021, `http://quantego.com/`.

[174] A. R. Queiroz and D. P. Morton. Sharing cuts under aggregated forecast when decomposing multi-stage stochastic programs. *Operations Research Letters*, 41(3):311–316, 2013.

[175] F. Quezada, C. Gicquel, and S. Kedad-Sidhoum. Combining polyhedral approaches and stochastic dual dynamic integer programming for solving the uncapacitated lot-sizing problem under uncertainty. *INFORMS Journal on Computing*, 34(2):1024–1041, 2022.

[176] H. Rahimian and S. Mehrotra. Distributionally robust optimization: a review. Preprint, available online at `https://arxiv.org/pdf/1908.05659.pdf`, 2019.

[177] L. Raso, P.-O. Malaterre, and J.-C. Bader. Effective streamflow process modeling for optimal reservoir operation using stochastic dual dynamic programming. *Journal of Water Resources Planning and Management*, 143(4):1–11, 2017.

[178] S. Rebennack. Generation expansion planning under uncertainty with emissions quotas. *Electric Power Systems Research*, 114:78–85, 2014.

[179] S. Rebennack. Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming. *Mathematical Programming*, 156:343–389, 2016.

[180] S. Rebennack, B. Flach, M.V.F. Pereira, and P.M. Pardalos. Stochastic hydro-thermal scheduling under CO_2 emissions constraints. *IEEE Transactions on Power Systems*, 27(1):58–68, 2012.

[181] S. Rebennack, N.A. Iliadis, M.V.F. Pereira, and P.M. Pardalos. Electricity and co_2 emissions system prices modeling and optimization. In *PowerTech, 2009 IEEE Bucharest*, pages 1–6. IEEE, 2009.

[182] S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors. *Handbook of power systems I.* Springer, 2010.

[183] S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors. *Handbook of power systems II.* Springer, 2010.

[184] M. Resener, S. Rebennack, P. M. Pardalos, and S. Haffner, editors. *Handbook of optimization in electric power distribution systems.* Springer, 2020.

[185] R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of risk*, 2(3):21–42, 2000.

[186] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis.* Springer Berlin Heidelberg, 2009.

[187] T. A. Rotting and A. Gjelsvik. Stochastic dual dynamic programming for seasonal scheduling in the Norwegian power system. *IEEE Transactions on Power Systems*, 7(1):273–279, February 1992.

[188] C. Rougé and A. Tilmant. Using stochastic dual dynamic programming in problems with multiple near-optimal solutions. *Water Resources Research*, 52(5):4151–4163, 2016.

[189] C. Rougé, A. Tilmant, B. Zaitchik, A. Dezfuli, and M. Salman. Identifying key water resource vulnerabilities in data-scarce transboundary river basins. *Water Resources Research*, 54(8):5264–5281, 2018.

[190] B. Rudloff, A. Street, and D. M. Valladão. Time consistency and risk averse dynamic decision models: definition, interpretation and practical consequences. *European Journal of Operational Research*, 234(3):743–750, 2014.

[191] A. Ruszczyński. Decomposition methods in stochastic programming. *Mathematical Programming*, 79(1-3):333–353, 1997.

[192] A. Ruszczyński. Risk-averse dynamic programming for Markov decision processes. *Mathematical Programming*, 125:235–261, 2010.

[193] A. Ruszczyński and A. Shapiro. Conditional risk mappings. *Mathematics of Operations Research*, 31(3):544–561, 2006.

[194] A. Ruszczyński and A. Swietanowski. Accelerating the regularized decomposition method for two stage stochastic linear problems. *European Journal of Operational Research*, 101(2):328–342, 1997.

[195] A. Shapiro. Inference of statistical bounds for multistage stochastic programming problems. *Mathematical Methods of Operations Research*, 58(1):57–68, 2003.

[196] A. Shapiro. On complexity of multistage stochastic programs. *Operations Research Letters*, 34(1):1–8, 2005.

[197] A. Shapiro. Stochastic programming approach to optimization under uncertainty. *Mathematical Programming*, 112(1):183–220, 2008.

[198] A. Shapiro. Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, 209(1):63–72, 2011.

[199] A. Shapiro. Tutorial on risk neutral, distributionally robust and risk averse multistage stochastic programming. *European Journal of Operational Research*, 288(1):1–13, 2021.

[200] A. Shapiro and Y. Cheng. Dual bounds for periodical stochastic programs. *Operations Research*, 71(1):120–128, 2022.

[201] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory, Second Edition.* Society for Industrial and Applied Mathematics, 2014.

[202] A. Shapiro and L. Ding. Upper bound for optimal value of risk averse multistage problems. Technical report, Georgia Tech, 2016.

[203] A. Shapiro and L. Ding. Periodical multistage stochastic programs. *SIAM Journal on Optimization*, 30(3):2083–2102, 2020.

[204] A. Shapiro, W. Tekaya, J. P. da Costa, and M. P. Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391, 2013.

[205] A. Shapiro, W. Tekaya, M. P. Soares, and J. P. da Costa. Worst-case-expectation approach to optimization under uncertainty. *Operations Research*, 61(6):1435–1449, 2013.

[206] P. Shinde, I. Kouveliotis-Lysikatos, and M. Amelin. Multistage stochastic programming for VPP trading in continuous intraday electricity markets. *IEEE Transactions on Sustainable Energy*, 13(2):1–12, 2022.

[207] M. Siddig and Y. Song. Adaptive partition-based SDDP algorithms for multistage stochastic linear programming with fixed recourse. *Computational Optimization and Applications*, 81:201–250, 2022.

[208] M. P. Soares, A. Street, and D. M. Valladão. On the solution variability reduction of stochastic dual dynamic programming applied to energy planning. *European Journal of Operational Research*, 258(2):743–760, 2017.

[209] A. Sorokin, S. Rebennack, P.M. Pardalos, N.A. Iliadis, and M.V.F. Pereira, editors. *Handbook of networks in power systems I.* Springer Science & Business Media, 2012.

[210] A. Sorokin, S. Rebennack, P.M. Pardalos, N.A. Iliadis, and M.V.F. Pereira, editors. *Handbook of networks in power systems II.* Springer Science & Business Media, 2012.

[211] G. Steeger, T. Lohmann, and S. Rebennack. Strategic bidding for a price-maker hydroelectric producer: stochastic dual dynamic programming and Lagrangian relaxation. *IISE Transactions*, 50(11):929–942, 2018.

[212] G. Steeger and S. Rebennack. Strategic bidding for multiple price-maker hydroelectric producers. *IIE Transactions*, 47(9):1013–1031, 2015.

[213] G. Steeger and S. Rebennack. Dynamic convexification within nested Benders decomposition using Lagrangian relaxation: an application to the strategic bidding problem. *European Journal of Operational Research*, 257(2):669–686, 2017.

[214] G.M. Steeger. *Strategic bidding for price-maker hydroelectric producers.* PhD thesis, Colorado School of Mines, 2014.

[215] A. Street, L. A. Barroso, B. Flach, M. V. Pereira, and S. Granville. Risk constrained portfolio selection of renewable sources in hydrothermal electricity markets. *IEEE Transactions on Power Systems*, 24(3):1136–1144, 2009.

[216] A. Street, A. Brigatto, and D. M. Valladão. Co-optimization of energy and ancillary services for hydrothermal operation planning under a general security criterion. *IEEE Transactions on Power Systems*, 32(6):4914–4923, 2017.

[217] X. A. Sun and A. J. Conejo. *Robust optimization in electric energy systems.* International Series in Operations Research & Management Science. Springer, 2022.

[218] S. Thevenin, Y. Adulyasak, and J.-F. Cordeau. Stochastic dual dynamic programming for multiechelon lot sizing with component substitution. *INFORMS Journal on Computing*, 34(6):3151–3169, 2022.

[219] F.S. Thomé. *Representação de não-convexidade no planejamento da operação hidrotérmica utilizando PDDE.* PhD thesis, COPPE-UFRJ, 2013.

[220] A. Tilmant, L. Beevers, and B. Muyunda. Restoring a flow regime through the coordinated operation of a multireservoir system: The case of the Zambezi River basin. *Water Resources Research*, 46(7):1–11, 2010.

[221] A. Tilmant and R. Kelman. A stochastic approach to analyze trade-offs and risks associated with large-scale water resources systems. *Water Resources Research*, 43(6), 2007.

[222] A. Tilmant and W. Kinzelbach. The cost of noncooperation in international river basins. *Water Resources Research*, 48(1):1–12, 2012.

[223] A. Tilmant, J. Lettany, and R. Kelman. Hydrological risk assessment in the Euphrates-Tigris river basin: A stochastic dual dynamic programming approach. *Water International*, 32(2):294–309, 2009.

[224] A. Tilmant, D. Pinte, and Q. Goor. Assessing marginal water values in multipurpose multireservoir systems via stochastic programming. *Water Resources Research*, 44(12):1–17, 2008.

[225] D. M. Valladão, T. Silva, and M. Poggi. Time-consistent risk-constrained dynamic portfolio optimization with transactional costs and time-dependent returns. *Annals of Operations Research*, 282:379–405, 2019.

[226] W. van Ackooij, W. de Oliveira, and Y. Song. On level regularization with normal solutions in decomposition methods for multistage stochastic programming problems. *Computational Optimization and Applications*, 74:1–42, 2019.

[227] W. van Ackooij and X. Warin. On conditional cuts for stochastic dual dynamic programming. *EURO Journal on Computational Optimization*, 8(2):173–199, 2020.

[228] R. M. van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

[229] F. Wahid. *River optimisation: short-term hydro-bidding under uncertainty.* Optimization and control, Université Paris-Saclay; University of Auckland, New Zealand, 2017.

[230] W.W.-G. Yeh. Reservoir management and operations models: a state-of-the-art review. *Water Resources Research*, 21(12):1797–1818, 1985.

[231] S. Zhang and X. A. Sun. Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 196:935–985, 2022.

[232] Q.P. Zheng, S. Rebennack, P.M. Pardalos, M.V.F. Pereira, and N.A. Iliadis, editors. *Handbook of CO2 in power systems.* Springer Science & Business Media, 2012.

[233] X. Zhu, M.G. Genton, Y. Gu, and L. Xie. Space-time wind speed forecasting for improved power system dispatch. *TEST*, 23(1):1–25, 2014.

[234] J. Zou, S. Ahmed, and X. A. Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.

[235] L. Zéphyr and C. L. Anderson. Stochastic dynamic programming approach to managing power system uncertainty with distributed storage. *Computational Management Science*, 15:87–110, 2018.

Paper B

Non-convex nested Benders decomposition

Christian Füllner [a], Steffen Rebennack [a]

[a] *Karlsruhe Institute of Technology (KIT), Institute for Operations Research, Stochastic Optimization, Karlsruhe, Germany*

Published in *Mathematical Programming*, suggested citation:
Füllner C., Rebennack S. (2022), Non-convex nested Benders decomposition, Mathematical Programming, 196, 987-1024. `https://doi.org/10.1007/s10107-021-01740-0`.

Mathematical Programming (2022) 196:987–1024
https://doi.org/10.1007/s10107-021-01740-0

FULL LENGTH PAPER
Series B

Non-convex nested Benders decomposition

Christian Füllner[1] · Steffen Rebennack[1]

Received: 2 April 2020 / Accepted: 8 November 2021 / Published online: 7 January 2022

Abstract
We propose a new decomposition method to solve multistage non-convex mixed-integer (stochastic) nonlinear programming problems (MINLPs). We call this algorithm *non-convex nested Benders decomposition* (NC-NBD). NC-NBD is based on solving dynamically improved mixed-integer linear outer approximations of the MINLP, obtained by piecewise linear relaxations of nonlinear functions. Those MILPs are solved to global optimality using an enhancement of nested Benders decomposition, in which regularization, dynamically refined binary approximations of the state variables and Lagrangian cut techniques are combined to generate Lipschitz continuous non-convex approximations of the value functions. Those approximations are then used to decide whether the approximating MILP has to be dynamically refined and in order to compute feasible solutions for the original MINLP. We prove that NC-NBD converges to an ε-optimal solution in a finite number of steps. We provide promising computational results for some unit commitment problems of moderate size.

Keywords Nested Benders decomposition · Mixed-integer nonlinear programming (MINLP) · Global optimization · Non-convexities · Non-convex value functions

Mathematics Subject Classification 90C26 · 90C11 · 49M27

1 Introduction

We propose a new decomposition method to solve multistage non-convex mixed-integer (stochastic) nonlinear programming problems (MINLPs), *i.e.*, optimization problems modeling a sequential decision making process. Continuous and integer

Christian Füllner
christian.fuellner@kit.edu

Steffen Rebennack
steffen.rebennack@kit.edu

1 Institute for Operations Research (IOR), Stochastic Optimization (SOP), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

decision variables and possibly non-convex objective functions and constraints are allowed for any of the T stages.

If multistage (stochastic) problems are too large to be solved by off-the-shelf solvers, then tailored solution techniques are required. One example are decomposition algorithms making use of the specific sequential and block-diagonal structure of the constraints. The problems are decomposed into a large number of smaller but coupled subproblems which are solved iteratively. One of the most common decomposition methods is Benders decomposition, introduced by Benders [6] for linear programs. Since then, it has been enhanced to several more general cases, such as convex problems (*generalized Benders decomposition* (GBD) [19]), two-stage stochastic linear problems (*L-shaped method* [49]) and multistage (stochastic) linear problems (*nested Benders decomposition* (NBD) [8]). To mitigate the curse-of-dimensionality related to NBD in the stochastic case, Pereira and Pinto introduced its sampling-based variant *stochastic dual dynamic programming* (SDDP) [35], which was followed by various extensions [23,37].

The basic principle of NBD is to use the dynamic programming formulation of a given multistage problem. For each stage $t \in \{1, \dots, T\}$, a parametric subproblem is considered. This subproblem contains only those constraints, variables and parts of the objective function related to this specific stage, plus a *value function* determining the optimal value of all following stages for a given stage t solution. Since the value functions are not known in advance, they are iteratively approximated with linear cutting-planes. However, this approach requires the value functions to be convex. Therefore, most decomposition methods for multistage problems cover linear programs, as their value functions are guaranteed to be piecewise linear and convex.

However, in many applications, also integer variables or non-linearities occur naturally. In such case, the value functions are no longer convex and may also no longer be continuous. Therefore, the classical Benders approach fails, as it is impossible to construct a tight convex polyhedral approximation [47].

Thus, more sophisticated approaches have been developed to use Benders-type decomposition methods for non-convex MINLPs, mostly for the two-stage case. Li et al. propose an extension of GBD to the non-convex case for two-stage stochastic MINLPs with functions separable in integer and continuous variables [29,30]. In [28], a branch-and-cut framework is presented, where in each node Lagrangian and generalized Benders cuts are constructed. Related methods are proposed in [26,33]. All these methods have not been generalized to the multistage case yet.

To handle non-convexities in multistage problems, a common idea is to use convex relaxations of the value function, *e.g.*, by relaxing the integrality constraints for MILPs or by convexifying nonlinear terms in a static manner. Dynamically convexifying the non-convex value functions using Lagrangian relaxation techniques allows for a polyhedral approximation by Lagrangian cuts [10,45,46]. None of these discussed approaches can guarantee to compute an optimal solution for non-convex multistage problems, though.

Only recently, some substantial progress has been made in generalizing the Benders decomposition idea to multistage problems with non-convex value functions directly. In [36], step functions are used, instead of cutting-planes, to approximate the value functions, presuming their monotonicity.

For the mixed-integer linear case, the *stochastic dual dynamic integer programming* (SDDiP) approach is proposed [55]. SDDiP is an enhancement of NBD and SDDP which allows the solution of multistage (stochastic) MILPs in case of binary state variables. The method is based on generating special Lagrangian cuts, which reproduce the lower convex envelope of the value function. As the latter is piecewise linear and exact at binary state variables, strong duality is ensured and the problem is solved to global optimality in a finite number of iterations. SDDiP is applied to multistage unit commitment in [54]. It is also applied to a problem containing non-convex functions in context of hydro power scheduling by using a static binary expansion of the state variables and a Big-M reformulation [22].

As long as the value functions are assured to be Lipschitz continuous and some recourse property is satisfied, the requirement of binary state variables can be dropped, as is shown by the Stochastic Lipschitz Dynamic programming (SLDP) method in [1]. Here, two types of non-convex Lipschitz continuous cuts are introduced: reverse-norm cuts and augmented Lagrangian cuts.

In [52], Zhang and Sun present a new framework to solve multistage non-convex stochastic MINLPs, generalizing both SDDiP and SLDP. Similarly to [1], nonlinear *generalized conjugacy cuts* are constructed by solving augmented dual problems. Moreover, as Lipschitz continuity is not assured for the value functions, a Lipschitz continuous regularized value function is considered within the decomposition method.

In this article, we propose a new method to solve multistage non-convex MINLPs to proven global optimality, which we refer to as *non-convex nested Benders decomposition* (NC-NBD). The method combines piecewise linear relaxations, regularization, binary approximation and the SDDiP Lagrangian cuts in a unique and dynamic fashion. Its basic idea is to solve a MINLP by iteratively improved MILP outer approximations, which in turn are solved using a NBD-based decomposition scheme similar to that in [52]. The binary and piecewise linear approximations are dynamically refined.

In particular, the original MINLP is outer approximated by MILPs, which are iteratively improved in an outer loop. Those MILPs are obtained by piecewise linear approximations of all occuring nonlinear functions, which is an established method in global optimization [50]. In general, using MILP relaxations is a common approach to global optimization solvers [27,32,53].

In an inner loop, the multistage MILPs are solved to approximate optimality in finitely many steps. This is achieved using a NBD-based decomposition method. In a forward pass through the stages, trial solutions for the dynamic programming equations are determined. As Lipschitz continuity of the value functions is not guaranteed, this is done solving a regularized forward pass problem, as proposed in [52]. For a sufficiently large, but finite parameter, the regularization is exact [14,52], so that still the desired MILP is solved.

In a backward pass through the stages, nonlinear non-convex cuts are constructed to approximate the non-convex value functions of the MILP. To this end, we make use of a binary approximation of the state variables in the subproblems. As proven in [55], for MILPs with binary state variables we obtain (sufficiently) tight cuts by solving Lagrangian dual problems. The constructed linear cuts are then projected back to the original state space, yielding a nonlinear, non-convex, but Lipschitz continuous approximation of the value functions. The binary approximation is refined dynamically within the inner loop if required. By careful construction, all existing cuts remain valid even with such refinements.

Once the MILP approximation is solved to approximate optimality, the cut approximation of the value functions is used in the outer loop to determine bounds for the optimal value of the original MINLP. If the bounds are sufficiently close, the algorithm terminates with an ε-optimal solution. Otherwise, the piecewise linear approximations are refined, and thus the approximating MILP is tightened. Again, by careful construction it is ensured that all previously generated cuts remain valid.

To our best knowledge, the above concepts have not been combined in this dynamic way to solve multistage non-convex MINLPs yet. In that regard, our work also differs significantly from the aforementioned solution techniques.

Our proposed decomposition scheme uses the same regularization technique and similar convergence ideas as in [52]. However, a fundamental difference is that we only apply this technique to solve MILP outer approximations of the original MINLP. This has the advantage that in our framework MINLPs have to be solved only occasionally. In contrast, in [52], MINLPs are assumed to be solved by some oracle in each iteration and cuts are generated directly for the MINLP, which is computationally challenging. Moreover, contrary to our approach, the method in [52] does not require recourse assumptions, but in return it only allows for state variables in the objective function.

In contrast to SDDiP [55] and SLDP [1], we solve MINLPs, and thus consider a larger solution framework with an inner and an outer loop. However, even the inner loop, in which MILPs are solved, differs from both approaches.

To solve MILPs with non-binary state variables using SDDiP, it is proposed to apply a *static* binary approximation [22,55]. This way, the original MILP is replaced by an approximating problem with only binary state variables. It can be shown that for a sufficiently small approximation precision, *i.e.*, an sufficiently large number of binary variables, an ε-optimal solution of an MILP can be determined with this approach under some recourse assumption [55]. However, for a given problem at hand, it is not necessarily clear in advance how this precision has to be chosen, as knowledge on a problem-specific Lipschitz constant is required. This becomes even more challenging in our framework, where an MINLP is iteratively approximated by MILPs, for which the required precision may change. On the contrary, within NC-NBD the binary approximation is refined dynamically if required.

More crucially, in NC-NBD the binary approximation is applied temporarily only to derive cuts in the backward pass. These cuts are then projected back to the original state space. This construction has a few key advantages: Firstly, it is ensured that cuts remain valid even if the binary precision is refined later on. Secondly, the original state variables remain continuous and are not limited to values which can be exactly represented by the binary approximation. This, in turn, ensures that the true MILPs are solved in the inner loop. Consequently, the generated cuts are valid for the value functions of these MILPs and, due to their relaxation property, also the original MINLP. Analogously, the obtained lower bounds are valid for the corresponding optimal values. Importantly, this is not true for SDDiP with static binary approximation, where the state space is permanently modified and only approximations of the true MILPs are solved in the inner loop. In our approach to solve MINLPs, it is crucial to determine guaranteed valid cuts for the value functions in both loops. Therefore, SDDiP cannot be used effectively in this setting.

Our cut generation approach also differs from that in SLDP [1] (and also [52]), where augmented Lagrangian problems are solved to determine nonlinear cuts. While our method comes at the cost of introducing additional (binary) variables and constraints compared to those approaches, *e.g.*, for the cut projection, we avoid solving dual problems containing nonlinear penalization in the objective. Such penalization may be disadvantageous as it prevents decomposition of the primal problems which are solved in the solution process of the dual problem. Additionally, in contrast to SLDP [1], we do not assume continuously complete recourse, but only the weaker complete recourse, as we circumvent the requirement of Lipschitz continuity of the true value functions by regularization.

The main contributions of this paper are as follows:

(1) We present the *non-convex nested Benders decomposition* (NC-NBD) method to globally solve general multistage non-convex MINLPs. The method combines piecewise linear relaxations, regularization, binary approximation and cutting-planes techniques in a unique way. In contrast to existing approaches, all approximations are improved dynamically where and when it is reasonable. To our knowledge, this is the first decomposition method for general multistage non-convex MINLPs.

(2) A crucial requirement using dynamic refinements is to ensure that all previously determined cuts remain valid within the refinement process and have not to be generated from scratch. We ensure this by a special cut projection and careful choice of the MILP relaxations.

(3) We prove that the proposed NC-NBD method converges to an ε-optimal solution of P in a finite number of steps under some mild assumptions.

(4) We provide first computational results of applying NC-NBD to moderate-sized instances of a unit commitment problem to illustrate its efficacy.

To enhance readability, we focus our discussions solely on deterministic MINLPs. However, the presented NC-NBD idea can also be applied to stochastic programs with stagewise independent and finite random variables.

The remainder of the paper is organized as follows. We present the considered problem formulation and assumptions in Sect. 2. Then, we introduce the NC-NBD with its different steps in Sect. 3, before presenting convergence results in Sect. 4. Afterwards, we provide computational results for instances of a simple unit commitment problem in Sect. 5. We conclude with Sect. 6.

2 Problem formulation

We consider the following multistage non-convex MINLP problems

$$(\boldsymbol{P}) \quad v := \min_{x_1,\dots,x_T,y_1,\dots,y_T} \sum_{t=1}^{T} f_t(x_t, y_t)$$
$$\text{s.t.} \quad (x_t, y_t) \in M_t(x_{t-1}) \quad \forall t = 1, \dots, T.$$

Here $t = 1, \ldots, T$ denotes the different stages with the final stage $T \in \mathbb{N}$. For each stage t, the decision variables can be separated into mixed-integer state variables $x_t \in \mathbb{R}_+^{n_t^1} \times \mathbb{Z}_+^{n_t^2}$ and local variables $y_t \in \mathbb{R}^{n_t^3} \times \mathbb{Z}^{n_t^4}$, with $x_0 = 0$. We define $n_t := n_t^1 + n_t^2$ as the number of state variables. The sets $M_t(x_{t-1})$ appearing in the constraints for each stage t are defined by

$$M_t(x_{t-1}) := \{(x_t, y_t) \in X_t \times Y_t \ : \ g_t(x_{t-1}, x_t, y_t) \leq 0, \ h_t(x_{t-1}, x_t, y_t) = 0\}.$$

X_t and Y_t denote box constraints; $X_0 := \{0\}$. As such, X_t and Y_t are compact sets for all stage-t variables. All functions $f_t : X_t \times Y_t \to \mathbb{R}$, $g_t : X_{t-1} \times X_t \times Y_t \to \mathbb{R}^{m_t^1}$ and $h_t : X_{t-1} \times X_t \times Y_t \to \mathbb{R}^{m_t^2}$ are well-defined on their domains.

To exploit its multistage structure, we solve $(\boldsymbol{P})$ by some extension of NBD. NBD makes use of the dynamic programming formulation of $(\boldsymbol{P})$, where each stage-t subproblem, $t = 1, \ldots, T$, can be denoted by

$$(\boldsymbol{P_t(x_{t-1})}) \quad Q_t(x_{t-1}) := \min_{x_t, y_t, z_t} \ f_t(x_t, y_t) + Q_{t+1}(x_t)$$
$$\text{s.t.} \quad (z_t, x_t, y_t) \in \mathcal{M}_t$$
$$z_t = x_{t-1},$$

with the value function $Q_t(\cdot)$ of stage t and $Q_{T+1}(\cdot) \equiv 0$. Note that x_t links different stages, *i.e.*, x_t is a decision variable for $(\boldsymbol{P_t(x_{t-1})})$ and a parameter for $(\boldsymbol{P_{t+1}(x_t)})$. For the first stage, we obtain that $Q_1(x_0) = v$ with $x_0 \equiv 0$. Importantly, subproblem $(\boldsymbol{P_t(x_{t-1})})$ is enhanced by introducing local copies z_t of the state variables x_{t-1} and the *copy constraints* $z_t = x_{t-1}$. Those copy constraints will prove crucial for the cut generation later on. Taking into account the local copies, we define

$$\mathcal{M}_t := \{(z_t, x_t, y_t) : z_t \in X_{t-1}, (x_t, y_t) \in M_t(z_t)\}.$$

As the subproblems $(\boldsymbol{P_t(x_{t-1})})$ are non-convex MINLPs, the value functions $Q_t(\cdot)$ may be non-continuous and non-convex, two detrimental properties for Benders decomposition approaches. To ensure that the value functions $Q_t(\cdot)$ are at least lower semicontinuous (l.sc.), we make the following technical assumptions:

(A1). For all $t = 1, \ldots, T$,

(a) the functions f_t are Lipschitz continuous on $X_t \times Y_t$,
(b) the functions g_t and h_t are continuous on $X_{t-1} \times X_t \times Y_t$.

(A2) (Complete recourse). For any stage t and any $\bar{x}_{t-1} \in X_{t-1}$, there exists some $(z_t, x_t, y_t) \in X_{t-1} \times X_t \times Y_t$ which is feasible for $(\boldsymbol{P_t(\bar{x}_{t-1})})$.

As all variables are box-constrained, the feasible set $M_t(x_{t-1})$ of $(\boldsymbol{P_t(x_{t-1})})$ is bounded. With assumption **(A1)** and the recourse assumption **(A2)**, all subproblems $(\boldsymbol{P_t(x_{t-1})})$ are feasible and bounded. Analogously, $(\boldsymbol{P})$ is feasible with finite optimal value v. Note that under assumption **(A2)** we can restrict to generating *optimality cuts* in NC-NBD without the need to introduce *Benders feasibility cuts*.

We obtain our required l.sc. property of the value functions $Q_t(\cdot)$.

Lemma 2.1 *Under assumptions* **(A1)** *and* **(A2)** *the value functions $Q_t(\cdot)$ are l.sc. for all $t = 1, \dots, T$.*

Proof Fixing all integer variables, the l.sc. follows from Exercise 1.19 in [41]. As X_t and Y_t are bounded, only finitely many different values can be attained by the integer variables. The minimum of finitely many l.sc. functions is l.sc. □

In the next section, we introduce the NC-NBD method, which combines regularization, piecewise linear approximations, binary expansion and special cutting-plane techniques in a unique way to solve $(\boldsymbol{P})$.

3 Non-convex nested Benders decomposition

3.1 The NC-NBD principle

The basic idea of the NC-NBD algorithm is to employ that MILP problems can be solved exactly by enhancements of NBD under certain assumptions and that MINLPs can be outer approximated by MILPs iteratively. Thus, the method consists of two main components. The first component is an inner loop which is used to determine an approximately optimal solution of some MILP outer approximation $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ of problem $(\boldsymbol{P})$. This approximation is determined by piecewise linear relaxations of nonlinear functions in $(\boldsymbol{P})$. The second component is an outer loop which refines this outer approximation iteratively (indexed by ℓ) to improve the approximation of the optimal value v of $(\boldsymbol{P})$. The NC-NBD is summarized in Algorithm 1 and illustrated in Fig. 1.

The inner loop follows the general principle of NBD to solve $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$. It consists of a *forward* and a *backward pass* through the stages $t = 1, \dots, T$ in each iteration i. In the forward pass, the stage-t subproblem $(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}^{\boldsymbol{\ell}}(\boldsymbol{x_{t-1}}))$ is approximated in two different ways: The value function $\widehat{Q}_{t+1}(\cdot)$ of the following stage is replaced by some outer approximation $\mathfrak{Q}_{t+1}^{\ell i}(\cdot)$. Moreover, a regularization is added to ensure Lipschitz continuity of the corresponding value functions. Thus, regularized subproblems $(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}^{\boldsymbol{R},\boldsymbol{\ell i}}(\boldsymbol{x_{t-1}}))$ are solved, as proposed in [52], yielding trial solutions $\widehat{x}_{t-1}^{\ell i}$ and an upper bound $\overline{\widehat{v}}^{\ell i}$ for $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$.

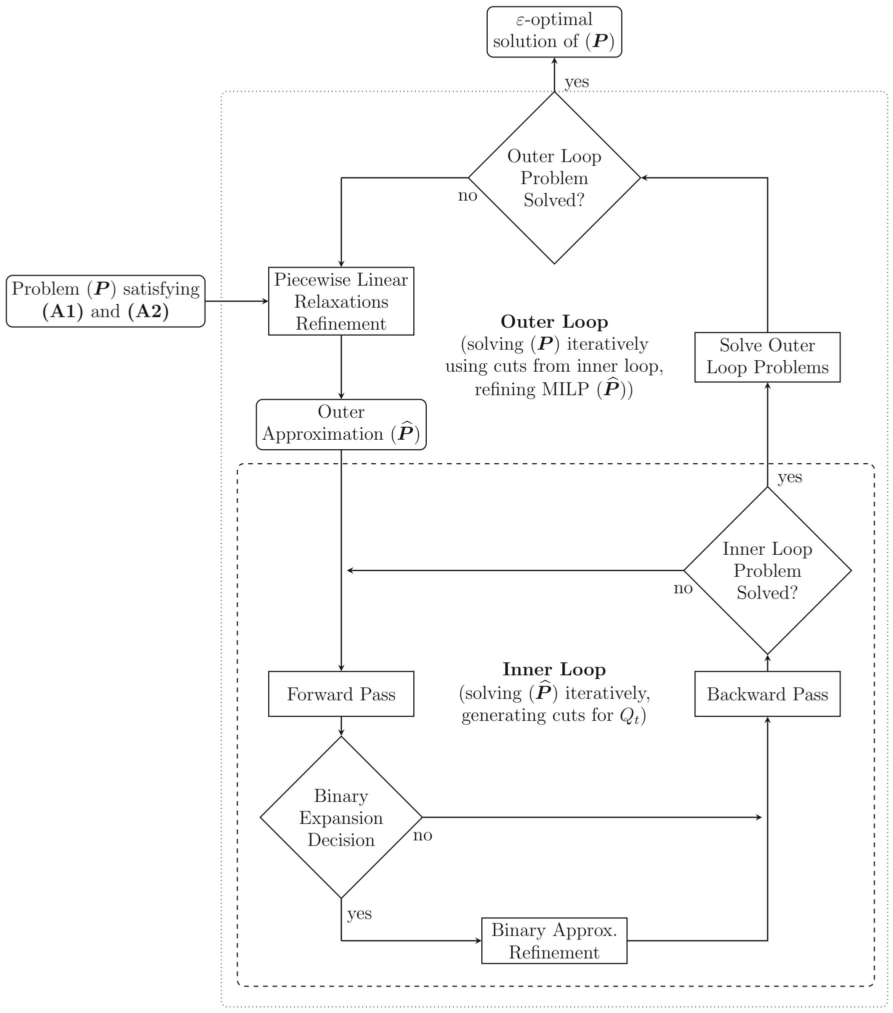
In the backward pass, the approximations $\mathfrak{Q}_{t+1}^{\ell i}(\cdot)$ of $\widehat{Q}_{t+1}(\cdot)$ are improved iteratively by constructing additional cuts. As the value functions are possibly non-convex, those cuts are nonlinear. Importantly, cuts for $\widehat{Q}_{t+1}(\cdot)$ are also valid for $Q_{t+1}(\cdot)$, as the first is an outer approximation of the latter.

In the literature, different ways are proposed to obtain nonlinear optimality cuts and to ensure that the inner loop converges to the optimal value $\widehat{v}^{\ell}$ of $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$. One method is to generate reverse-norm cuts [1]. However, this only works if the value functions themselves are Lipschitz continuous which is not guaranteed in our setting. Another, more general method is to solve some augmented Lagrangian dual problem, as proposed in [1,52].

We propose a third and new method, based on the SDDiP technique [55]. We utilize that we can generate sufficiently tight cuts by solving a Lagrangian dual in a lifted space, where all state variables are binary. Thus, we (temporarily) approximate the

Algorithm 1 NC-NBD

```
Input: Problem $(\boldsymbol{P})$ satisfying (A1), (A2), tolerances $\varepsilon > \widehat{\varepsilon} > 0$, scalar $K_t$ for all $t$, initial bin. approx.
    precision $\beta_t \in (0,1)^{K_{t-1}}$, upper bounds $\overline{v}^0 = +\infty$, lower bound $\underline{\widehat{v}}^0$, initial $\mathfrak{Q}_t^0(\cdot)$ for all $t$ and triangulations
    $\mathcal{T}_\gamma^0$ for all $\gamma \in \Gamma$, $\ell \leftarrow 0$.
 1: while $\overline{v}^\ell - \underline{\widehat{v}}^\ell > \varepsilon$ do
 2:     Set $\ell \leftarrow \ell + 1$, $i \leftarrow 1$. Set $\mathfrak{Q}_t^{\ell,1}(\cdot) \leftarrow \mathfrak{Q}_t^{\ell-1}(\cdot)$, $\underline{\widehat{v}}^{\ell,0} \leftarrow \underline{\widehat{v}}^{\ell-1}$, $\overline{\widehat{v}}^{\ell,0} \leftarrow \overline{\widehat{v}}^{\ell-1}$.
        ▷ PIECEWISE LINEAR RELAXATION REFINEMENT
 3:     Refine the piecewise linear approx. of all $\gamma \in \Gamma$ to obtain $\mathcal{T}_\gamma^\ell$ by longest-edge
        bisection of the simplex corresponding to $\big((\widehat{z}_t^\ell, \widehat{x}_t^\ell, \widehat{y}_t^\ell)\big)_{t=1,\dots,T}$.
 4:     Determine an outer approximation $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ of $(\boldsymbol{P})$ by using a MILP model and
        appropriate shifts for the piecewise linear approximations.
        ▷ INNER LOOP
 5:     Solve subproblem $(\widehat{\boldsymbol{P}}_{\boldsymbol{1}}^{\boldsymbol{R,\ell i}}(\boldsymbol{x}_{\boldsymbol{0}}^{\boldsymbol{\ell i}}, \boldsymbol{\mathfrak{Q}}_{\boldsymbol{2}}^{\boldsymbol{\ell i}}))$. Store the optimal point $(\widehat{z}_1^{\ell i}, \widehat{x}_1^{\ell i}, \widehat{y}_1^{\ell i})$.
 6:     while $\overline{\widehat{v}}^{\ell i} - \underline{\widehat{v}}^{\ell i} > \widehat{\varepsilon}$ do
            ▷ FORWARD PASS
 7:         for stages $t = 2, ..., T$ do
 8:             Solve subproblem $(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}^{\boldsymbol{R,\ell i}}(\widehat{\boldsymbol{x}}_{\boldsymbol{t-1}}^{\boldsymbol{\ell i}}, \boldsymbol{\mathfrak{Q}}_{\boldsymbol{t+1}}^{\boldsymbol{\ell i}}))$ satisfying (A3). Store the optimal
                point $(\widehat{z}_t^{\ell i}, \widehat{x}_t^{\ell i}, \widehat{y}_t^{\ell i})$.
 9:             $\overline{\widehat{v}}^{\ell,i} = \min\Big\{\overline{\widehat{v}}^{\ell,i-1}, \sum_{t=1}^T \Big(\widehat{f}_t(\widehat{x}_t^{\ell i}, \widehat{y}_t^{\ell i}) + \sigma_t \|\widehat{x}_{t-1}^{\ell i} - \widehat{z}_t^{\ell i}\|\Big)\Big\}$.
10:         end for
            ▷ BINARY APPROXIMATION REFINEMENT
11:         if Forward Pass solution in $i$ equals that in $i-1$ then
12:             Set $K_{tj} \leftarrow K_{tj} + 1$ for all $t$ and $j$. Set $\beta_{tj} = U_j \Big(\sum_{k=1}^{K_{t_j}} 2^{k-1}\Big)^{-1}$.
13:         end if
            ▷ BACKWARD PASS
14:         for stages $t = T, ..., 2$ do
15:             Determine the best binary approx. $\widehat{x}_{\mathbb{B},t-1}^i = B_{t-1}\lambda_{t-1}^i$ of the state $\widehat{x}_{t-1}^{\ell i}$.
16:             Solve subproblem $(\boldsymbol{D}_{\mathbb{B}\boldsymbol{t}}^{\boldsymbol{\ell i}}(\boldsymbol{\lambda}_{\boldsymbol{t-1}}^{\boldsymbol{i}}, \boldsymbol{\mathfrak{Q}}_{\boldsymbol{t+1}}^{\boldsymbol{\ell,i+1}}))$. Store the optimal multiplier $\pi_t^{\ell i}$ and
                the corresponding optimal value $c_t^{\ell i}$ of the Lagrangian dual function.
17:             Construct the cut $\phi_{\mathbb{B}t}^{\ell i}(\lambda_{t-1}) = c_t^{\ell i} + (\pi_t^{\ell i})^\top \lambda_{t-1}$ in the binary state space.
18:             Model the optimal value function $\phi_t^{\ell i}$ of projecting $\phi_{\mathbb{B}t}^{\ell i}$ to the original space
                by MILP constraints using the KKT conditions.
19:             Set $\mathfrak{Q}_t^{\ell,i+1}(x_t) = \max\{\mathfrak{Q}_t^{\ell i}(x_t), \phi_t^{\ell i}(x_t)\}$.
20:         end for
            ▷ FIRST STAGE UPDATE
21:         Solve subproblem $(\widehat{\boldsymbol{P}}_{\boldsymbol{1}}^{\boldsymbol{R,\ell i}}(\boldsymbol{x}_{\boldsymbol{0}}^{\boldsymbol{\ell i}}, \boldsymbol{\mathfrak{Q}}_{\boldsymbol{2}}^{\boldsymbol{\ell,i+1}}))$. Store the optimal point $(\widehat{z}_1^{\ell i}, \widehat{x}_1^{\ell i}, \widehat{y}_1^{\ell i})$.
22:         Update $\underline{\widehat{v}}^{\ell,i}$ to the optimal value $\underline{\widehat{Q}}_1^{\ell i}(0, \mathfrak{Q}_2^{\ell,i+1})$.
23:         $i \leftarrow i + 1$.
24:     end while
25:     Set $\underline{\widehat{v}}^\ell \leftarrow \underline{\widehat{v}}^{\ell i}$, $\overline{\widehat{v}}^\ell \leftarrow \overline{\widehat{v}}^{\ell i}$, $\mathfrak{Q}_t^\ell(\cdot) \leftarrow \mathfrak{Q}_t^{\ell i}(\cdot)$ for all $t = 2, ..., T$.
        ▷ OUTER LOOP PROBLEMS
26:     for stages $t = 1, \dots, T$ do
27:         Solve subproblem $(\boldsymbol{P}_{\boldsymbol{t}}^{\boldsymbol{\ell}}(\boldsymbol{x}_{\boldsymbol{t-1}}^{\boldsymbol{\ell}}, \boldsymbol{\mathfrak{Q}}_{\boldsymbol{t+1}}^{\boldsymbol{\ell}}))$. Store the optimal solution $(z_t^\ell, x_t^\ell, y_t^\ell)$.
28:     end for
29:     $\overline{v}^\ell = \min\Big\{\overline{v}^{\ell-1}, \sum_{t=1}^T f_t(x_t^\ell, y_t^\ell)\Big\}$.
30: end while
Output: $\varepsilon$-optimal solution $\big((z_t^\ell, x_t^\ell, y_t^\ell)\big)_{t=1,\dots,T}$ of $(\boldsymbol{P})$.
```



ε-optimal solution of ($\boldsymbol{P}$)
yes
Outer Loop Problem Solved?
no
Problem ($\boldsymbol{P}$) satisfying (A1) and (A2)
Piecewise Linear Relaxations Refinement
Outer Loop (solving ($\boldsymbol{P}$) iteratively using cuts from inner loop, refining MILP ($\widehat{\boldsymbol{P}}$))
Solve Outer Loop Problems
Outer Approximation ($\widehat{\boldsymbol{P}}$)
yes
Inner Loop Problem Solved?
no
Inner Loop (solving ($\widehat{\boldsymbol{P}}$) iteratively, generating cuts for Q_t)
Forward Pass
Backward Pass
Binary Expansion Decision
no
yes
Binary Approx. Refinement

Fig. 1 Conceptual overview of NC-NBD

state variables with binary ones, construct cuts in the binary space and then project those cuts back to the original space. As we show, these projections can be modeled by mixed-integer linear constraints in the original space. By careful construction, these cuts remain valid even if the binary approximation is refined in later iterations.

In this way, we circumvent solving an augmented Lagrangian dual, which may be even more expensive than solving the classical Lagrangian dual, as with the additional nonlinear term in the objective, the primal problems lose their decomposability. In return, we require more (binary) variables and constraints in the Lagrangian duals and for an MILP representation of our cuts than the approach in [1].

In principle, the MILPs as they occur in the inner loop could also be solved by using SDDiP with a static binary approximation of the state variables [55]. As discussed in

Sect. 1, this approach has some properties which prevent an efficient integration into our algorithmic framework, though.

As we show in the next section, for a sufficiently fine binary approximation, the obtained cuts in the NC-NBD provide a sufficiently good approximation at the trial solutions $\widehat{x}_{t-1}^{\ell i}$. Additionally, the cut approximations $\mathfrak{Q}_t^{\ell}(\cdot)$ are generated in such a way that they are Lipschitz continuous. This is sufficient to ensure convergence to a globally optimal solution of $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$.

At the end of the backward pass, a lower bound $\underline{\widehat{v}}^{\ell i}$ is determined. If $\overline{\widehat{v}}^{\ell i}$ and $\underline{\widehat{v}}^{\ell i}$ are sufficiently close to each other, an approximate globally minimal point $\big((\widehat{z}_t^{\ell}, \widehat{x}_t^{\ell}, \widehat{y}_t^{\ell})\big)_{t=1,\dots,T}$ of $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ has been identified and the inner loop is left. Otherwise, further cuts have to be constructed or the binary approximation has to be refined. We discuss this decision in more detail in Sect. 3.3.6.

Once the inner loop is left, subproblems $(\boldsymbol{P_t(x_{t-1}, \mathfrak{Q}_{t+1}^{\ell})})$ are solved to determine trial points x_{t-1}^{ℓ} and an upper bound $\overline{v}^{\ell}$ to v for the original problem $(\boldsymbol{P})$. If this upper bound is sufficiently close to $\underline{\widehat{v}}^{\ell}$, the solution $\big((z_t^{\ell}, x_t^{\ell}, y_t^{\ell})\big)_{t=1,\dots,T}$ is approximately optimal for problem $(\boldsymbol{P})$. If not, the MILP relaxation $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell+1}})$ is created by refining $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ in the neighborhood of $\big((\widehat{z}_t^{\ell}, \widehat{x}_t^{\ell}, \widehat{y}_t^{\ell})\big)_{t=1,\dots,T}$ and a new inner loop is started.

As for the inner loop, it is crucial that with these refinements in the outer loop all previously generated cuts remain valid. Otherwise, the cut approximation $\mathfrak{Q}_{t+1}^{\ell}(\cdot)$ would have to be built from scratch, counteracting the idea of a dynamic solution framework. In the following subsections, we show how such persistent validity can be achieved by careful design. Note that, even though we make use of the same regularization idea, our framework with nested loops and dynamic refinements also differs from the method presented in [52].

We explain the different steps of NC-NBD in more detail in the following subsections, before we discuss convergence results in Sect. 4. As long as the index ℓ is not needed for the discussions of the inner loop, we omit it for notational convenience. Moreover, we note that several of the considered subproblems require the introduction of additional decision variables, *e.g.*, for piecewise linear approximation or cut projection. For reasons of clarity and comprehensibility, by the terms *optimal point* or *optimal solution* we refer to the projection of their actual optimal points to the space $X_{t-1} \times X_t \times Y_t$, which we are interested in.

3.2 Piecewise linear relaxations

In the outer loop of NC-NBD, all nonlinear functions $\gamma \in \Gamma$ in problem $(\boldsymbol{P})$ are approximated by some piecewise linear functions. This is achieved by determining a triangulation of their domain, which in our box-constrained setting is always possible. Then, the piecewise linear functions can be defined on the simplices of this triangulation using the function values of γ at their vertices. For a thorough discussion and state-of-the-art approaches to construct piecewise linear approximations and triangulations, see [18,39,40].

The piecewise linear approximations can then be reformulated as mixed-integer linear constraints using auxiliary continuous and binary variables. In the literature, several modeling techniques have been proposed, such as the convex combination model, the incremental model and some logarithmic variants [4,18,38,51]. Later on, we draw on refinement and convergence ideas from [9], which work for several of these models, such as the generalized incremental model [9] or the disaggregated logarithmic convex combination model [51].

By shifting the approximations appropriately, it can be ensured that the obtained MILP $(\widehat{\boldsymbol{P}}^j)$ is indeed a relaxation of the original problem $(\boldsymbol{P})$ [18]. Alternatively, one can construct piecewise linear underestimators and overestimators, yielding tubes for nonlinear equations [25].

Applying the piecewise linear approximations to problem $(\boldsymbol{P})$, we obtain the MILP outer approximation with copy constraints

$$
\begin{aligned}
(\widehat{\boldsymbol{P}}) \quad \widehat{v} := \min_{\substack{x_1,\dots,x_T,y_1,\dots,y_T \\ z_1,\dots,z_T}} \quad & \sum_{t=1}^{T} \widehat{f}_t(x_t, y_t) \\
\text{s.t.} \quad & (z_t, x_t, y_t) \in \widehat{M}_t \qquad \forall t = 1, \dots, T \\
& z_t = x_{t-1} \qquad\qquad \forall t = 1, \dots, T.
\end{aligned}
$$

For reasons of clarity, we denote the piecewise linear relaxations of $f_t(\cdot)$, $g_t(\cdot)$ and $h_t(\cdot)$ by $\widehat{f}_t(\cdot)$, $\widehat{g}_t(\cdot)$ and $\widehat{h}_t(\cdot)$, although they are modeled using auxiliary constraints and variables. The set $\widehat{M}_t$ is defined by replacing the functions $g_t(\cdot)$ and $h_t(\cdot)$ in M_t or $M_t(x_{t-1})$, respectively, with $\widehat{g}_t(\cdot)$ and $\widehat{h}_t(\cdot)$.

The dynamic programming equations for $t = 1, \dots, T$ are given by

$$
\begin{aligned}
(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}(\boldsymbol{x}_{\boldsymbol{t-1}})) \quad \widehat{Q}_t(x_{t-1}) := \min_{z_t, x_t, y_t} \quad & \widehat{f}_t(x_t, y_t) + \widehat{Q}_{t+1}(x_t) \\
\text{s.t.} \quad & (z_t, x_t, y_t) \in \widehat{M}_t \\
& z_t = x_{t-1}.
\end{aligned}
$$

For the MILP subproblems $(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}(\cdot))$, we obtain the following properties.

Lemma 3.1 *Under assumption* ***(A2)****, subproblem* $(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}(\cdot))$ *has complete recourse and the value function* $\widehat{Q}_t(\cdot)$ *is l.sc. for all* $t = 1, \dots, T$.

The complete recourse follows from the complete recourse of $(\boldsymbol{P}_{\boldsymbol{t}}(\cdot))$ by construction. The l.sc. then follows from Theorem 3.1 in [31].

3.3 The inner loop

In the inner loop of NC-NBD, the MILP subproblems $(\widehat{\boldsymbol{P}}_{\boldsymbol{t}}(\boldsymbol{x}_{\boldsymbol{t-1}}))$ are considered. As stated before, we omit the index ℓ for its discussion.

The copy constraints are crucial for all problems solved in the inner loop. In the forward pass, to ensure Lipschitz continuity, we consider regularized subproblems. The regularization is based on relaxing and penalizing the copy constraints. In the backward pass, to generate cuts, a special Lagrangian dual subproblem is solved based on dualizing the copy constraints. This is effective, since combined with a binary expansion of the state variables, the copy constraints yield a local convexification [55].

3.3.1 Regularization

Lipschitz continuity of the value functions is difficult to ensure in the general non-convex case. However, as shown recently in [52], for l.sc. value functions, it is possible to determine some underestimating Lipschitz continuous function by enhancing the original subproblem with an appropriate penalty function ψ_t. In contrast to the more general regularization approach in [52], we require only so-called *sharp* penalty functions $\psi_t(x_{t-1}) = \|x_{t-1}\|$ to regularize the subproblems $(\boldsymbol{\widehat{P}_t(x_{t-1})})$, for some norm $\|\cdot\|$.

Definition 3.2 (Regularized subproblem and value function) Let $\sigma_t > 0$ for $t = 2, \dots T$, $\sigma_1 = 0$ and define

$$(\boldsymbol{\widehat{P}_t^R(x_{t-1})}) \quad \widehat{Q}_t^R(x_{t-1}) := \min_{z_t, x_t, y_t} \quad \widehat{f}_t(x_t, y_t) + \sigma_t \|x_{t-1} - z_t\| + \widehat{Q}_{t+1}^R(x_t)$$
$$\text{s.t.} \quad (z_t, x_t, y_t) \in \widehat{M}_t.$$

$(\boldsymbol{\widehat{P}_t^R})$ is called *regularized subproblem* and $\widehat{Q}_t^R(\cdot)$ *regularized value function*.

By recursion, this approach yields the regularized optimal value $\widehat{v}^R := \widehat{Q}_1^R(x_0)$ for the first stage. Lemma 3.1 implies that under assumption **(A2)**, the function $\widehat{Q}_t(\cdot)$ is l.sc. Then, the regularized value function $\widehat{Q}_t^R(\cdot)$ has the following properties.

Lemma 3.3 (Proposition 2 in [52]) *For all $t = 1, \dots, T$ we have:*

(a) $\widehat{Q}_t^R(x_{t-1}) \leq \widehat{Q}_t(x_{t-1})$ *for all $x_{t-1} \in X_{t-1}$,*
(b) *Under assumptions **(A1)** and **(A2)**, the regularized value function $\widehat{Q}_t^R(\cdot)$ is Lipschitz continuous on X_{t-1}.*

As also stated in [52], using sharp penalty functions as in Definition 3.2, the penalization is exact for sufficiently large (but finite) $\sigma_t > 0$. For such σ_t, the problems $(\boldsymbol{\widehat{P}})$ and $(\boldsymbol{\widehat{P}^R})$ have the same optimal points and $\widehat{v}^R = \widehat{v}$. This result goes back to [14], in which augmented Lagrangian problems are analyzed for MILPs. It is shown that using sharp penalty functions and a sufficiently large augmenting parameter, strong duality holds. As this result holds for any value of the dual multipliers, it is also valid for the regularized subproblems.

Lemma 3.4 (Proposition 8 in [14]) *Using sharp penalty functions ψ_t, there exist some $\bar{\sigma}_t > 0$ such that the penalty reformulation in $(\boldsymbol{\widehat{P}_t^R(x_{t-1})})$ is exact for all $\sigma_t > \bar{\sigma}_t$.*

Lemma 3.4 indicates that using the regularized subproblems within our decomposition method NC-NBD, we obtain convergence to $\widehat{v}$ in the inner loop. To exploit this, we take the following assumption:

(A3). All $\sigma_t > 0$ are chosen sufficiently large such that Lemma 3.4 is satisfied.

If **(A3)** is not satisfied, σ_t has to be increased gradually in the course of the NC-NBD method to ensure convergence.

3.3.2 Forward pass

In the forward pass of the inner loop we solve approximations of the regularized subproblems ($\boldsymbol{\widehat{P}_t^R(x_{t-1})}$).

For iteration i, the stage-t forward pass problem is defined as follows

$$\begin{aligned}
&(\boldsymbol{\widehat{P}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^i)})\\
&\underline{\widehat{Q}}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^i) := \min_{z_t, x_t, y_t} \quad \widehat{f}_t(x_t, y_t) + \sigma_t \|\widehat{x}_{t-1}^i - z_t\| + \mathfrak{Q}_{t+1}^i(x_t)\\
&\qquad\qquad\qquad\qquad\quad \text{s.t.} \quad (z_t, x_t, y_t) \in \widehat{M}_t,
\end{aligned}$$

for the trial state variable $\widehat{x}_{t-1}^i$, with $\widehat{x}_0^i \equiv 0$. Function $\mathfrak{Q}_{t+1}^i(\cdot)$, in some sense, approximates the value functions $\underline{\widehat{Q}}_{t+1}^{R,i}(\cdot, \mathfrak{Q}_{t+2}^i)$ and $\underline{\widehat{Q}}_{t+1}^i(\cdot, \mathfrak{Q}_{t+2}^i)$. This approximation is constructed in the backward pass, see Sect. 3.3.4. As those value functions are non-convex, the cut approximation $\mathfrak{Q}_{t+1}^i(\cdot)$ is required to be nonlinear and non-convex. However, as we show later, it can be expressed with mixed-integer linear constraints by lifting the problems to a higher dimension. Therefore, in addition to x_t, y_t and z_t, the forward pass problem contains further decision variables, which are hidden in $\mathfrak{Q}_{t+1}^i(\cdot)$ and the piecewise linear relaxations $\widehat{f}_t$, $\widehat{g}_t$ and $\widehat{h}_t$.

Note that expressing $\mathfrak{Q}_{t+1}^i(\cdot)$ by mixed-integer linear constraints with bounded integer variables, the same reasoning as in Lemma 3.1 can be applied to show that $\underline{\widehat{Q}}_t^i(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^i)$ is l.sc. and therefore, $\underline{\widehat{Q}}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^i)$ is Lipschitz continuous.

Even with a mixed-integer linear representation of $\mathfrak{Q}_{t+1}^i(\cdot)$, subproblem ($\boldsymbol{\widehat{P}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^i)}$) is a MINLP due to the regularization. For $\|\cdot\|_1$ or $\|\cdot\|_\infty$, it can be modeled by MILP constraints using standard reformulation techniques for absolute values, though.

The optimal point $(\widehat{z}_t^i, \widehat{x}_t^i, \widehat{y}_t^i)$ of each subproblem ($\boldsymbol{\widehat{P}_t^{R,i}(\widehat{x}_{t-1}^i)}$) is stored and $\widehat{x}_t^i$ is passed to the following stage. Since $\big((\widehat{z}_t^i, \widehat{x}_t^i, \widehat{y}_t^i)\big)_{t=1,\dots,T}$ satisfies all constraints of ($\boldsymbol{\widehat{P}^R}$), after all stages have been considered, an upper bound $\overline{\widehat{v}}$ on the optimal value $\widehat{v}^R$ of the regularized problem can be determined by

$$\overline{\widehat{v}}^i = \min \left\{ \overline{\widehat{v}}^{i-1}, \sum_{t=1}^T \left(\widehat{f}_t(\widehat{x}_t^i, \widehat{y}_t^i) + \sigma_t \|\widehat{x}_{t-1}^i - \widehat{z}_t^i\| \right) \right\}.$$

With assumption **(A3)** and Lemma 3.4, this is also an upper bound to $\widehat{v}$.

3.3.3 Backward pass–Part 1: binary approximation

The aim of the backward pass of an inner loop iteration i is twofold: Firstly, a lower bound $\underline{\widehat{v}}^i$ on $\widehat{v}$ is determined. Secondly, cuts for $Q_t(\cdot)$ are derived to improve and update the current approximation $\mathfrak{Q}_t^i(\cdot)$.

As mentioned before, we use a dynamically refined binary approximation of the state variables and then apply cutting-plane techniques from the SDDiP algorithm [55]. This approximation is based on static binary expansion [21].

Binary expansion can be applied component-wise to some vector x_t. Some integer component $x_{tj} \in \{0, ..., U_j\}$ can be exactly and uniquely expressed as

$$x_{tj} = \sum_{k=1}^{K_{tj}} 2^{k-1} \lambda_{tkj}$$

with variables $\lambda_{tkj} \in \{0, 1\}$ and $K_{tj} = \lfloor \log_2 U_j \rfloor + 1$. Some continuous component $x_{tj} \in [0, U_j]$ can be expressed by discretizing the interval with precision $\beta_{tj} \in (0, 1)$. We then have

$$x_{tj} = \sum_{k=1}^{K_{tj}} 2^{k-1} \beta_{tj} \lambda_{tkj} + r_{tj}$$

with $K_{tj} = \lfloor \log_2 \left(\frac{U_j}{\beta_{tj}} \right) \rfloor + 1$ and some error $r_{tj} \in \left[-\frac{\beta_{tj}}{2}, \frac{\beta_{tj}}{2} \right]$.

For vector x_t, this yields $K_t = \sum_{j=1}^{n_t} K_{tj}$ number of binary variables. Defining an $(n_t \times K_t)$-matrix B_t containing all the coefficients of the binary expansion and collecting all binary variables in one large vector $\lambda_t \in \mathbb{B}^{K_t}$, the binary expansion then can be written compactly as $x_t = B_t \lambda_t + r_t$.

Based on this definition, to generate cuts, for each stage t and iteration i, a binary approximation of $\widehat{x}_{t-1}^i$ is used, *i.e.*, it is replaced by $B_{t-1}\lambda_{t-1}^i$. Note that the approximation is not necessarily exact for continuous components of $\widehat{x}_{t-1}^i$. Therefore, the cuts are not necessarily constructed at the *trial point* $\widehat{x}_{t-1}^i$ but at the deviating *anchor point* $\widehat{x}_{\mathbb{B},t-1}^i := B_{t-1}\lambda_{t-1}^i$.

In the backward pass, we start from the following subproblem, where due to the binary approximation of the state variables, we also adapt the copy constraint to $\lambda_{t-1}^i = \mathfrak{z}_t$ with variables $\mathfrak{z}_t \in [0, 1]^{K_{t-1}}$.

$$(\widehat{\boldsymbol{P}}_{\mathbb{B}t}^{i}(\boldsymbol{\lambda}_{t-1}^{i}, \boldsymbol{\mathfrak{Q}}_{t+1}^{i+1})) \quad \underline{\widehat{Q}}_{\mathbb{B}t}^{i}(\lambda_{t-1}^{i}, \mathfrak{Q}_{t+1}^{i+1}) := \min_{\substack{x_t, y_t, \\ \mathfrak{z}_t, z_t}} \widehat{f}_t(x_t, y_t) + \mathfrak{Q}_{t+1}^{i+1}(x_t)$$
$$\begin{aligned} \text{s.t.} \quad & (z_t, x_t, y_t) \in \widehat{M}_t \\ & z_t = B_{t-1}\mathfrak{z}_t \\ & \mathfrak{z}_t \in [0, 1]^{K_{t-1}} \\ & \mathfrak{z}_t = \lambda_{t-1}^i . \end{aligned}$$

Remark 3.5 Subproblem $(\widehat{\boldsymbol{P}}_{\mathbb{B}t}^{i}(\boldsymbol{\lambda}_{t-1}^{i}, \boldsymbol{\mathfrak{Q}}_{t+1}^{i+1}))$ is equivalent to subproblem $(\widehat{\boldsymbol{P}}_{t}^{i}(\widehat{\boldsymbol{x}}_{\mathbb{B},t-1}^{i}, \boldsymbol{\mathfrak{Q}}_{t+1}^{i+1}))$ because $z_t = B_{t-1}\mathfrak{z}_t = B_{t-1}\lambda_{t-1}^i = \widehat{x}_{\mathbb{B},t-1}^i$.

Asymptotically, *i.e.*, for an infinitely fine binary approximation, the anchor point converges to the actual trial point.

Lemma 3.6 *We have* $\lim_{\beta_{t-1}\to 0} \widehat{x}^i_{\mathbb{B}t-1} = \widehat{x}^i_{t-1}$.

With Lemma 3.6, asymptotically, the cuts are constructed at $\widehat{x}^i_{t-1}$. While this is not directly useful in practice, since it requires an infinite number of binary variables, it also implies that for componentwise sufficiently small $\beta_{t-1} \in (0,1)$, the cuts are constructed very close to $\widehat{x}^i_{t-1}$. As NC-NBD constructs Lipschitz continuous cuts, this guarantees a sufficiently good approximation of the value function at $\widehat{x}^i_{t-1}$, as we show in Sect. 4.

Importantly, in our framework the binary approximation is only applied temporarily to derive cuts, while the state variables x_{t-1} in the forward pass remain continuous. In other words, the anchor points determine where cuts can be *constructed*, but do not limit where they can be *evaluated*. This is a crucial difference to applying a *static* binary expansion, as suggested in the original SDDiP work to solve MILPs with continuous state variables [55].

Moreover, let us emphasize again that applying such static approximation is not appropriate in our inner loop, as the obtained lower bounds are not guaranteed to be valid for $\widehat{v}$ or v. Similarly, the obtained cuts are not guaranteed to be valid for $\widehat{Q}_t(\cdot)$ or $Q_t(\cdot)$, and therefore cannot be re-used within the outer loop. Our proposed inner loop method does not share these issues. We follow a *dynamic* approach where the binary precision is dynamically refined if required and, as we show later, all cuts remain valid with later refinements.

3.3.4 Backward pass–Part 2: cut generation

As proposed in [55], the copy constraint is dualized to generate cuts. Applied to our context, the following Lagrangian dual subproblem has to be solved

$$(\boldsymbol{D}^{i}_{\mathbb{B}t}(\boldsymbol{\lambda}^{i}_{t-1}, \boldsymbol{\mathfrak{Q}}^{i+1}_{t+1})) \quad \max_{\|\pi_t\|_* \le l_t} \quad \mathcal{L}^i_{\mathbb{B}t}(\pi_t, \mathfrak{Q}^{i+1}_{t+1}) + \pi_t^\top \lambda^i_{t-1},$$

where $\mathcal{L}^i_{\mathbb{B}t}(\cdot)$ denotes the Lagrangian function for π_t defined by

$$\begin{aligned} \mathcal{L}^i_{\mathbb{B}t}(\pi^i_t, \mathfrak{Q}^{i+1}_{t+1}) := \min_{x_t, y_t, \mathfrak{z}_t, z_t} \quad & \widehat{f}_t(x_t, y_t) + \mathfrak{Q}^{i+1}_{t+1}(x_t) - \pi_t^\top \mathfrak{z}_t \\ \text{s.t.} \quad & (z_t, x_t, y_t) \in \widehat{M}_t \\ & z_t = B_{t-1}\mathfrak{z}_t \\ & \mathfrak{z}_t \in [0,1]^{K_{t-1}} \end{aligned}$$

and $\|\cdot\|_*$ denotes the dual norm to the norm used in the regularized forward pass problems $(\widehat{\boldsymbol{P}}^{R,i}_t(\widehat{\boldsymbol{x}}^i_{t-1}, \boldsymbol{\mathfrak{Q}}^i_{t+1}))$.

A linear (optimality) cut in binary space $\{0,1\}^{K_{t-1}}$ is then given by

$$\phi_{\mathbb{B}t}(\lambda_{t-1}) := \underbrace{\mathcal{L}^i_{\mathbb{B}t}(\pi^i_t, \mathfrak{Q}^{i+1}_{t+1})}_{=:c^i_t} + (\pi^i_t)^\top \lambda_{t-1}, \tag{1}$$

where π_t^i is an optimal solution of the Lagrangain dual subproblem $(\boldsymbol{D}^i_{\mathbb{B}t}(\boldsymbol{\lambda}^i_{t-1}, \boldsymbol{\mathfrak{Q}}^{i+1}_{t+1}))$. Those Lagrangian cuts are introduced in [55] and identified to be finite, valid and tight in the SDDiP setting. In our setting, we obtain the following validity result.

Lemma 3.7 *Let $\widehat{Q}_{\mathbb{B}t}(\cdot)$ denote the MILP value function of stage t with additional binary approximations. Then,*

(a) *for all $\lambda_{t-1} \in [0,1]^{K_{t-1}}$*

$$\widehat{Q}_{\mathbb{B}t}(\lambda_{t-1}) \geq \phi_{\mathbb{B}t}(\lambda_{t-1}),$$

(b) *for all x_{t-1}*

$$\widehat{Q}_t(x_{t-1}) \geq \phi_{\mathbb{B}t}(\lambda_{t-1})$$

for any $\lambda_{t-1} \in [0,1]^{K_{t-1}}$, such that $x_{t-1} = B_{t-1}\lambda_{t-1}$.

Lemma 3.7 a) follows directly from the validity proof for the SDDiP cuts, which does also hold for $\lambda_{t-1} \in [0,1]^{K_{t-1}}$ instead of $\lambda_{t-1} \in \{0,1\}^{K_{t-1}}$ (see Theorem 3 in [55]). Part b) then follows using similar arguments as in Remark 3.5. Hence, $\phi_{\mathbb{B}t}$ is, in fact, a valid cut in $[0,1]^{K_{t-1}}$. This enables us to obtain valid under-approximations also for those points, which are not exactly approximated by the current binary expansion. As it refers to an outer approximation, $\widehat{Q}_t(\cdot)$ underestimates the original MINLP value function $Q_t(\cdot)$. Thus, the obtained cuts are valid for $Q_t(\cdot)$ as well.

Contrary to [55], but following [52], we bound the dual variable π_t in the Lagrangian dual subproblem. Therefore, tightness for $\underline{\widehat{Q}}^i_{\mathbb{B}t}(\cdot, \mathfrak{Q}^{i+1}_{t+1})$ is not guaranteed. However, the cuts are at least guaranteed to overestimate the value function $\underline{\widehat{Q}}^{R,i}_{\mathbb{B}t}(\cdot, \mathfrak{Q}^{i+1}_{t+1})$ at λ^i_{t-1}. This value function is obtained by regularizing $\underline{\widehat{Q}}^i_{\mathbb{B}t}(\cdot, \mathfrak{Q}^{i+1}_{t+1})$ in the binary space using the same norm as in the forward pass problem. By careful choice of the regularization factor, then, also the regularized value function $\underline{\widehat{Q}}^{R,i}_t(\cdot, \mathfrak{Q}^{i+1}_{t+1})$ in the original space is overestimated at $x^i_{\mathbb{B},t-1}$. This result is formalized in the following lemma.

Lemma 3.8 *Assume that we use $\|\cdot\|_1$ for regularization and its dual norm $\|\cdot\|_\infty$ for bounding the dual multipliers. Then, as long as $l_t \geq \sigma_t \|B_{t-1}\|$, where the latter denotes the induced matrix norm of B_{t-1}, we have*

$$\phi_{\mathbb{B}t}(\lambda^i_{t-1}) \geq \underline{\widehat{Q}}^{R,i}_{\mathbb{B}t}(\lambda^i_{t-1}, \mathfrak{Q}^{i+1}_{t+1}) \geq \underline{\widehat{Q}}^{R,i}_t(x^i_{\mathbb{B},t-1}, \mathfrak{Q}^{i+1}_{t+1}).$$

Proof See Appendix A. □

Remark 3.9 The induced matrix norm $\|B_{t-1}\|$ depends on the chosen precision of the binary approximation. It can be bounded from above independent of the precision, *e.g.*, $\|B_{t-1}\|_1 \leq U_{t-1,\max}$ with $U_{t-1,\max}$ the largest component of the upper bounds in X_{t-1}.

3.3.5 Backward pass–Part 3: cut projection

Solving the forward pass problems ($\widehat{\boldsymbol{P}}^{R,i}_t(\widehat{\boldsymbol{x}}^i_{t-1}, \boldsymbol{\mathfrak{Q}}^i_{t+1})$) and the backward pass dual problems ($\boldsymbol{D}^i_{\mathbb{B}t}(\boldsymbol{\lambda}^i_{t-1}, \boldsymbol{\mathfrak{Q}}^{i+1}_{t+1})$) requires expressing the cut approximation $\mathfrak{Q}^i_{t+1}(\cdot)$ in the original state variables x_t. Recall that the computed cut $\phi_{\mathbb{B},t+1}(\cdot)$ is a function of $[0,1]^{K_t}$.

According to Lemma 3.7 a), the obtained cuts $\phi_{\mathbb{B},t+1}(\cdot)$ are not only valid for all binary points, but for all values in $[0,1]^{K_t}$. Allowing for $\lambda_t \in [0,1]^{K_t}$ in the binary approximation, there exist infinitely many combinations of λ_t to exactly describe some point $x_t \in X_t$, though. Therefore, following from Lemma 3.7 b), one cut in binary space entails infinitely many underestimators of $Q_{t+1}(\cdot)$ at x_t in the original space X_t. Including infinitely many inequalities in $\mathfrak{Q}_{t+1}(\cdot)$ is computationally infeasible. Instead, we consider the pointwise maximum of the projection of the cuts to X_t. That way, only the best underestimation for each point x_t is taken into account. In doing so, we obtain a nonlinear, *i.e.*, piecewise linear, cut in the original state space. For simplicity, in the following, by *cut projection* we always mean the pointwise maximum of the actual projection.

The projection of some cut $\phi_{\mathbb{B},t+1}(\cdot)$ to X_t can be described as the value function

$$\phi_{t+1}(x_t) := \max_{\lambda_t} \left\{ c_{t+1} + (\pi_{t+1})^\top \lambda_t \, : \, B_t \lambda_t = x_t, \lambda_t \le e, \lambda_t \ge 0 \right\} \tag{2}$$

of a linear program where e denotes a vector of ones of dimension K_t. The dual problem to (2) yields

$$\phi^D_{t+1}(x_t) := \min_{\eta_t,\mu_t} \left\{ c_{t+1} + x_t^\top \eta_t + e^\top \mu_t \; : \; B_t^\top \eta_t + I \mu_t \ge \pi_{t+1}, \mu_t \ge 0 \right\}. \tag{3}$$

Note that the dual feasible region does not depend on x_t and has a finite number of extreme points. Therefore, the cut projection is piecewise linear and concave.

As problem (2) is feasible and bounded for any $x_t \in X_t$, this also holds for the dual problem (3). Therefore, in a dual optimal solution, η_t and μ_t are bounded. Note that this bound may change with the binary approximation precision β_t, though, and that, if we would generate tight cuts for $\underline{\widehat{Q}}^i_{t+1}(\cdot, \mathfrak{Q}^{i+1}_{t+2})$, those cuts may become infinitely steep close to discontinuities. However, as we can bound π_t in the Lagrangian dual subproblem independent of β_t, see Remark 3.9, and thus construct cuts which at least overestimate the regularized value function $\underline{\widehat{Q}}^{R,i}_{t+1}(\cdot, \mathfrak{Q}^{i+1}_{t+2})$ at the anchor point $x^i_{\mathbb{B},t}$, such cases should be ruled out.

We formalize this by assuming the existence of a global bound for η_t.

(A4). There exists some $\rho_t > 0$, such that for all $t = 1, \dots, T$, any binary precision β_t and any x_t, the optimal dual variable η_t in problem (3) can be bounded, *i.e.*, $\|\eta_t\| \le \rho_t$.

For example, if we obtain cuts which are, in fact, tight for $\underline{\widehat{Q}}^{R,i}_{t+1}(\cdot, \mathfrak{Q}^{i+1}_{t+2})$ at $x^i_{\mathbb{B},t}$ and consider only basic solutions in the Lagrangian dual, the gradient of the cuts is bounded by σ_{t+1}. With Assumption **(A4)** it follows that the linear cuts $\phi_{\mathbb{B},t+1}(\cdot)$

derived in the binary space yield a nonlinear, but Lipschitz continuous projection $\phi_{t+1}(\cdot)$ in the original space.

To express this projection by mixed-integer linear constraints, we use the KKT conditions to problems (2) and (3). To emphasize that these conditions are considered for the projection of one specific cut r (the index denoting the r-th cut constructed), we index all occurring variables and coefficients by r.

$$-\pi^r_{t+1} - \nu^r_t + \mu^r_t + (B^r_t)^\top \eta^r_t = 0 \tag{4}$$
$$B^r_t \lambda^r_t - x_t = 0 \tag{5}$$
$$\lambda^r_t \geq 0 \tag{6}$$
$$\lambda^r_t - e \leq 0 \tag{7}$$
$$\nu^r_t, \mu^r_t \geq 0 \tag{8}$$
$$-(\nu^r_t)^\top \lambda^r_t = 0 \tag{9}$$
$$(\mu^r_t)^\top (\lambda^r_t - e) = 0. \tag{10}$$

The complementary slackness constraints (9) and (10) are nonlinear, but componentwise can be expressed linearly using a Big-$\mathcal{M}$ formulation (alternatively, SOS-1 constraints may be used):

$$\lambda^r_{tk} \leq \mathcal{M}_{1k}\omega^r_{tk}, \quad \nu^r_{tk} \leq \mathcal{M}_{2k}(1-\omega^r_{tk}), \quad \omega^r_{tk} \in \{0,1\} \tag{11}$$
$$\lambda^r_{tk} - 1 \geq \mathcal{M}_{3k}u^r_{tk}, \quad \mu^r_{tk} \leq \mathcal{M}_{4k}(1-u^r_{tk}), \quad u^r_{tk} \in \{0,1\} \tag{12}$$

For all components k, we can choose $\mathcal{M}_{1k} = 1$ and $\mathcal{M}_{3k} = -1$ due to $\lambda_{tk} \in [0, 1]$. Moreover, using **(A4)**, we are able to obtain explicit choices for $\mathcal{M}_{2k}$ and $\mathcal{M}_{4k}$ as well.

Lemma 3.10 *Under (A4), there exist explicit, finite bounds for ν^r_{tk} and μ^r_{tk}.*

Proof See Appendix B. □

The cut approximation $\mathfrak{Q}^{i+1}_{t+1}(\cdot)$ is then defined as the maximum of all cuts $\phi^r_{\mathbb{B},t+1} = c^r_{t+1} + (\pi^r_{t+1})^\top \lambda^r_t$ where the variable λ^r_t satisfies the linearized KKT conditions (4)–(8) and (11)–(12) for the r-th cut. With Assumption **(A4)**, it is Lipschitz continuous.

Lemma 3.11 *The cut approximation $\mathfrak{Q}_{t+1}(\cdot)$ is Lipschitz continuous in X_t with Lipschitz constant ρ_t.*

The cut projection requires to introduce the variables $\lambda^r_t, \nu^r_t, \mu^r_t, w^r_t, u^r_t, \eta^r_t$ and constraints (4)–(8) and (11)–(12) for each cut r. In particular, each cut is associated with a variable $\lambda^r_t \in [0, 1]^{K^r_t}$ where K^r_t corresponds to the number of binary variables at the time of the cut's generation. This increases the problem size considerably, as the number of variables and constraints to be added per cut is in $\mathcal{O}\big(n_t \log\big(\frac{1}{\beta_t}\big)\big)$. In return, it ensures that cuts do not have to be generated from scratch after each refinement.

3.3.6 Stopping and refining

At the end of the backward pass, a lower bound $\widehat{\underline{v}}^i$ is determined by solving the first-stage subproblem $(\widehat{\boldsymbol{P}}_{\mathbf{1}}^{\boldsymbol{i}}(\mathbf{0}, \boldsymbol{\mathfrak{Q}}_{\mathbf{2}}^{\boldsymbol{i+1}}))$. Here, no Lagrangian dual is solved, since no cuts have to be derived. The lower bound is non-decreasing because the cut approximation is only improved.

If the updated bounds are sufficiently close to each other, *i.e.*, if

$$\widehat{\overline{v}}^i - \widehat{\underline{v}}^i \leq \widehat{\varepsilon}$$

for some predefined tolerance $\widehat{\varepsilon} > 0$, an approximately optimal point of problem $(\widehat{\boldsymbol{P}})$ has been determined. We show in the following section that this is the case after finitely many iterations i.

If the gap between the bounds does not meet the stopping criteria yet, two cases are possible: In the first case, the algorithm has not determined the best possible approximation for the given binary approximation precision, yet. New cuts have been determined in iteration i such that the lower bound $\widehat{\underline{v}}^i$ has been updated, and the forward solution will change in iteration $i + 1$ as the previous one is cut away.

In the second case, despite not meeting the stopping criterion, the forward solution does not change at the beginning of iteration $i + 1$. This case is related to the binary approximation. It can occur if the binary approximation is too coarse and therefore, for all t, the determined cuts at $\widehat{x}_{\mathbb{B}t}^i$ do not improve the approximation at $\widehat{x}_t^i$. Moreover, it can occur if in subsequent iterations the same cuts are constructed, since $\widehat{x}_{B,t-1}^i = \widehat{x}_{B,t-1}^{i+1}$. Finally, it can also occur if all possible cuts have been generated: For a fixed binary approximation, there exist only finitely many points $\widehat{x}_{\mathbb{B}t}$. If we restrict the Lagrangian dual subproblem to basic solutions, then only finitely many different cuts can be determined [55].

In the second case, at the beginning of the backward pass of iteration i, the binary approximation is refined. The refinement is computed by increasing K_{tj} by +1 for all components j and all stages t with

$$\beta_{tj} = \frac{U_j}{\sum_{k=1}^{K_{tj}} 2^{k-1}}.$$

For simplicity, we refine in Algorithm 1 all stages and components equally by +1. Note that each refinement requires the introduction of an additional vector λ_t, as described in the previous subsection.

As all previously generated cuts have been projected to the original space X_t, they remain valid and have not to be recomputed when refining the binary approximation. This is computationally important.

3.4 The outer loop

3.4.1 The outer loop problem

Once the inner loop is left, we set $\widehat{\underline{v}}^{\ell} := \widehat{\underline{v}}^{\ell i}, \overline{\widehat{v}}^{\ell} := \overline{\widehat{v}}^{\ell i}$ and $\mathfrak{Q}_t^{\ell}(\cdot) := \mathfrak{Q}_t^{\ell i}(\cdot)$ for all $t = 2, ..., T$. Note that $\overline{\widehat{v}}^{\ell}$ is not guaranteed to be a valid upper bound for v because $\widehat{v}^{\ell} \leq v$. Moreover, we set $\big((\widehat{z}_t^{\ell}, \widehat{x}_t^{\ell}, \widehat{y}_t^{\ell})\big)_{t=1,...,T} := \big((\widehat{z}_t^{\ell i}, \widehat{x}_t^{\ell i}, \widehat{y}_t^{\ell i})\big)_{t=1,...,T}$.

To approximate the optimal value v of $(\boldsymbol{P})$, we solve subproblems

$$(\boldsymbol{P}_t^{\ell}(\boldsymbol{x}_{t-1}^{\ell}, \mathfrak{Q}_{t+1}^{\ell})) \quad \underline{Q}_t^{\ell}(x_{t-1}^{\ell}, \mathfrak{Q}_{t+1}^{\ell}) := \min_{z_t, x_t, y_t} \; f_t(x_t, y_t) + \mathfrak{Q}_{t+1}^{\ell}(x_t)$$
$$\text{s.t.} \quad (z_t, x_t, y_t) \in M_t$$
$$z_t = x_{t-1}^{\ell}$$

in a forward manner for $t = 1, \ldots, T$ with $x_0^{\ell} \equiv 0$ and $x_t^{\ell} := x_t$, where x_t is an optimal solution of $(\boldsymbol{P}_t^{\ell}(\boldsymbol{x}_{t-1}^{\ell}, \mathfrak{Q}_{t+1}^{\ell}))$ for t. Here, we exploit that the cut approximation $\mathfrak{Q}_t^{\ell}(\cdot)$, constructed in the inner loop, is valid for $Q_t(\cdot)$ by design as well. By solving these subproblems, we obtain a feasible solution $\big((z_t^{\ell}, x_t^{\ell}, y_t^{\ell})\big)_{t=1,...,T}$ for $(\boldsymbol{P})$ and we can determine a valid upper bound for v as $\overline{v}^{\ell} = \min\left\{\overline{v}^{\ell-1}, \sum_{t=1}^{T} f_t(x_t^{\ell}, y_t^{\ell})\right\}$.

The subproblems $(\boldsymbol{P}_t^{\ell}(\boldsymbol{x}_{t-1}^{\ell}, \mathfrak{Q}_{t+1}^{\ell}))$ are non-convex MINLP problems. This means that in order to solve the original non-convex problem $(\boldsymbol{P})$, easier, but still non-convex subproblems have to be solved to optimality for each stage t in each outer loop iteration ℓ. This might be a hard challenge by itself. We make the following assumption for the remainder of this article:

(**A5**). An oracle exists that is able to solve subproblems $(\boldsymbol{P}_t^{\ell}(\boldsymbol{x}_{t-1}^{\ell}, \mathfrak{Q}_{t+1}^{\ell}))$ to global optimality.

In case that no such global optimization algorithm is available, one can solve appropriate inner approximations of $(\boldsymbol{P}_t^{\ell}(\boldsymbol{x}_{t-1}^{\ell}, \mathfrak{Q}_{t+1}^{\ell}))$, which are improved in the course of the algorithm.

If $\overline{v}^{\ell} - \widehat{\underline{v}}^{\ell} \leq \varepsilon$, then NC-NBD terminates and $\big((z_t^{\ell}, x_t^{\ell}, y_t^{\ell})\big)_{t=1,...,T}$ is an ε-optimal solution for $(\boldsymbol{P})$. Otherwise, the cut approximations $\mathfrak{Q}_{t+1}^{\ell}(\cdot)$ are not sufficiently good underestimators for the true value functions, even though they give a good approximation of $\widehat{Q}_t^{\ell}(\cdot)$. This implies that the piecewise linear relaxations have to be improved. Instead of refining them everywhere, they are refined dynamically where it is promising, *i.e.*, in a neighborhood of the approximate optimal solution $\big((\widehat{z}_t^{\ell}, \widehat{x}_t^{\ell}, \widehat{y}_t^{\ell})\big)_{t=1,...,T}$ of $(\widehat{\boldsymbol{P}}^{\ell})$. In refining the piecewise linear relaxations in its neighborhood, the current solution can be excluded in the next inner loop and the lower bound $\widehat{\underline{v}}^{\ell}$ improves.

Remark 3.12 Instead of $\widehat{\underline{v}}^{\ell}$, an even better lower bound for v is given by the optimal value of the first stage subproblem $(\boldsymbol{P}_1^{\ell}(\boldsymbol{x}_0^{\ell}, \mathfrak{Q}_2^{\ell}))$.

3.4.2 Refining the piecewise linear relaxations

The refinement consists of two steps: (1) the piecewise linear approximations are refined and (2) the corresponding MILP ($\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}}$) is updated – in such a way that the new MILP ($\widehat{\boldsymbol{P}}^{\boldsymbol{\ell+1}}$) again yields a relaxation of ($\boldsymbol{P}$).

Different strategies are possible to achieve this. For a thorough overview, we refer to [18]. In the following, we make use of a specific adaptive refinement scheme for triangulations from [9] for any nonlinear function $\gamma_t \in \Gamma_t$. The given piecewise linear approximation at iteration ℓ is defined by a triangulation $\mathcal{T}$ of $X_{t-1} \times X_t \times Y_t$ (or a subspace) and the corresponding function values of γ_t. Instead of refining this triangulation everywhere now, the main idea is to only refine it in a neighborhood of $\big((\widehat{z}_t^\ell, \widehat{x}_t^\ell, \widehat{y}_t^\ell)\big)_{t=1,\dots,T}$. Therefore, first, the simplex in $\mathcal{T}$ containing this point is identified. It is then divided by a longest-edge bisection, yielding a refined triangulation, for which a new MILP model can be set up. As proven in [9], this refinement strategy has some favorable properties with respect to convergence, see Sect. 4.2.

It is important that the obtained relaxation ($\widehat{\boldsymbol{P}}^{\boldsymbol{\ell+1}}$) is tighter than ($\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}}$) so that the corresponding value functions improve monotonically. This is required to ensure that previously generated cuts remain valid in later iterations. For concave functions, this is always satisfied using the presented refinement strategy. For other functions, *e.g.*, convex ones, a more careful determination of the relaxation is required or the MILP models for earlier relaxations have to be kept instead of being replaced. For our theoretical results, it is sufficient that such monotonically improving relaxations can always be determined.

After refining the piecewise linear relaxations, a new iteration $\ell + 1$ is started, beginning with the inner loop.

4 Convergence results

In this section, we prove the convergence of the NC-NBD algorithm. We start proving the convergence of the inner loop to an optimal solution of ($\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}}$) based on some results on the binary refinements. Afterwards, we prove that the outer loop converges to an optimal solution of the original problem ($\boldsymbol{P}$).

4.1 Convergence of the inner loop

As explained in Sect. 3.3.3, within NC-NBD the cuts are not generated at the trial points $\widehat{x}_{t-1}^i$, but instead at anchor points $\widehat{x}_{\mathbb{B},t-1}^i := B_{t-1}\lambda_{t-1}^i$. This means that the generated cuts, and with that also the cut approximations $\mathfrak{Q}_t(\cdot)$, implicitly depend on the binary approximation precision β_t.

However, Lemma 3.6 implies that $\widehat{x}_{t-1}^i$ and $\widehat{x}_{\mathbb{B},t-1}^i$ should become equal asymptotically in the refinements of the binary approximations. Therefore, asymptotically, the cuts are guaranteed to overestimate $\underline{\widehat{Q}}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^{i+1})$ and, due to their Lipschitz continuity, for some sufficiently small precision, they are at least $\varepsilon_{\mathbb{B}_t}$-close. This, in turn, leads to convergence of the inner loop, as we formalize and prove below.

Prior to this, let us introduce two useful ideas. Firstly, using the Lipschitz continuity results from Lemma 3.3, page 12 and Lemma 3.11, we can bound the cut approximation error in $\widehat{x}_{t-1}^i$ as follows:

Lemma 4.1 *With Assumption **(A4)**, for any iteration i and stage t it follows*

$$\mathfrak{Q}_t^{i+1}(\widehat{x}_{t-1}^i) - \underline{Q}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^{i+1}) \geq -(L_t^R + \rho_t)\|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\|.$$

Proof See Appendix C. □

Secondly, for any stage t and any fixed binary approximation, if we restrict to basic solutions in the Lagrangian duals, only finitely many different realizations of cut approximations $\mathfrak{Q}_t(\cdot)$ can be generated. Thus, after a finite number of iterations, the binary approximation is refined. Assuming that the inner loop does not terminate for $\widehat{\varepsilon} = 0$, we can then observe infinitely many such refinements. Hence, with $j \to \infty$, we also get $\beta_t \to 0$ for all $t = 1, \ldots, T$.

Now, we address convergence of the inner loop of NC-NBD to an optimal solution of $(\widehat{\boldsymbol{P}})$. First, we provide a preliminary result, which can be proven by backward induction using Lemmas 3.11 and 4.1.

Lemma 4.2 *Suppose that the inner loop does not terminate for $\widehat{\varepsilon} = 0$. Then, the infinite sequence of forward pass trial solutions $(\widehat{x}^i)_{i\in\mathbb{N}}$ possesses some cluster point $\widehat{x}^*$ with a corresponding convergent subsequence $(\widehat{x}^{i_j})_{j\in\mathbb{N}}$. This subsequence satisfies*

$$\lim_{j\to\infty} \mathfrak{Q}_t^{i_j}(\widehat{x}_{t-1}^{i_j}) \geq \widehat{Q}_t^R(\widehat{x}_{t-1}^*). \tag{13}$$

Proof See Appendix D. □

Using this result, convergence can be proven.

Theorem 4.3 *Suppose that the inner loop does not terminate for $\widehat{\varepsilon} = 0$. Then, the sequence $(\underline{\widehat{v}}^i)_{i\in\mathbb{N}}$ of lower bounds determined by the algorithm converges to $\widehat{v}$ and every cluster point of the sequence of feasible forward pass solutions generated by the inner loop is an optimal solution of $(\widehat{\boldsymbol{P}})$.*

Note that with a similar argument it can be shown that the inner loop terminates as soon as $\mathfrak{Q}_t^i(\widehat{x}_{t-1}^i) \geq \widehat{Q}_t^R(\widehat{x}_{t-1}^i)$ for all $t = 2, \ldots, T$.

Considering that the inner loop is integrated into an outer loop improving the MILP approximations of $(\boldsymbol{P})$, infinite convergence is not directly useful. Moreover, infinitely many binary refinements are not computationally feasible. However, we can deduce that an approximately optimal solution of $(\widehat{\boldsymbol{P}})$ is determined in a finite number of iterations.

Corollary 4.4 *For any stopping tolerance $\widehat{\varepsilon} > 0$, the inner loop stops in a finite number of iterations with an $\widehat{\varepsilon}$-optimal solution of $(\widehat{\boldsymbol{P}})$.*

4.2 Convergence of the outer loop

We start our convergence analysis of the outer loop with a feasibility result for the solutions determined in the inner loop, which follows from the convergence results in [9]. The main idea is that, as the domain is bounded for all functions $\gamma \in \Gamma$, using a longest-edge bisection, after a finite number of steps, all considered simplices become sufficiently small (since in the worst case all simplices have been refined sufficiently often).

Lemma 4.5 ([9]) *Using longest-edge bisection for the piecewise linear relaxation refinements within NC-NBD yields optimal solutions $\big((\widehat{z}_t^\ell, \widehat{x}_t^\ell, \widehat{y}_t^\ell)\big)_{t=1,\dots,T}$ for $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ in the inner loop, which*

(a) *are approximately feasible for $(\boldsymbol{P})$ after a finite number of steps ℓ,*
(b) *become feasible for $(\boldsymbol{P})$ asymptotically in the number of refinements ℓ.*

Next we show that with decreasing the feasibility error also the deviation in the optimal value between $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ and $(\boldsymbol{P})$ can be controlled.

As a preliminary result, we obtain that for sufficiently small feasibility tolerances $\widehat{\varepsilon}_\gamma$ for all $\gamma \in \Gamma$, there exists a neighborhood of the optimal solution $\widehat{\mathrm{x}}^\ell := \big((\widehat{z}_t^\ell, \widehat{x}_t^\ell, \widehat{y}_t^\ell)\big)_{t=1,\dots,T}$ of problem $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ containing a feasible point $\widetilde{\mathrm{x}}^\ell := \big((\widetilde{z}_t^\ell, \widetilde{x}_t^\ell, \widetilde{y}_t^\ell)\big)_{t=1,\dots,T}$ of $(\boldsymbol{P})$. This follows primarily from the continuity of all functions in $(\boldsymbol{P})$.

Lemma 4.6 *For any $\delta > 0$, there exists some $\hat{\ell} \in \mathbb{N}$ such that for all $\ell \geq \hat{\ell}$ there exists some feasible point $\widetilde{\mathrm{x}}^\ell$ of $(\boldsymbol{P})$ with*
$\|\widetilde{\mathrm{x}}^\ell - \widehat{\mathrm{x}}^\ell\|_2 \leq \delta.$

Applying Lemma 4.6 yields the following result with respect to the deviation in the optimal value between $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ and $(\boldsymbol{P})$.

Theorem 4.7 *There exists some $\hat{\hat{\ell}} \in \mathbb{N}$ such that for all $\ell \geq \hat{\hat{\ell}}$ we have*

$$0 \leq v - \widehat{v}^\ell \leq \varepsilon.$$

Proof The proof makes use of the Lipschitz continuity of f_t, Lemma 4.5 and Lemma 4.6 to bound $v - \widehat{v}^\ell$ from above by $L_f\delta + \sum_{t=1}^T \widehat{\varepsilon}_{f_t}$ (with $\widehat{\varepsilon}_{f_t}$ deduced from $\widehat{\varepsilon}_\gamma$ with $\gamma = f_t$). The assertion then follows with $\varepsilon := L_f\delta + \sum_{t=1}^T \widehat{\varepsilon}_{f_t}$. For a detailed proof see Appendix F. □

We obtain the central convergence result for NC-NBD:

Theorem 4.8 *NC-NBD has the following convergence properties:*

(a) *Assume that for all ℓ the MILP $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ is solved to global optimality in a finite number of steps. Then, if NC-NBD does not terminate with $\varepsilon = 0$, the sequence of lower bounds $(\underline{v}^\ell)_{\ell\in\mathbb{N}}$ converges to v and the outer loop solutions converge to an optimal solution of $(\boldsymbol{P})$.*

(b) *Let* $\varepsilon = \widehat{\varepsilon} > 0$. *Then, if NC-NBD does not terminate, it converges to an* $\widehat{\varepsilon}$*-optimal solution of* $(\boldsymbol{P})$.

(c) *For any* $\varepsilon > \widehat{\varepsilon} > 0$, *NC-NBD terminates with an* ε*-optimal solution of* $(\boldsymbol{P})$ *after a finite number of steps.*

Proof See Appendix G. □

5 Computational results

We illustrate the adequacy of using the NC-NBD to solve multistage non-convex MINLPs by applying it to moderate-sized instances of a unit commitment problem. NC-NBD is implemented in Julia-1.5.3 [7] based on the `SDDP.jl` package [12], which provides an existing implementation for SDDP. More implementation details are presented in Appendix H.

The considered unit commitment problem is formally described in detail in Appendix I. Importantly, the considered problem contains binary state variables, but also continuous state variables, such that a binary approximation of the state variables is required in the backward pass of NC-NBD. Additionally, all instances contain a nonlinear function in the objective. In the *base instances*, we consider a concave quadratic emission cost curve in the objective. In the *valve-point instances*, additionally, we consider a non-convex fuel cost curve with a sinusoidal term. In both cases, we analyze instances with 2 to 36 stages and 3 to 10 generators, resulting in 6 to 20 state variables. More details on our parameter settings and the complete test results for all instances are presented in Appendix I.

The results show that NC-NBD succeeds to solve multistage non-convex MINLPs with a moderate number of stages and state variables to (approximate) global optimality. It converges to the globally minimal point for each of the instances and, considering our 1% tolerance, terminates with valid upper and lower bounds for v.

For the base instances, we observe long computation times of several minutes compared to state-of-the-art solvers for MINLPs, which solve the problems in a few seconds, though. We address some of the reasons and possible solutions for this behavior at the end of this section. As the results for problems with a small number of state variables, but many stages look most promising, for our valve-point instance tests we focus on such instances.

For these instances, the sinusoidal terms in the objective exclude many existing general purpose solvers from application. A sample of the obtained results is presented in Table 1, for the complete ones, see Appendix I. The results show that NC-NBD is less efficient than existing solvers for problems with few stages, but becomes competitive with a larger number of stages. Especially for the instances with 36 stages, conventional solvers have difficulties closing the optimality gap while NC-NBD manages to solve the instances in reasonable time.

These results confirm that NC-NBD should be best suited for multistage problems with a large number of stages, but a relatively small number of state variables, as the obtained subproblems remain sufficiently small even for a larger number of iterations, while general purpose solvers may start to struggle due to the combination of many

Table 1 Solution times in sec. for valve-point instances with three different demand time series

T	# of Gen.	Demand	LINDOGlobal	Couenne	NC-NBD
10	3	v1	201	29	1074
16	3	v1	135	273	760
24	3	v1	–	4427	1408
24	3	v2	1562	647	771
24	4	v1	–	349	1220
24	4	v2	2603	–	4191
36	4	v1	–	–	6816
36	4	v2	2733	7501	3646
36	4	v3	–	–	3497

stages and nonlinear terms. Therefore, NC-NBD may also be useful for stochastic programs where the deterministic equivalent becomes computationally infeasible for monolith approaches. To investigate this is left for future research.

While some of the test results look promising, we still see substantial potential for improvement. This should also help to make NC-NBD more efficient and competitive for problems with a larger number of state variables. It is a known drawback of SDDiP [55], which is inherited by NC-NBD, that existing methods to solve the Lagrangian dual problems may take extremely long to converge. To some extent, this could possibly be mitigated by additionally using different cut types such as strengthened Benders cuts [55], thus, only constructing tight cuts every few iterations. Yet, developing more efficient solution methods is an important open research question.

Additionally, with each projected cut, the considered subproblems become considerably larger. While we implemented a simple cut selection scheme to reduce the subproblem size, more sophisticated approaches may be required to keep the subproblems tractable for applications with many state variables.

Finally, so far, we assume that the outer loop MINLPs are solved to global optimality **(A5)** directly. In a more efficient implementation of NC-NBD, these subproblems should be approximated as well.

6 Conclusion

We propose the non-convex nested Benders decomposition (NC-NBD) method to solve multistage non-convex MINLPs. The method is based on combining piecewise linear relaxations, regularization, binary approximation and cutting-plane techniques in a unique and dynamic way. We are able to prove that NC-NBD is guaranteed to compute an ε-optimal solution for the original MINLP in a finite number of steps. Computational results for some moderate-sized instances of a unit commitment problem demonstrate its applicability to multistage problems.

We require all constraints to be continuous and the objective function to be Lipschitz continuous, which are common assumptions in nonlinear optimization. We also assume complete recourse for the multistage problem. Moreover, the regularization factors are assumed to be sufficiently large to ensure exact penalization in the regularized subproblems. If this is not the case, the factors can be increased gradually within NC-NBD.

In contrast to previous approaches to solve multistage non-convex problems, we do not require the value functions to be monotonic in the state variables [36] and allow the state variables to enter not only the objective function, but also the constraints. The latter avoids the assumption of oracles to handle indicator functions [52].

In NC-NBD, we combine dynamic binary approximation of the state variables, cutting-plane techniques tailor-made for binary state variables and a projection from the binary to the original space. This way, we are able to obtain non-convex, piecewise linear cuts to approximate the non-convex value functions of multistage MILPs. Using some additional regularization, this is even possible if those value functions are not (Lipschitz) continuous. Together with piecewise linear relaxations, this yields non-convex underestimators for the non-convex value functions of MINLPs. All approximations are refined dynamically and, by careful design, it is ensured that all cuts remain valid even with such refinements.

The proposed method can be enhanced to solve stochastic MINLPs as well. In particular, a sampling-based approach like in SDDP could be used. In such case some adaptions have to be made with respect to the refinement criteria (forward solutions may remain unchanged for several iterations until the *right* scenarios are sampled) or the convergence checks, though.

While the presented version of NC-NBD already uses approximations which are dynamically refined, different strategies may be even more dynamic and efficient in practice. For instance, the piecewise linear relaxations could be refined dynamically in the inner loop as well.

The main drawback of NC-NBD is that the considered subproblems can become severely large, since for binary approximation, for piecewise linear approximations and for cut projection, a high number of additional variables and constraints may have to be introduced. This can become problematic, especially, if a very high binary expansion precision is required to approximate the value functions sufficiently good in the forward solutions. Recent results show that the number of binary variables K required grows linearly with the dimension n_t of state variables and logarithmically with the inverse of the binary precision β_t [55].

Therefore, in its current form, NC-NBD is best applicable to multistage MINLPs which are too large to solve in their extensive form, but for which each subproblem is sufficiently small and contains only a few nonlinear functions.

Acknowledgements Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)–445857709. The authors are grateful to the anonymous reviewers and the editor for their thoughtful and substantial remarks on earlier versions of this manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

A Proof of Lemma 3.8

Proof We start proving the second inequality. We have

$$\begin{aligned}
\underline{\widehat{Q}}_t^{R,i}(x^i_{\mathbb{B}t-1}, \mathfrak{Q}^{i+1}_{t+1}) &= \min_{z_t \in X_{t-1}} \underline{\widehat{Q}}_t^{i}(z_t, \mathfrak{Q}^{i+1}_{t+1}) + \sigma_t \|x^i_{\mathbb{B}t-1} - z_t\| \\
&= \min_{\mathfrak{z}_t \in [0,1]^{K_{t-1}}} \underline{\widehat{Q}}_t^{i}(B_{t-1}\mathfrak{z}_t, \mathfrak{Q}^{i+1}_{t+1}) + \sigma_t \|B_{t-1}(\lambda^i_{t-1} - \mathfrak{z}_t)\| \\
&\leq \min_{\mathfrak{z}_t \in [0,1]^{K_{t-1}}} \underline{\widehat{Q}}_t^{i}(B_{t-1}\mathfrak{z}_t, \mathfrak{Q}^{i+1}_{t+1}) + \sigma_t \|B_{t-1}\| \|\lambda^i_{t-1} - \mathfrak{z}_t\| \\
&= \min_{\mathfrak{z}_t \in [0,1]^{K_{t-1}}} \underline{\widehat{Q}}_{\mathbb{B}t}^{i}(\mathfrak{z}_t, \mathfrak{Q}^{i+1}_{t+1}) + \sigma_t \|B_{t-1}\| \|\lambda^i_{t-1} - \mathfrak{z}_t\| \\
&= \underline{\widehat{Q}}_{\mathbb{B}t}^{R,i}(\lambda^i_{t-1}, \mathfrak{Q}^{i+1}_{t+1}).
\end{aligned}$$

The inequality follows from $\|B_{t-1}\|$ being the induced matrix norm to the used vector norm. The last equality is obtained by choosing the same norm and $\alpha_t := \sigma_t \|B_{t-1}\|$ as regularization factor in $(\boldsymbol{\widehat{P}^{R,i}_{\mathbb{B}t}(\lambda^i_{t-1}, \mathfrak{Q}^{i+1}_{t+1})})$.

To show the first inequality, we construct a dual vector componentwise by

$$\widehat{\pi}_{tj} := \begin{cases} l_t, \text{ if } \lambda^i_{t-1,j} = 1 \\ -l_t, \text{ if } \lambda^i_{t-1,j} = 0. \end{cases}$$

This vector is feasible, as it satisfies $\|\widehat{\pi}_t\|_\infty \leq l_t$. By feasibility and by definition of the Lagrangian dual $(\boldsymbol{D^i_{\mathbb{B}t}(\lambda^i_{t-1}, \mathfrak{Q}^{i+1}_{t+1})})$ it follows

$$\phi_{\mathbb{B}t}(\lambda^i_{t-1}) \geq \min_{\mathfrak{z}_t \in [0,1]^{K_{t-1}}} \underline{\widehat{Q}}_{\mathbb{B}t}^{i}(\mathfrak{z}_t, \mathfrak{Q}^{i+1}_{t+1}) + \widehat{\pi}_t^\top (\lambda^i_{t-1} - \mathfrak{z}_t). \tag{14}$$

Moreover, by construction we have $\widehat{\pi}_{tj}(\lambda^i_{t-1,j} - \mathfrak{z}_{tj}) = l_t |\lambda^i_{t-1,j} - \mathfrak{z}_{tj}|$ in each component j. Inserting this into (14) and choosing $l_t \geq \alpha_t$, we obtain the first inequality.

B Proof of Lemma 3.10

Proof Consider line (4) in the KKT conditions. By rearranging and taking norms on both sides, we obtain

$$\|\nu_t - \mu_t\| = \|\pi_{t+1} + B_t^\top \eta_t\| \leq \|\pi_{t+1}\| + \|B_t^\top \eta_t\| \leq \|\pi_{t+1}\| + \|B_t^\top\| \|\eta_t\|. \tag{15}$$

The inequalities follow with the triangle inequality and with the compatibility of the matrix norm.

We can bound all three norms in (15) individually. In the Lagrangian dual, we have $\|\pi_{t+1}\| \leq \sigma_{t+1}\|B_t\|$. With Remark 3.9, we can bound $\|B_t^\top\|$ and with Assumption **(A4)**, we have $\|\eta_t\| \leq \rho_t$.

For example, using the ∞-norm, we obtain

$$\|\nu_t - \mu_t\| \leq \|\pi_{t+1}\|_\infty + \|B_t^\top\|_\infty \|\eta_t\|_\infty \leq \sigma_{t+1}\|B_t\|_1 + \|B_t\|_1 \rho_t \leq U_{t,\max}(\sigma_{t+1} + \rho_t).$$

By the equivalence of norms, we obtain similar bounds using other norms. This means that every entry of $\nu_t - \mu_t$ is bounded by this constant. Moreover, since in each component only ν_t or μ_t can be non-zero, this also implies that the components of ν_t and μ_t are bounded by this constant. □

C Proof of Lemma 4.1

Proof From the Lipschitz continuity of $\underline{Q}_t^{R,i}$ we have

$$\underline{Q}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^{i+1}) - \underline{Q}_t^{R,i}(\widehat{x}_{\mathbb{B},t-1}^i, \mathfrak{Q}_{t+1}^{i+1}) \leq L_t^R \|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\|. \tag{16}$$

Analogously, using Assumption **(A4)** and Lemma 3.11, for the cut approximation we obtain

$$\mathfrak{Q}_t^{i+1}(\widehat{x}_{\mathbb{B},t-1}^i) - \mathfrak{Q}_t^{i+1}(\widehat{x}_{t-1}^i) \leq \rho_t \|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\|. \tag{17}$$

Starting with (17) it follows

$$\begin{aligned}
\mathfrak{Q}_t^{i+1}(\widehat{x}_{t-1}^i) &\geq \mathfrak{Q}_t^{i+1}(\widehat{x}_{\mathbb{B},t-1}^i) - \rho_t \|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\| \\
&\geq \phi_{\mathbb{B}t}^{i+1}(\lambda_{t-1}^i) - \rho_t \|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\| \\
&\geq \widehat{\underline{Q}}_t^{R,i}(\widehat{x}_{\mathbb{B},t-1}^i, \mathfrak{Q}_{t+1}^{i+1}) - \rho_t \|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\| \\
&\geq \underline{Q}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^{i+1}) - (L_t^R + \rho_t)\|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\|.
\end{aligned}$$

The second inequality follows from the definition of $\mathfrak{Q}_t^{i+1}(\cdot)$. The third inequality follows from Lemma 3.8 and the last one is obtained using (16). □

D Proof of Lemma 4.2

Proof The structure of the proof is inspired by the proof for Lemma 4 in [1].

As the inner loop does not terminate and X is compact, there exists an infinite sequence of forward pass trial solutions $(\widehat{x}^i)_{i\in\mathbb{N}}$ with cluster points. Let $\widehat{x}^* \in X$ be such cluster point and $(\widehat{x}^{i_j})_{j\in\mathbb{N}}$ a subsequence of $(\widehat{x}^i)_{i\in\mathbb{N}}$ with $\lim_{j\to\infty} \widehat{x}^{i_j} = \widehat{x}^*$.

We show that $\lim_j \mathfrak{Q}_t^{i_j}(\widehat{x}_{t-1}^{i_j}) \geq \widehat{Q}_t^R(\widehat{x}_{t-1}^*)$ holds by backward induction.

For $t = T + 1$, this relation is trivially true, since no stage follows after T. Now, assume it already holds for stage $t + 1$, *i.e.*,

$$\lim_j \mathfrak{Q}_{t+1}^{i_j}(\widehat{x}_t^{i_j}) \geq \widehat{Q}_{t+1}^R(\widehat{x}_t^*).$$

We consider two subsequent indices in the subsequence $(\widehat{x}^{i_j})_{j\in\mathbb{N}}$.

$$\mathfrak{Q}_t^{i_j+1}(\widehat{x}_{t-1}^{i_j}) \geq \mathfrak{Q}_t^{i_{j-1}+1}(\widehat{x}_{t-1}^{i_j}) \geq \mathfrak{Q}_t^{i_{j-1}+1}(\widehat{x}_{t-1}^{i_{j-1}}) - \rho_t\|\widehat{x}_{t-1}^{i_j} - \widehat{x}_{t-1}^{i_{j-1}}\|,$$

where the first inequality follows from the monotonicity of $\mathfrak{Q}_t(\cdot)$ in i and the second inequality uses Lemma 3.11.

By adding zero, we obtain

$$\begin{aligned}\mathfrak{Q}_t^{i_j+1}(\widehat{x}_{t-1}^{i_j}) \geq\ & \underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) + \mathfrak{Q}_t^{i_{j-1}+1}(\widehat{x}_{t-1}^{i_{j-1}}) \\ & - \underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) - \rho_t\|\widehat{x}_{t-1}^{i_j} - \widehat{x}_{t-1}^{i_{j-1}}\|.\end{aligned}$$

We can now also use the monotonicity of $\underline{\widehat{Q}}_t^R(\cdot)$ in i and apply Lemma 4.1 to obtain

$$\begin{aligned}\mathfrak{Q}_t^{i_j+1}(\widehat{x}_{t-1}^{i_j}) \geq\ & \underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) + \mathfrak{Q}_t^{i_{j-1}+1}(\widehat{x}_{t-1}^{i_{j-1}}) \\ & - \underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}+1}) - \rho_t\|\widehat{x}_{t-1}^{i_j} - \widehat{x}_{t-1}^{i_{j-1}}\| \\ \geq\ & \underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) \\ & - (L_t^R + \rho_t)\|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\| - \rho_t\|\widehat{x}_{t-1}^{i_j} - \widehat{x}_{t-1}^{i_{j-1}}\|\end{aligned} \tag{18}$$

Moreover, we expand

$$\underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) = \widehat{f}_t(\widehat{x}_t^{i_{j-1}}, \widehat{y}_t^{i_{j-1}}) + \sigma_t\|\widehat{x}_{t-1}^{i_{j-1}} - \widehat{z}_t^{i_{j-1}}\| + \mathfrak{Q}_{t+1}^{i_{j-1}}(\widehat{x}_t^{i_{j-1}}) \tag{19}$$

and

$$\widehat{Q}_t^R(\widehat{x}_{t-1}^{i_{j-1}}) = \widehat{f}_t(\tilde{x}_t, \tilde{y}_t) + \sigma_t\|\widehat{x}_{t-1}^{i_{j-1}} - \tilde{z}_t\| + \widehat{Q}_{t+1}^R(\tilde{x}_t) \tag{20}$$

with corresponding optimal points $(\widehat{z}_t^{i_{j-1}}, \widehat{x}_t^{i_{j-1}}, \widehat{y}_t^{i_{j-1}})$ and $(\tilde{z}_t, \tilde{x}_t, \tilde{y}_t)$.

Then with (20) it follows

$$\widehat{Q}_t^R(\widehat{x}_{t-1}^{i_{j-1}}) \leq \widehat{f}_t(\widehat{x}_t^{i_{j-1}}, \widehat{y}_t^{i_{j-1}}) + \sigma_t\|\widehat{x}_{t-1}^{i_{j-1}} - \widehat{z}_t^{i_{j-1}}\| + \widehat{Q}_{t+1}^R(\widehat{x}_t^{i_{j-1}})$$

as the solution from (19) is feasible. Thus,

$$\widehat{Q}_t^R(\widehat{x}_{t-1}^{i_{j-1}}) - \underline{\widehat{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) \leq \widehat{Q}_{t+1}^R(\widehat{x}_t^{i_{j-1}}) - \mathfrak{Q}_{t+1}^{i_{j-1}}(\widehat{x}_t^{i_{j-1}}).$$

Rearranging yields

$$\widehat{\underline{Q}}_t^{R,i_{j-1}}(\widehat{x}_{t-1}^{i_{j-1}}, \mathfrak{Q}_{t+1}^{i_{j-1}}) \geq \widehat{Q}_t^R(\widehat{x}_{t-1}^{i_{j-1}}) + \mathfrak{Q}_{t+1}^{i_{j-1}}(\widehat{x}_t^{i_{j-1}}) - \widehat{Q}_{t+1}^R(\widehat{x}_t^{i_{j-1}}). \tag{21}$$

With inserting (21) in (18), we obtain

$$\begin{aligned}\mathfrak{Q}_t^{i_j+1}(\widehat{x}_{t-1}^{i_j}) \geq{} & \underbrace{\widehat{Q}_t^R(\widehat{x}_{t-1}^{i_{j-1}})}_{(*)} + \underbrace{\mathfrak{Q}_{t+1}^{i_{j-1}}(\widehat{x}_t^{i_{j-1}}) - \widehat{Q}_{t+1}^R(\widehat{x}_t^{i_{j-1}})}_{(\#)} \\ & - \underbrace{(L_t^R + \rho_t)\|\widehat{x}_{t-1}^i - \widehat{x}_{\mathbb{B},t-1}^i\|}_{(+)} - \underbrace{\rho_t\|\widehat{x}_{t-1}^{i_j} - \widehat{x}_{t-1}^{i_{j-1}}\|}_{(-)}\end{aligned}$$

We take limits on both sides. $(*)$ converges to $\widehat{Q}_t^R(\widehat{x}_{t-1}^*)$, since the function is continuous. $(\#)$ becomes greater than or equal to zero by the induction hypothesis. $(+)$ tends to zero with Lemma 3.6 since with j to ∞, the binary precision β_t goes to 0. $(-)$ tends to zero as $\widehat{x}_{t-1}^{i_j}$ and $\widehat{x}_{t-1}^{i_{j-1}}$ both converge to $\widehat{x}_{t-1}^*$.

Thus, the induction is proven for t. As this result holds for any cluster point of $(\widehat{x}^i)_{i\in\mathbb{N}}$, the assertion follows. □

E Proof of Theorem 4.3

Proof Consider the first stage optimal value $\widehat{v}^{FP,i}$ of the forward pass. By recursion we obtain

$$\widehat{v}^{FP,i} = \widehat{\overline{v}}^i + \sum_{t=2}^T \mathfrak{Q}_t(\widehat{x}_{t-1}^i) - \widehat{\underline{Q}}_t^{R,i}(\widehat{x}_{t-1}^i, \mathfrak{Q}_{t+1}^i)$$

and hence

$$\widehat{v}^{FP,i} \geq \widehat{\overline{v}}^i + \sum_{t=2}^T \mathfrak{Q}_t(\widehat{x}_{t-1}^i) - \widehat{Q}_t^R(\widehat{x}_{t-1}^i) \geq \widehat{v}^R + \sum_{t=2}^T \mathfrak{Q}_t(\widehat{x}_{t-1}^i) - \widehat{Q}_t^R(\widehat{x}_{t-1}^i). \tag{22}$$

As in the proof of Lemma 4.2, let $(\widehat{x}^{i_j})_{j\in\mathbb{N}}$ denote a convergent subsequence of $(\widehat{x}^i)_{i\in\mathbb{N}}$, with $\lim_j \widehat{x}^{i_j} = \widehat{x}^*$. Applying (22) to this subsequence and taking limits on both sides, yields

$$\begin{aligned}\lim_j \widehat{v}^{FP,i_j} &\geq \widehat{v}^R + \lim_j \left(\sum_{t=2}^T \mathfrak{Q}_t(\widehat{x}_{t-1}^{i_j}) - \widehat{Q}_t^R(\widehat{x}_{t-1}^{i_j})\right) \\ &\geq \widehat{v}^R + \sum_{t=2}^T \underbrace{\left(\widehat{Q}_t^R(\widehat{x}_{t-1}^*) - \widehat{Q}_t^R(\widehat{x}_{t-1}^*)\right)}_{=0}.\end{aligned}$$

The second inequality here stems from Equation (13). Using Lemma 3.4 and Assumption **(A3)**, yields $\lim_j \widehat{v}^{FP,i_j} \geq \widehat{v}$.

As $\widehat{v}^{FP,i_j}$ is also a lower bound to $\widehat{v}$, we have $\lim_j \widehat{v}^{FP,i_j} \leq \widehat{v}$. Thus, $\lim_j \widehat{v}^{FP,i_j} = \widehat{v}$. Since this is true for any cluster point $\widehat{x}^*$ of $(\widehat{x}^i)_{i\in\mathbb{N}}$, the inner loop converges to the optimal value $\widehat{v}$. With a similar reasoning it follows that every such cluster point is an optimal point of $(\widehat{P})$. □

F Proof of Theorem 4.7

Proof Let $\mathrm{x}^* := \big((z_t^*, x_t^*, y_t^*)\big)_{t=1,\dots,T}$ be an optimal point of $(\boldsymbol{P})$ and let $\big((\widehat{z}_t^\ell, \widehat{x}_t^\ell, \widehat{y}_t^\ell)\big)_{t=1,\dots,T}$ be an optimal point of its outer approximation $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$. Then, we have

$$v - \widehat{v}^\ell = \sum_{t=1}^{T} f_t(x_t^*, y_t^*) - \widehat{f}_t^\ell(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \tag{23}$$

As $(\widehat{\boldsymbol{P}}^{\boldsymbol{\ell}})$ is a relaxation of $(\boldsymbol{P})$ this expression is clearly non-negative for all ℓ. Moreover, analogously to the feasibility result in Lemma 4.5 for sufficiently large ℓ, for all t we have

$$0 \leq f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) - \widehat{f}_t^\ell(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \leq \varepsilon_{f_t}. \tag{24}$$

We distinguish two cases: First, let $\sum_{t=1}^T f_t(x_t^*, y_t^*) \leq \sum_{t=1}^T f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell)$, *e.g.*, because $(\widehat{z}_t^\ell, \widehat{x}_t^\ell, \widehat{y}_t^\ell)$ is feasible for $(\boldsymbol{P})$. Then, inserting this into (23) and using (24) it directly follows $v - \widehat{v}^\ell \leq \sum_{t=1}^T \varepsilon_{f_t}$.

Now let $\sum_{t=1}^T f_t(x_t^*, y_t^*) > \sum_{t=1}^T f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell)$. With Lemma 4.6, for any $\delta > 0$, there exists some $\widehat{\ell} \in \mathbb{N}$ such that for all $\ell \geq \widehat{\ell}$ there exists some feasible point $\widetilde{\mathrm{x}}^\ell := \big((\widetilde{z}_t^\ell, \widetilde{x}_t^\ell, \widetilde{y}_t^\ell)\big)_{t=1,\dots,T}$ of $(\boldsymbol{P})$ with $\|\widetilde{\mathrm{x}}^\ell - \widehat{\mathrm{x}}^\ell\|_2 \leq \delta$.

Clearly, $\sum_{t=1}^T f_t(x_t^*, y_t^*) \leq \sum_{t=1}^T f_t(\widetilde{x}_t^\ell, \widetilde{y}_t^\ell)$. Therefore,

$$0 \leq \sum_{t=1}^{T} f_t(x_t^*, y_t^*) - \sum_{t=1}^{T} f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \leq \sum_{t=1}^{T} \Big(f_t(\widetilde{x}_t^\ell, \widetilde{y}_t^\ell) - f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \Big). \tag{25}$$

With Assumption **(A1)** f_t is Lipschitz continuous with some constant $L_{f_t} > 0$. Thus, $\sum_{t=1}^T f_t$ is Lipschitz continuous with constant $L_f := \sum_{t=1}^T L_{f_t}$ and (25) can be bounded from above by $L_f \delta$.

We can write the right-hand side of (23) as

$$\sum_{t=1}^{T} \Big(f_t(x_t^*, y_t^*) - f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \Big) + \sum_{t=1}^{T} \Big(f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) - \widehat{f}_t^\ell(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \Big).$$

Then, with (24) and the previous result it follows that $v - \widehat{v}^\ell \leq L_f \delta + \sum_{t=1}^T \varepsilon_{f_t}$.

Choosing $\varepsilon := L_f \delta + \sum_{t=1}^{T} \varepsilon_{f_t}$ proves the assertion. □

G Proof of Theorem 4.8

Proof (a) From Theorem 4.7 it follows that if NC-NBD does not terminate for $\varepsilon = 0$, infinitely many piecewise linear relaxation refinements occur and $\widehat{v}^\ell$ converges to v. Using the premise, we have $\widehat{v}^\ell = \underline{\widehat{v}}^\ell$. Therefore, $\underline{\widehat{v}}^\ell$ converges to v. This also implies that the cut approximations $\mathfrak{Q}_{t+1}^\ell(\cdot)$ become tight at x_t^ℓ asymptotically. Thus, the solutions $\big((z_t^\ell, x_t^\ell, y_t^\ell)\big)_{t=1,\ldots,T}$ converge to an optimal solution for $(\boldsymbol{P})$.

(b) For sufficiently large ℓ, as in the proof of Theorem 4.7, we have

$$\sum_{t=1}^{T} f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \leq \sum_{t=1}^{T} \widehat{f}_t^\ell(\widehat{x}_t^\ell, \widehat{y}_t^\ell) + \sum_{t=1}^{T} \varepsilon_{f_t}.$$

Using this, and the termination of the inner loop, it follows

$$\sum_{t=1}^{T} f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \leq \overline{\widehat{v}}^\ell + \sum_{t=1}^{T} \varepsilon_{f_t} \leq \underline{\widehat{v}}^\ell + \widehat{\varepsilon} + \sum_{t=1}^{T} \varepsilon_{f_t} \leq v + \widehat{\varepsilon} + \sum_{t=1}^{T} \varepsilon_{f_t}.$$

For ℓ approaching infinity, $\widehat{x}^\ell$ becomes feasible. Thus,

$$v \leq \lim_\ell \overline{v}^\ell \leq \lim_\ell \sum_{t=1}^{T} f_t(\widehat{x}_t^\ell, \widehat{y}_t^\ell) \leq \lim_\ell v + \widehat{\varepsilon} + \sum_{t=1}^{T} \varepsilon_{f_t} = v + \widehat{\varepsilon}.$$

Since, $\overline{v}^\ell$ is bounded from above and non-increasing, the limit exists. This proves the assertion.

(c) This follows directly from b).

□

H Implementation details

The NC-NBD method is implemented in Julia-1.5.3 [7] using the `JuMP.jl` package [13] for optimization. The implementation is mainly derived from package `SDDP.jl` [12], which is enhanced by extensions specific to NC-NBD. To model piecewise linear approximations of multidimensional functions, we draw on `Delaunay.jl` [43] to determine triangulations. All MILP subproblems are solved with CPLEX and all MINLP subproblems are solved with appropriate MINLP solvers, both accessed using `GAMS.jl` [17]. The Lagrangian duals are solved using Kelley's cutting-plane method or a Level Bundle method as implemented in `SDDiP.jl` [24]. To reduce the size of the considered subproblems, a very basic Level cut selection technique is used based on `SDDP.jl`. In our case, however, not only the previously visited trial points, but also the anchor points are used to determine dominated cuts. Our code is available on GitHub [15].

All computations are performed on a machine with Intel(R) Xeon(R) E5-1630 v4 CPU and 128 GB RAM. All benchmark runs using state-of-the-art solvers are executed in GAMS 32.1.0.

I Unit commitment problem formulation and results

We consider a unit commitment problem with thermal generators based on [2] for some first tests of NC-NBD. To formulate this problem, we define the following elements:
Sets:

- $\mathcal{G}$: set of thermal generators

Data:

- p_g^f: price of fuel for generator g [EUR/MWh]
- $p_g^{\overline{s}}$: price of start up for generator g [EUR]
- $p_g^{\underline{s}}$: price of shut down for generator g [EUR]
- p_d: price for not meeting demand or load shedding [EUR/MWh]
- p_e: tax on emissions from generators [EUR/kg]
- d_t: demand at hour t [MWh]
- $\overline{c}_g$: maximum hourly generation of generator g [MWh]
- $\underline{c}_g$: minimum hourly generation of generator g [MWh]
- $\overline{r}_g$: ramp-up rate of generator g [MWh/h]
- $\underline{r}_g$: ramp-up rate of generator g [MWh/h]
- a_g, b_g, c_g: coefficients of the emission cost curve
- $v_g^a, v_g^b, v_g^c, v_g^d, v_g^e$: coefficients of the fuel cost curve

Decision Variables:

- x_{gt} : electricity production from generator g at time t [MWh]
- y_{gt} : binary variable modeling commitment of generator g at time t
- $\overline{y}_{gt}$: binary variable modeling start-up of generator g at time t
- $\underline{y}_{gt}$: binary variable modeling shut-down of generator g at time t
- $\underline{d}_t, \overline{d}_t$: variables modeling demand slack at time t

The objective is to minimize the total costs of electricity generation, which consists of different cost components. For all instances, the objective function is nonlinear due to a concave quadratic function modeling emission costs with $a_g < 0$ for all $g \in \mathcal{G}$ [11].

Additionally, we consider two different types of fuel cost function. In the first case (*base instances*), the fuel cost function is linear

$$c_{gt}^f(x_{gt}, y_{gt}) = p_g^f x_{gt}, \tag{26}$$

with a static fuel price p_g^f. In the second case (*valve-point instances*), we consider the more sophisticated cost function

$$c_{gt}^f(x_{gt}, y_{gt}) = v_g^a x_{gt}^2 + v_g^b x_{gt} + v_g^c y_{gt} + v_g^d \left| \sin(v_g^e(\underline{c}_g - x_{gt}))\right|, \tag{27}$$

including a convex quadratic term and a sinusoidal term, modeling the so-called valve point effect of steam turbines [34].

Then, the unit commitment model reads

$$(\boldsymbol{P}_{UC}) \quad \min \quad \sum_{g\in\mathcal{G}}\sum_{t=1}^{T} c_{gt}^f(x_{gt}, y_{gt}) + \sum_{g\in\mathcal{G}}\sum_{t=1}^{T} p_g^{\bar{s}} \bar{y}_{gt} + \sum_{g\in\mathcal{G}}\sum_{t=1}^{T} p_g^{\underline{s}} \underline{y}_{gt} \tag{28}$$

$$+ \sum_{g\in\mathcal{G}}\sum_{t=1}^{T} p_g^e \left(a_g x_{gt}^2 + b_g x_{gt} + c_g y_{gt}\right) + \sum_{t=1}^{T} p_d(\bar{d}_t + \underline{d}_t) \tag{29}$$

$$\text{s.t.} \quad \sum_{g\in\mathcal{G}} x_{gt} + \bar{d}_t - \underline{d}_t = d_t, \quad \forall t = 1, \dots, T \tag{30}$$

$$x_{gt} \le \bar{c}_g y_{gt}, \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{31}$$

$$x_{gt} \ge \underline{c}_g y_{gt}, \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{32}$$

$$x_{gt} - x_{g,t-1} \le \bar{r}_g y_{g,t-1} + \underline{c}_g(1 - y_{g,t-1}), \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{33}$$

$$x_{g,t-1} - x_{g,t} \le \underline{r}_g y_{gt} + \underline{c}_g(1 - y_{gt}), \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{34}$$

$$\bar{y}_{gt} \ge y_{gt} - y_{g,t-1}, \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{35}$$

$$\underline{y}_{gt} \ge y_{g,t-1} - y_{gt}, \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{36}$$

$$x_{gt} \ge 0, \quad \forall g \in \mathcal{G}, t = 1, \dots, T \tag{37}$$

$$\underline{d}_t, \bar{d}_t \ge 0, \quad \forall t = 1, \dots, T \tag{38}$$

$$y_{gt}, \bar{y}_{gt}, \underline{y}_{gt} \in \{0, 1\}, \quad \forall g \in \mathcal{G}, t = 1, \dots, T. \tag{39}$$

In this model, both, the continuous generation variables x_{gt} and the commitment variables y_{gt}, for all $g \in G, t = 1, \dots, T$, act as state variables. For the states at time $t = 0$, we use some fixed inputs. (30) denotes the balance between generation and demand. It also contains slack variables for unmet or overfulfilled demand, which are penalized in the objective. By this construction, Assumption **(A2)** is satisfied. (31)–(32) denote limits to the generator output, while (33)–(34) define ramping constraints. Those ramping constraints require x_{gt} to be a state variable. (33)–(34) are required to model start-up and shut-down costs.

For our input data, we draw on unit commitment instances created by Frangioni and published in the OR-Library [3]. The data is enhanced by own assumptions, as it does not cover all inputs in our problem formulation. We consider a number T of stages between 2 and 36 and a number G of generators between 3 and 10.

We solve all instances using NC-NBD with a relative optimality tolerance of 1% for the outer loop. All MILP subproblems are solved exactly while the outer loop MINLPs

Table 2 Bounds obtained for base instances in 100 EUR

T	G	BARON v	NC-NBD UB	NC-NBD LB	NC-NBD Time (s)
2	5	587.4	587.4	586.5	8
2	10	1266.1	1266.4	1262.3	93
3	5	849.8	849.8	849.3	17
3	10	1830.5	1830.5	1816.3	1502
4	5	1110.2	1110.2	1106.8	25
5	5	1374.7	1374.7	1372.6	202
10	3	2593.6	2598.3	2587.5	325
24	3	7037.5	7037.5	7037.5	674

are solved to an optimality tolerance of 10^{-3}. We use the lower bound provided by the MINLP solver to ensure that still a valid lower bound is obtained in the outer loop. The Lagrangian duals are solved with an optimality tolerance of 10^{-4}. In case that σ_t is not chosen large enough for some stage t from the beginning, it is increased iteratively within the solution procedure once identified. For the base instances, we use BARON [42,48] to solve the outer loop subproblem, for the valve-point instances we draw on LINDOGlobal [44], as BARON does not support sinusoidal functions. For the same reason, ANTIGONE [32], SCIP [16] and Gurobi [20] cannot be applied to the valve-point instances, so that we refer to LINDOGlobal and Couenne [5] as benchmarks.

The obtained upper bounds (UB) and lower bounds (LB) for the base instances are summarized in Table 2 and compared with the optimal point obtained by BARON.

All test instances can be solved by benchmark solvers in a few seconds, thus, outperforming NC-NBD. Still, these results can be regarded as a proof of concept for applying NC-NBD to multistage non-convex MINLPs, as in each case, the globally minimal point is succesfully approximated.

For the valve-point instances, we consider a larger number of stages, but only 3 or 4 generators, *i.e.*, 6 or 8 state variables, to focus on cases, in which NC-NBD looks most promising. For cases with many stages, we test differently scaled demand time series, as this seems to have a considerable effect on solution times. All instances are solved with a maximum solution time of two hours. The results are summarized in Table 3. If a solver does not terminate within the time limit, this is indicated by "-".

For a small number of stages, NC-NBD takes significantly more time than conventional solvers. With a larger number of stages, this difference vanishes, though. For 36 stages, NC-NBD manages to solve all considered instances within less than two hours, while LINDOGlobal and Couenne show more variance in computation time. For one instance, when terminated after two hours, they still show a 5% optimality gap.

Table 3 Results obtained for valve-point instances in 100 EUR

T	G	Demand		LINDOGlobal	Couenne	NC-NBD
10	3	v1	UB	3187.8	3178.0	3185.8
			LB	3166.7	3161.2	3158.6
			Time (s)	201	29	1074
16	3	v1	UB	5602.4	5584.1	5584.1
			LB	5553.0	5584.1	5548.3
			Time (s)	135	273	760
24	3	v1	UB	8593.4	8554.8	8554.8
			LB	8446.3	8545.0	8498.6
			Time (s)	–	4427	1408
24	3	v2	UB	9518.0	9508.8	9509.7
			LB	9428.5	9414.9	9432.4
			Time (s)	1562	647	771
24	4	v1	UB	9977.2	9965.1	9964.8
			LB	9848.3	9865.5	9867.5
			Time (s)	–	349	1220
24	4	v2	UB	8557.9	8587.1	8572.3
			LB	8473.1	8310.9	8492.2
			Time (s)	2603	–	4191
36	4	v1	UB	12,654.3	12,662.4	12,638.8
			LB	11,960.6	11,981.0	12,547.8
			Time (s)	–	–	6816
36	4	v2	UB	10,742.7	10,737.5	10,737.5
			LB	10,651.7	10,737.5	10,630.4
			Time (s)	2733	7501	3646
36	4	v3	UB	14,013.3	13,981.5	13,978.7
			LB	13,678.2	13,767.1	13,879.1
			Time (s)	–	–	3497

References

1. Ahmed, S., Cabral, F. G., Freitas Paulo da Costa, B.: Stochastic Lipschitz dynamic programming. Math. Programm. (2020)
2. Bacci, T., Frangioni, A., Gentile, C., Tavlaridis-Gyparakis, K.: New MINLP formulations for the unit commitment problems with ramping constraints. Preprint, http://www.optimization-online.org/DB_FILE/2019/10/7426.pdf (2019)
3. Beasley, J.E.: OR-library: distributing test problems by electronic mail. J. Oper. Res. Soc. **41**(11), 1069 (1990)
4. Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., Mahajan, A.: Mixed-integer nonlinear optimization. Acta Numer. **22**, 1–131 (2013)
5. Belotti, P., Lee, J., Liberti, L., Margot, F., Wächter, A.: Branching and bounds tightening techniques for non-convex minlp. Optim. Methods Softw. **24**, 597–634 (2009)
6. Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. Numer. Math. **4**(1), 238–252 (1962)

7. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: a fresh approach to numerical computing. SIAM Rev. **59**(1), 65–98 (2017)
8. Birge, J.R.: Decomposition and partitioning methods for multistage stochastic linear programs. Oper. Res. **33**(5), 989–1007 (1985)
9. Burlacu, R., Geißler, B., Schewe, L.: Solving mixed-integer nonlinear programmes using adaptively refined mixed-integer linear programmes. Optim. Methods Softw. **35**(1), 37–64 (2020)
10. Cerisola, S., Latorre, J.M., Ramos, A.: Stochastic dual dynamic programming applied to nonconvex hydrothermal models. Eur. J. Oper. Res. **218**(3), 687–697 (2012)
11. Van Dinter, J., Rebennack, S., Kallrath, J., Denholm, P., Newman, A.: The unit commitment model with concave emissions costs: a hybrid benders' decomposition with nonconvex master problems. Ann. Oper. Res. **210**(1), 361–386 (2013)
12. Dowson, O., Kapelevich, L.: SDDP.jl: a Julia package for stochastic dual dynamic programming. INFORMS J. Comput. (2020)
13. Dunning, I., Huchette, J., Lubin, M.: JuMP: a modeling language for mathematical optimization. SIAM Rev. **59**(2), 295–320 (2017)
14. Feizollahi, M.J., Ahmed, S., Sun, A.: Exact augmented Lagrangian duality for mixed integer linear programming. Math. Program. **161**(1–2), 365–387 (2017)
15. Füllner, C.: NCNBD.jl. Code released on GitHub https://github.com/ChrisFuelOR/NCNBD.jl (2021)
16. Gamrath, G., Anderson, D., Bestuzheva, K., Chen, W.-K., Eifler, L., Gasse, M., Gemander, P., Gleixner, A., Gottwald, L., Halbig, K., Hendel, G., Hojny, C., Koch, T., Le Bodic, P., Maher, S. J., Matter, F., Miltenberger, M., Mühmer, E., Müller, B., Pfetsch, M. E., Schlösser, F., Serrano, F., Shinano, Y., Tawfik, C., Vigerske, S., Wegscheider, F., Weninger, D., Witzig, J.: The SCIP optimization suite 7.0. Technical report, Optimization Online (2020)
17. GAMS Software GmbH. GAMS.jl. Code released on GitHub https://github.com/GAMS-dev/gams.jl (2020)
18. Geißler, B.: Towards globally optimal solutions for MINLPs by discretization techniques with applications in gas network optimization. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (2011)
19. Geoffrion, A.M.: Generalized Benders decomposition. J. Optim. Theory Appl. **10**(4), 237–260 (1972)
20. LLC Gurobi Optimization. Gurobi optimization reference manual (2021)
21. Glover, F.: Improved linear integer programming formulations of nonlinear integer problems. Manage. Sci. **22**(4), 455–460 (1975)
22. Hjelmeland, M.N., Zou, J., Helseth, A., Ahmed, S.: Nonconvex medium-term hydropower scheduling by stochastic dual dynamic integer programming. IEEE Trans. Sustain. Energy **10**(1) (2019)
23. Infanger, G., Morton, D.P.: Cut sharing for multistage stochastic linear programs with interstage dependency. Math. Program. **75**(2), 241–256 (1996)
24. Kapelevich, L.: SDDiP.jl. Code released on GitHub https://github.com/lkapelevich/SDDiP.jl (2018)
25. Kallrath, J., Rebennack, S.: Computing area-tight piecewise linear overestimators, underestimators and tubes for univariate functions. In: Optimization in Science and Engineering, pp. 273–292. Springer (2014)
26. Kannan, R.: Algorithms, analysis and software for the global optimization of two-stage stochastic programs. PhD thesis, Massachusetts Institute of Technology (2018)
27. Kilinç, M.R., Sahinidis, N.V.: Exploiting integrality in the global optimization of mixed-integer nonlinear programming problems with baron. Optim. Methods Softw. **33**(3), 540–562 (2018)
28. Li, C., Grossmann, I.E.: A generalized Benders decomposition-based branch and cut algorithm for two-stage stochastic programs with nonconvex constraints and mixed-binary first and second stage variables. J. Global Optim. **75**(2), 247–272 (2019)
29. Li, X., Chen, Y., Barton, P.I.: Nonconvex generalized Benders decomposition with piecewise convex relaxations for global optimization of integrated process design and operation problems. Ind. Eng. Chem. Res. **51**(21), 7287–7299 (2012)
30. Li, X., Tomasgard, A., Barton, P.I.: Nonconvex generalized Benders decomposition for stochastic separable mixed-integer nonlinear programs. J. Optim. Theory Appl. **151**, 425–454 (2011)
31. Meyer, R.R.: Integer and mixed-integer programming models: general properties. J. Optim. Theory Appl. **3**(4) (1975)
32. Misener, R., Floudas, C.A.: Antigone: algorithms for continuous/integer global optimization of nonlinear equations. J. Global Optim. **59**(2–3), 503–526 (2014)

33. Ogbe, E., Li, X.: A joint decomposition method for global optimization of multiscenario nonconvex mixed-integer nonlinear programs. J. Global Optim. **75**, 595–629 (2018)
34. Pedroso, J. P., Kubo, M., Viana, A.: Unit commitment with valve-point loading effect. Technical report, Universidade do Porto (2014)
35. Pereira, M.V.F., Pinto, L.M.V.G.: Multi-stage stochastic optimization applied to energy planning. Math. Program. **52**(1–3), 359–375 (1991)
36. Philpott, A.B., Wahid, F., Bonnans, F.: MIDAS: a mixed integer dynamic approximation scheme. Math. Program. (2019)
37. Rebennack, S.: Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming. Math. Program. **156**(1–2), 343–389 (2016)
38. Rebennack, S.: Computing tight bounds via piecewise linear functions through the example of circle cutting problems. Math. Methods Oper. Res. **84**(1), 3–57 (2016)
39. Rebennack, S., Kallrath, J.: Continuous piecewise linear delta-approximations for univariate functions: computing minimal breakpoint systems. J. Optim. Theory Appl. **167**(2), 617–643 (2015)
40. Rebennack, S., Krasko, V.: Piecewise linear function fitting via mixed-integer linear programming. INFORMS J. Comput. (2020)
41. Rockafellar, R.T., Wets, R.J.B.: Variational Analysis. Springer, Berlin (2009)
42. Sahinidis, N.V.: BARON 17.8.9: global optimization of mixed-integer nonlinear programs, User's Manual (2017)
43. Schnetter, E.: Delaunay.jl. Code released on GitHub https://github.com/eschnett/Delaunay.jl (2020)
44. Schrage, L.: *LindoSystems: LindoAPI* (2004)
45. Steeger, G., Lohmann, T., Rebennack, S.: Strategic bidding for a price-maker hydroelectric producer: stochastic dual dynamic programming and Lagrangian relaxation. IISE Trans. **47**, 1–14 (2018)
46. Steeger, G., Rebennack, S.: Dynamic convexification within nested Benders decomposition using Lagrangian relaxation: an application to the strategic bidding problem. Eur. J. Oper. Res. **257**(2), 669–686 (2017)
47. Tawarmalani, M., Sahinidis, N.: Convex extensions and envelopes of lower semi-continuous functions. Math. Program. **93**, 247–263 (2002)
48. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. **103**, 225–249 (2005)
49. van Slyke, R.M., Wets, R.: L-shaped linear programs with applications to optimal control and stochastic programming. J. SIAM Appl. Math. **17**(4), 638–663 (1969)
50. Vielma, J.P., Ahmed, S., Nemhauser, G.: Mixed-integer models for nonseparable piecewise-linear optimization: unifying framework and extensions. Oper. Res. **58**(2), 303–315 (2010)
51. Vielma, J.P., Nemhauser, G.L.: Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. Math. Program. **128**(1–2), 49–72 (2011)
52. Zhang, S., Sun, X. A.: Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization. (2021). Available at https://arxiv.org/abs/1912.13278. Accessed 29 Nov 2021
53. Zhou, K., Kilinç, M.R., Chen, X., Sahinidis, N.V.: An efficient strategy for the activation of mip relaxations in a multicore global minlp solver. J. Global Optim. **70**(3), 497–516 (2018)
54. Zou, J., Ahmed, S., Sun, X.: Multistage stochastic unit commitment using stochastic dual dynamic integer programming. IEEE Trans. Power Syst. **34**(3) (2019)
55. Zou, J., Ahmed, S., Sun, X.A.: Stochastic dual dynamic integer programming. Math. Program. **175**(1–2), 461–502 (2019)

B.1 Errata for the published paper

- Page 994, Algorithm 1: Lines 9 and 10 have to be swapped, so that the upper bound is computed outside of the for-loop.
- Page 1019, line 16 should be replaced with "p_g^e: tax on emissions from generator g [EUR/kg]".
- Page 1019, line 21: $\underline{r}_g$ should be named the "ramp-down rate".

Paper C

On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming

Christian Füllner[a], X. Andy Sun[b], Steffen Rebennack[a]

[a] *Karlsruhe Institute of Technology (KIT), Institute for Operations Research, Stochastic Optimization, Karlsruhe, Germany*

[b] *Sloan School of Management, Massachusetts Institute of Technology, USA*

working paper, suggested citation:
Füllner C., Sun X. A., Rebennack S. (2024), On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming.

On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming

Christian Füllner[1*], X. Andy Sun[2], and Steffen Rebennack[1]

[1]Institute for Operations Research (IOR), Stochastic Optimization (SOP), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[*]Address correspondence to: christian.fuellner@kit.edu
[2]Sloan School of Management, Massachusetts Institute of Technology, USA

Abstract

We provide new theoretical insight on the generation of linear and non-convex cuts for value functions of multistage stochastic mixed-integer programs based on Lagrangian duality. First, we analyze in detail the impact that the introduction of copy constraints, and especially, the choice of the accompanying constraint set for the copy variable have on the properties of the Lagrangian dual and the obtained cuts. We show that the well-known tightness result for Lagrangian cuts in stochastic dual dynamic programming (SDDiP) crucially depends on this choice, and not on the introduction of copy constraints in itself. Afterwards, we generalize our results to the case where a Lipschitz regularization is applied to the value functions. In particular, we show a deep relation between norm-bounded Lagrangian dual problems and the closed convex envelope of the regularized value functions. For linear Lagrangian cuts, using an appropriate regularization, this result can be used to enhance the tightness result from SDDiP to the regularized case. For the generation of non-convex cuts, we pick up on the lift-and-project idea proposed by Füllner and Rebennack in their non-convex nested Benders decomposition (NC-NBD) method. We generalize this cut generation idea to the stochastic case. We then show that by careful choice of the norm used for regularization in the lifted space, Lipschitz continuity of the obtained non-convex cuts can be guaranteed. By that, we resolve an open theoretical question from the original NC-NBD paper. We highlight all our results by simple illustrative examples. Our work allows for a profound understanding of how and to which effect copy constraints and regularization may be used in decomposition methods in stochastic mixed-integer programming.

1 Introduction

1.1 Motivation and prior work

In many practical applications, sequential decisions have to be made over a finite number of stages, while some of the problem data of the following stages are subject to uncertainty. Such decision-making processes can be modeled as multistage stochastic programs. Often it is assumed that the number of scenarios describing the uncertain data is finite. In this case, such stochastic problems can be reformulated as large-scale deterministic problems. However, for a practically relevant number of scenarios, these problems get too large to be solved by off-the-shelf solvers. Therefore, they are usually approached by decomposition methods.

For multistage stochastic linear programs (MS-LP), these decomposition methods have a long tradition and are well-studied. Among the most prominent ones are nested Benders decomposition (NBD) [5] and stochastic dual dynamic programming (SDDP) [20]. One of their key ideas is to decompose the original multistage problem by stage and scenario into subproblems, which are linked by state variables and (expected) value functions. These functions are piecewise-linear and convex, and thus can be iteratively approximated by linear cutting-planes. Finitely many such cuts are sufficient to ensure (almost sure) convergence.

However, in many applications, some of the decisions have to be integer or binary, which yields a multistage stochastic mixed-integer linear program (MS-MILP). Problems of this class are very hard to solve, as they combine the challenges of dynamic and stochastic programming with the non-convexity of mixed-integer programs. In particular, the value functions become non-convex and discontinuous, which aggravates their approximation.

Various strategies have been proposed to solve MS-MILPs. A natural approach is to relax the integer constraints to obtain an MS-LP that can be solved by existing methods. However, in that case not the original MS-MILP is solved, which may hamper the practical benefit of this approach. The same issue occurs if the expected value functions are statically or dynamically convexified [8, 25, 24]. Another approach is to approximate the original value functions with linear Benders cuts [3, 28] or Lagrangian cuts [31] without convexifying the problem. In general, these cuts only yield a non-exact convex approximation of the expected value functions. Therefore, in such cases, convergence of decomposition methods is not guaranteed. As a relief, in two-stage stochastic programming linear cuts are often incorporated into branch-and-bound approaches, where convergence is guaranteed by additional branching [7, 9]. However, for multistage problems this is computationally intractable.

Still, two strategies have been proposed recently on how linear cuts can be used to solve stochastic MILPs to arbitrary precision. The first one is to use *scaled cuts* which are guaranteed to recover the convex envelope of the expected value functions [27]. This approach has only been applied to the two-stage case so far. The second one is to use stochastic dual dynamic integer programming (SDDiP) in a lifted space [31]. The SDDiP method uses special Lagrangian cuts to approximate the expected value functions of MS-MILPs. These cuts are tight, and thus ensuring convergence if all state variables are binary (or bounded integer). For general MS-MILPs it is therefore proposed to approximate the state variables using a static binary expansion [31]. Then, in the lifted binary state space, linear cuts can be used to approximate the expected value functions. In this case, however, not the original MS-MILP, but an approximation is solved. While it is possible to derive theoretical results on the approximation quality [31], it is not immediately clear how the static binary expansion should be chosen in practice.

The special Lagrangian cuts in SDDiP rely on the introduction of copy constraints, adding local copies of the state variables, which are then dualized in the Lagrangian relaxation. Even though this is barely discussed in [31], these copy constraints are accompanied by additional constraints for the new variables. Importantly, different choices of these constraints may lead to distinct cuts with different approximation quality. In this paper, we study this aspect in more theoretical detail and by that provide a new, generalized perspective on the generation of Lagrangian cuts.

Only recently, more focus has been put on deriving non-convex approximations (also called non-convex *cuts*) for the non-convex and discontinuous value functions. In [21], step functions are used instead of cutting-planes to approximate them, presuming their monotonicity. In stochastic Lipschitz dynamic programming (SLDP) [1], Lipschitz cuts are proposed under the assumption of Lipschitz continuity of the value functions, as well as knowledge of a Lipschitz constant. Moreover, non-convex cuts can be generated by solving augmented Lagrangian dual problems [1] instead of classical Lagrangian dual problems. However, the approach in [1] requires a strong recourse assumption, namely the complete continuous recourse.

In [29], the authors propose a new class of SDDP-type algorithms for solving multistage stochastic mixed-integer nonlinear programs with *non-Lipschitzian* value functions. In particular, the paper proposes a new cut generation framework using *generalized conjugacy* with *regularization*, which is guaranteed to obtain a global optimum without the assumption of complete recourse. This significantly generalizes SDDP, SDDiP, and SLDP. A complete oracle complexity analysis is also achieved in the paper. In [30], SDDP-type algorithms are extended to multistage distributionally robust convex optimization and a new type of SDDP algorithm that adaptively chooses the forward or backward direction at each node is proposed with complete oracle complexity analysis.

An alternative approach to obtain non-convex cuts is to use the binary approximation idea from [31] in a dynamic and temporary fashion, paired with *Lipschitz regularization*. This is one of the key ingredients of the non-convex nested Benders decomposition (NC-NBD) method proposed in [13], where MILP relaxations are solved iteratively in order to solve an MINLP. In the backward pass of this iteration, the state variables are temporarily lifted to a binary space, where tight Lagrangian cuts are generated as in SDDiP. These linear cuts are then projected back to the original state space, which yields a tight non-convex approximation of the value functions. Under some strong technical assumption, it is shown that these approximations are Lipschitz continuous. In order to improve the approximation quality, the binary approximation precision is iteratively refined if required. We generalize these results to the stochastic case in this paper and explore the regularization in more detail, which allows us to drop the technical assumption taken in [13].

Whereas regularization is particularly helpful for deriving non-convex, but Lipschitz continuous approximations of non-Lipschitzian value functions, it may also be useful when generating linear Lagrangian cuts. It naturally ensures feasibility of the subproblems, so that there is no need for an additional recourse assumption. Also, as shown in [12, 29], under mild conditions, regularization of an MS-MILP ensures exact solution of the original MS-MILP. Despite these amenities, there exists no study yet focusing on the effects and the theoretical backbone of using Lipschitz regularization in MS-MILPs in order to derive linear Lagrangian cuts. We close this scientific gap in this work.

1.2 Contribution

In this paper, we investigate in detail the effects of copy constraints and Lipschitz regularization on the generation and properties of linear and non-convex cuts for the value functions of MS-MILPs, which are constructed based on Lagrangian duality. Our results allow for a more profound understanding of how copy constraints and Lipschitz regularization may be utilized in multistage stochastic programming.

As we provide new theoretical insight on Lagrangian cuts and duality, our work can be considered as complementary to the work on Lagrangian cuts by Zou et al. [31], on conjugacy cuts and regularization by Zhang and Sun [30, 29], on augmented Lagrangian duality by Feizollahi et al. [12] and on Lagrangian-based non-convex cuts by Füllner and Rebennack [13].

With respect to **linear Lagrangian cuts** we make the following key contributions.

1. In Sect. 3, we thoroughly explore the role of copy constraints when deriving Lagrangian cuts. More precisely, we show that accompanying the new copy variables with different types of constraints leads to cuts with different approximation characteristics. This way, we provide a new theoretical perspective on Lagrangian relaxation and cuts in general. In particular, we show that the well-known tightness result for Lagrangian cuts in SDDiP [31] actually relies on the accompanying constraints more than on the introduction of copy constraints itself.

2. In Sect. 4, we consider the case of Lipschitz regularization and generalize our previous results to this case. We prove a deep relation between norm-bounded Lagrangian dual problems and primal convexifications of the regularized subproblems in MS-MILPs, in the sense that they yield the same optimal value if dual norms are used in both cases. While such relation is known for non-regularized problems and unbounded Lagrangian duals, we are not aware of any literature covering this result for the regularized case.

3. We use this result to show that the obtained Lagrangian cuts are tight for the closed convex envelope of the regularized value functions. This also clarifies which kind of cuts are constructed if (artificial) multiplier bounds are introduced in Lagrangian dual problems in practice. Furthermore, for the 1-norm penalty function, tightness for the true regularized value functions can be achieved as long as all state variables are binary. This generalizes the tightness result from SDDiP [31] to the regularized case.

With respect to **non-convex cuts** we make the following key contributions.

4. We significantly extend the idea from the NC-NBD method by Füllner and Rebennack [13] to compute linear Lagrangian cuts in a lifted binary state space and to project them back to the original state space to obtain non-convex approximations of the value functions. First, we generalize this cut generation idea from the deterministic to the stochastic case. Additionally, we show that by using appropriate weighted norms in the Lipschitz regularization and in the Lagrangian dual, Lipschitz continuity of the obtained non-convex cuts is ensured. This is crucial to guarantee convergence of NC-NBD. In doing that, we show that the technical Assumption (A4) in [13] can be dropped, and thus close an open theoretical question from [13].

We underline all our results by providing illustrative examples.

1.3 Structure

This paper is structured as follows. In Sect. 2 we start with formulating the MS-MILP and stating some basic concepts and assumptions. In Sect. 3 we discuss classical Lagrangian duality and cuts, but with special focus on the impact of the chosen copy constraint approach. In Sect. 4 we first introduce Lipschitz regularization in a formal way. Then, we present our main result on norm-bounded Lagrangian duality and the associated Lagrangian cuts. In Sect. 5 we recapitulate the NC-NBD approach by Füllner and Rebennack [13] with special focus on the applied regularization. We enhance the cut generation idea to the stochastic case and show how Lipschitz continuity of the non-convex cuts can be ensured. We finish with a conclusion in Sect. 6. For reasons of clarity some technical proofs are shifted to the appendix.

2 Problem formulation

We consider MS-MILPs with a finite number $T \in \mathbb{N}$ of stages, where some of the problem data is uncertain and evolves according to a known stochastic process $\xi := (\xi_1, \dots, \xi_T)$ with deterministic ξ_1. We assume that the random data vectors $\xi_t, t = 1, \dots, T$, are discrete and finite, and thus the uncertainty can be modeled by a finite scenario tree.

Let $\mathcal{T} = (\mathcal{N}, \mathcal{E})$ denote a finite scenario tree with a set of nodes $\mathcal{N}$ and a set of edges $\mathcal{E}$. For each node $n \in \mathcal{N}$, the unique ancestor node is denoted by $a(n)$ and the set of child nodes is denoted by $\mathcal{C}(n)$. The probability for some node n to realize is $p_n > 0$ and assumed to be known. The transition probabilities between adjacent nodes $n, m \in \mathcal{N}$ (*i.e.*, edges $(n,m) \in \mathcal{E}$) can then be determined as $p_{nm} := \frac{p_m}{p_n}$. For the root node r, we assume $a(r) = \emptyset$ and $p_r = 1$. We define $\overline{\mathcal{N}} := \mathcal{N} \setminus \{r\}$ to address the set of nodes without the root node and $\widetilde{\mathcal{N}}$ to address the set of nodes without the leaf nodes.

2.1 Dynamic programming equations

The MS-MILP can be expressed recursively by its dynamic programming equations (for details see Zou et al. [31]). For the root node, we obtain

$$\begin{aligned} v^* := \min_{x_r, y_r} \quad & f_r(x_r, y_r) + \mathcal{Q}_{\mathcal{C}(r)}(x_r) \\ \text{s.t.} \quad & (x_r, y_r) \in \mathcal{F}_r(x_{a(r)}) \end{aligned} \tag{1}$$

with $x_{a(r)} = 0$, and v^* is the optimal value of the MS-MILP. Let $\overline{\mathbb{R}} := \mathbb{R} \cup \{+\infty\}$. For all $n \in \widetilde{\mathcal{N}}$, the *expected value function* $\mathcal{Q}_{\mathcal{C}(n)}(\cdot) : \mathbb{R}^{d_{a(n)}} \to \overline{\mathbb{R}}$ is defined by

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) := \sum_{m \in \mathcal{C}(n)} p_{nm} Q_m(x_n), \tag{2}$$

with the *value function* $Q_n(\cdot) : \mathbb{R}^{d_{a(n)}} \to \overline{\mathbb{R}}$ defined by

$$\begin{aligned} Q_n(x_{a(n)}) := \min_{x_n, y_n} \quad & f_n(x_n, y_n) + \mathcal{Q}_{\mathcal{C}(n)}(x_n) \\ \text{s.t.} \quad & (x_n, y_n) \in \mathcal{F}_n(x_{a(n)}) \end{aligned} \tag{3}$$

for all $n \in \overline{\mathcal{N}}$. For the leaf nodes $n \in \mathcal{N} \setminus \widetilde{\mathcal{N}}$, we set $\mathcal{Q}_{\mathcal{C}(n)}(x_n) \equiv 0$. Moreover, we set $Q_n(x_{a(n)}) = +\infty$ if $\mathcal{F}_n(x_{a(n)}) = \emptyset$, and denote by

$$\text{dom}(Q_n) := \left\{ x_{a(n)} \in \mathbb{R}^{d_{a(n)}} \ : \ Q_n(x_{a(n)}) < +\infty \right\}$$

the *effective domain* of $Q_n(\cdot)$. The same applies to $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$.

For each node $n \in \mathcal{N}$, we distinguish the state variables $x_n \in \mathbb{R}^{d_n}$, which enter the child nodes' subproblems, and the local variables $y_n \in \mathbb{R}^{\tilde{d}_n}$. Moreover, $x_{a(n)}$ is the state variable from the ancestor node of n and is a fixed parameter in (3) of node n. Furthermore, $f_n(\cdot)$ denotes the objective function and $\mathcal{F}_n(x_{a(n)})$ denotes the feasible set which depends on the state $x_{a(n)}$. As we consider an MS-MILP, we assume that $f_n(x_n, y_n)$ is a linear function in x_n and y_n, and that $\mathcal{F}_n(x_{a(n)})$ is a mixed-integer polyhedral set for all $x_{a(n)}$. More precisely, we assume it to be defined by

$$\mathcal{F}_n(x_{a(n)}) := \left\{ (x_n, y_n) \in \mathbb{R}^{d_n} \times \mathbb{R}^{\tilde{d}_n} \ : \begin{array}{l} x_n \in X_n, \ y_n \in Y_n, \\ A_n x_{a(n)} + B_n x_n + C_n y_n \geq b_n \end{array} \right\}. \tag{4}$$

Here, A_n, B_n, C_n, b_n denote appropriately defined data matrices and vectors. The sets X_n and Y_n comprise constraints only associated with x_n or y_n, *e.g.*, box constraints or non-negativity constraints. More precisely, we assume that both sets are intersections of polyhedral sets $\bar{X}_n, \bar{Y}_n$ and possible integrality constraints. In the following, we also refer to X_n as the *state space*.

Remark 2.1. *We make two comments on the definition of $Q_n(\cdot)$ in* (3). *First, we should emphasize that regarding $Q_n(\cdot)$ as a function on $\mathbb{R}^{d_{a(n)}}$ is not necessarily standard in stochastic programming. Often it is (implicitly) assumed to be defined only on the domain $X_{a(n)}$. However, from our view, allowing $Q_n(\cdot)$ to be defined on $\mathbb{R}^{d_{a(n)}}$ with extended real values proves beneficial when we discuss the impact of copy constraints later on. Second, as the co-domain of $Q_n(\cdot)$ is $\overline{\mathbb{R}}$, $Q_n(\cdot)$ should be more rigorously defined as the infimum of the objective values in problem* (3). *However, below we take assumptions under which the minimization problem is bounded and finite infima are always attained. Therefore, we stick to the* min *operator in* (3) *with the additional definition of $Q_n(x_{a(n)}) = +\infty$ given that $\mathcal{F}_n(x_{a(n)}) = \emptyset$. This approach is also chosen for all other value functions throughout this paper.*

For the remainder of this article, we make some basic assumptions.

Assumption 1. *The following conditions are satisfied by* (1)-(4)*:*

(A1) For all $n \in \mathcal{N}$, the sets X_n and Y_n are compact.

(A2) For all $n \in \mathcal{N}$, all coefficients in A_n, B_n, C_n, b_n, f_n, $\bar{X}_n$ and $\bar{Y}_n$ are rational.

(A3) The MS-MILP has a feasible solution for each scenario, i.e., there exists some $(x_n, y_n)_{n \in \mathcal{N}}$ such that $(x_n, y_n) \in \mathcal{F}_n(x_{a(n)})$ for all $n \in \mathcal{N}$.

Note that the boundedness in (A1) immediately implies that $F_n(x_{a(n)})$ is bounded for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$ and $n \in \mathcal{N}$. By (A1) and [18, Theorem 2.1], it follows that the subproblems (1) and (3) are either infeasible or attain a finite infimum.

Furthermore, we obtain the following well-known properties for the value functions. For completeness, we provide a proof in Appendix A.

Lemma 2.2. *Under Assumption 1, for all $n \in \overline{\mathcal{N}}$, the value functions $Q_n(\cdot)$, and for all $n \in \mathcal{N}$, the expected value functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ are proper, lsc (lower semicontinuous), and piecewise polyhedral with finitely many pieces. Moreover,* $\mathrm{dom}(Q_n)$ *is closed.*

By applying the properness reasoning to the root node, we conclude that v^* is finite.

2.2 Closed convex envelopes

The main challenge in decomposition methods for MS-MILPs is that the (expected) value functions are not guaranteed to be continuous or convex. Therefore, approximations of the value functions based on linear cutting-planes may at best yield their closed convex envelopes. To deal with this concept, we denote by $\operatorname{conv}(S)$ the convex hull of some set $S \subseteq \mathbb{R}^d$. For a function $f : S \to \mathbb{R}$, its *closed convex envelope* $\overline{\operatorname{co}}(f) : \operatorname{conv}(S) \to \mathbb{R}$ (also called *convex closure*) is defined as the pointwise supremum of all affine functions majorized by f on S [4].

In our setting, for all $n \in \bar{N}$ the value functions $Q_n(\cdot)$ are defined on $\mathbb{R}^{d_{a(n)}}$. Hence, $\overline{\operatorname{co}}(Q_n)(\cdot)$ is the pointwise supremum of all affine functions defined on $\mathbb{R}^{d_{a(n)}}$ and majorized by $Q_n(\cdot)$ on $\mathbb{R}^{d_{a(n)}}$. With $Q_n(x_{a(n)}) = +\infty$ for all $x_{a(n)} \notin \operatorname{dom}(Q_n)$, the crucial part is $\overline{\operatorname{co}}(Q_n)(x_{a(n)}) \leq Q_n(x_{a(n)})$ for all $x_{a(n)} \in \operatorname{dom}(Q_n)$.

It is well-known that the closed convex envelope $\overline{\operatorname{co}}(Q_n)(\cdot)$ is equivalent to the biconjugate $(Q_n)^{**}(\cdot)$ of $Q_n(\cdot)$ in this setting. For a formal definition of biconjugate functions and a general introduction to the conjugacy theory we refer to [4].

Lemma 2.3. *Under Assumption 1, for all $n \in \bar{N}$ and all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$*

$$\overline{\operatorname{co}}(Q_n)(x_{a(n)}) = (Q_n)^{**}(x_{a(n)}).$$

We provide a proof in Appendix B.

Remark 2.4. *The pointwise supremum of all convex (not necessarily affine) functions defined on $\mathbb{R}^{d_{a(n)}}$ and majorized by $Q_n(\cdot)$ on $\mathbb{R}^{d_{a(n)}}$ is the convex envelope $\operatorname{co}(Q_n)(\cdot)$. If $\operatorname{dom}(Q_n)$ is compact, then from the lower semicontinuity of $Q_n(\cdot)$ (Lemma 2.2) and Lemma 2.3 it follows that*

$$\overline{\operatorname{co}}(Q_n)(x_{a(n)}) = \operatorname{co}(Q_n)(x_{a(n)})$$

for all $x_{a(n)} \in \operatorname{conv}(\operatorname{dom}(Q_n))$, see [11, Theorem 2.2].

3 Classical Lagrangian cuts and the role of copy constraints

We revisit some central results on Lagrangian duality and its usage in decomposition methods to generate cuts. In doing that, we focus specifically on the role that copy constraints and constraints accompanying them have on the obtained results. This yields a generalization of some known results from the literature.

3.1 Introducing copy constraints

We follow the SDDiP approach [31] and introduce local copies z_n together with copy constraints $x_{a(n)} = z_n$ to all subproblems (3). Crucially, in addition, we also impose the accompanying constraints $z_n \in Z_{a(n)}$ on z_n to restrict their potential values, with $Z_{a(n)} \subseteq \mathbb{R}^{d_{a(n)}}$. This yields the subproblems

$$\begin{aligned} Q_n(x_{a(n)}) = \min_{x_n, y_n, z_n} \quad & f_n(x_n, y_n) + \mathcal{Q}_{\mathcal{C}(n)}(x_n) \\ \text{s.t.} \quad & (z_n, x_n, y_n) \in \mathcal{F}_n \\ & z_n = x_{a(n)} \\ & z_n \in Z_{a(n)}. \end{aligned} \tag{5}$$

Here, $\mathcal{F}_n := \Big\{(x_n, y_n, z_n) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R}^{d_n} \times \mathbb{R}^{\tilde{d}_n} \ : \ (x_n, y_n) \in \mathcal{F}_n(z_n)\Big\}$.

For any $x_{a(n)} \in Z_{a(n)}$, subproblem (5) is equivalent to subproblem (3) due to the copy constraint. However, for any $x_{a(n)} \notin Z_{a(n)}$, the copy constraint accompanied with

$z_{a(n)} \in Z_{a(n)}$ immediately induces infeasibility, and thus $Q_n(x_{a(n)}) = +\infty$. This means that the additional constraints on $z_{a(n)}$ provide a natural way to restrict the effective domain of $Q_n(\cdot)$ such that $\mathrm{dom}(Q_n) \subseteq Z_{a(n)}$. On the other hand, we should choose $Z_{a(n)} \supseteq X_{a(n)}$ in order to not exclude feasible points.

As long as $Z_{a(n)}$ contains all feasible values of $x_{a(n)}$, the actual choice of $Z_{a(n)}$ may seem of minor importance at first glance. However, it turns out that it has an important effect on the considered closed convex envelope $\overline{\mathrm{co}}(Q_n)(\cdot)$ (recall that $\overline{\mathrm{co}}(Q_n)(\cdot)$ underestimates $Q_n(\cdot)$ on $\mathrm{dom}(Q_n)$), and by that on the quality of the obtained Lagrangian cuts. As we shall see, choosing $Z_{a(n)}$ appropriately is also the main secret behind the tightness results for SDDiP [31].

We discuss different choices for $Z_{a(n)}$:

- $\boldsymbol{Z_{a(n)} = X_{a(n)}}$. This is the most intuitive choice, as it yields $\mathrm{dom}(Q_n) \subseteq X_{a(n)}$, *i.e.*, we restrict $Q_n(\cdot)$ to the actual state space. This choice also yields the best possible polyhedral underestimators of $Q_n(\cdot)$ on $X_{a(n)}$. It has been considered in [9, 22, 31] for instance.

- $\boldsymbol{Z_{a(n)} = \mathrm{conv}(X_{a(n)})}$. This choice may yield worse approximations of $Q_n(\cdot)$ on $X_{a(n)}$, as it leads to valid under-approximators on the larger set $\mathrm{conv}(X_{a(n)})$. However, this property may also be exploited on purpose, as it is done in the NC-NBD method [13] that we discuss in detail in Sect. 5. This choice is also considered in the original SDDiP work [31], but without further explanation.

- $\boldsymbol{Z_{a(n)} = \bar{X}_{a(n)}}$. In this case, $Z_{a(n)}$ is the LP relaxation of $X_{a(n)}$, which has the advantage that no additional integer variables have to be considered in the reformulated subproblem (5).

- $\boldsymbol{Z_{a(n)} = \mathbb{R}^{d_{a(n)}}}$. This choice leads to the same Lagrangian cuts as if no copy constraints are introduced at all, but instead the original coupling constraints $A_n x_{a(n)} + B_n x_n + C_n y_n \geq b_n$ are dualized in the Lagrangian relaxation.

We take the following assumption on $Z_{a(n)}$, which is satisfied in most practical applications.

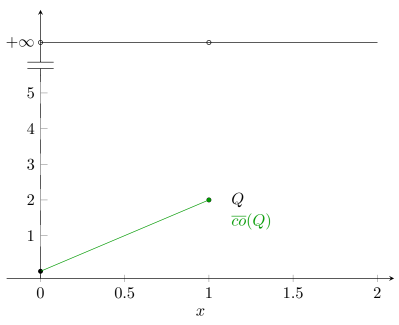
Assumption 2. *The set $Z_{a(n)}$ is closed and either satisfies $Z_{a(n)} = \mathbb{R}^{d_{a(n)}}$ or is rational MILP-representable.*

In the remainder of this paper, we use two recurring examples to illustrate our results. We start with the first one to highlight the differences in the convex envelopes for different choices of $Z_{a(n)}$.

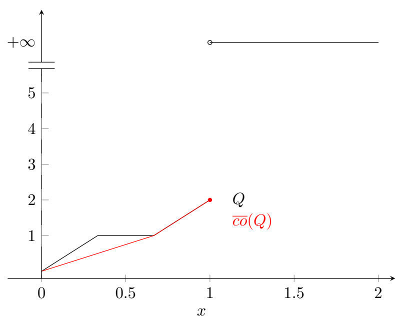
Example 3.1. *Consider the value function*

$$Q(x) = \min\Big\{ y_1 + y_2 \ : \ 2y_1 + y_2 \geq 3x, \ 0 \leq y_1 \leq 2, \ 0 \leq y_2 \leq 3, \ y_1 \in \mathbb{Z} \Big\} \tag{6}$$

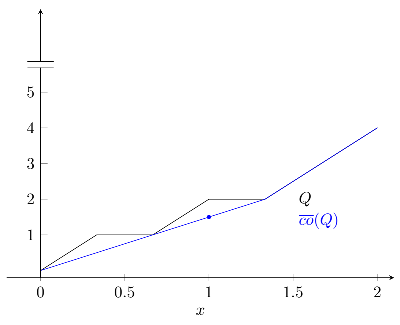
with state space $X = \{0, 1\}$ [31, Example 2]. We introduce the local variable z, the copy constraint $z = x$, and constraint $z \in Z$. Depending on the choice of Z, the effective domain of $Q(\cdot)$ defined in (5) *changes. If we set $Z = X$, then* $\mathrm{dom}(Q) = \{0, 1\}$. *If we set $Z = \mathrm{conv}(X)$, then* $\mathrm{dom}(Q) = [0, 1]$. *And if we set $Z = \mathbb{R}$ (or do not introduce copy constraints at all), then* $\mathrm{dom}(Q) = [0, 2]$. *Since* $\overline{\mathrm{co}}(Q)(\cdot)$ *underestimates $Q(\cdot)$ on* $\mathrm{dom}(Q)$, *the approximation quality on the actual state space X may vary. This is illustrated in Fig. 1, where the approximation at $x = 1$ is highlighted by dots. Clearly,* $\overline{\mathrm{co}}(Q)(\cdot)$ *is tight for $Q(\cdot)$ at $x = 1$ in the first two cases, but not in the third one.*



(a) $Z = \{0, 1\}$.



(b) $Z = [0, 1]$.



(c) $Z = \mathbb{R}$.

Figure 1: $Q(\cdot)$ on $[0, 2]$ and $\overline{\text{co}}(Q)(\cdot)$ on $\text{dom}(Q)$ for different choices of Z in Example 3.1.

Remark 3.2. *As shown in Example 3.1, essentially, different choices of $Z_{a(n)}$ yield distinct functions $Q_n(\cdot)$ and $\overline{\text{co}}(Q_n)(\cdot)$. For the remainder of this paper, this dependence should be kept in mind, even if we do not state it explicitly for clarity of notation.*

For the second example, the value function is not only non-convex, but also discontinuous.

Example 3.3. *Consider the value function*

$$\begin{aligned} Q(x) = \quad \min_{y,z} \quad & y_1 - \frac{3}{4}y_2 + \frac{3}{4}y_3 + \frac{9}{4}y_4 \\ s.t. \quad & \frac{5}{4}y_1 - y_2 + \frac{1}{2}y_3 + \frac{1}{3}y_4 = z \\ & y_1, y_2, y_3, y_4 \geq 0 \\ & y_1, y_2 \in \mathbb{Z} \\ & z = x \\ & z \in [0, 2] \end{aligned} \tag{7}$$

with continuous state space $X = [0, 2]$, where we already introduced copy constraints with $Z = X$. The value function and its closed convex envelope are illustrated in Fig. 2.

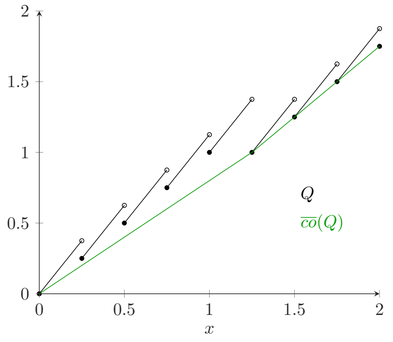


Figure 2: $Q(\cdot)$ and $\overline{\text{co}}(Q)(\cdot)$ for Example 3.3.

3.2 The approximate subproblem

In Benders-like decomposition methods for multistage problems such as SDDP or SD-DiP, the functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ are iteratively approximated by cutting-planes. For this reason, in the recursion (5) $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ are replaced by polyhedral outer approximations $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$, which are then iteratively updated over the iterations i. Due to the non-convex character of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, for MS-MILPs these approximations will not be tight in general.

In each iteration i, in a forward pass, the tree $\mathcal{T}$ is traversed in forward direction. For each node $n \in \mathcal{N}$ (or some sampled subset) and some given set $Z_{a(n)}$, the subproblems (5) are solved with $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ being replaced by $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$. This yields incumbents $x^i_{a(n)} \in X_{a(n)}$ for each considered node n, which are handed as parameters to the child nodes $\mathcal{C}(n)$. These incumbents are also used in a backward pass, where the tree $\mathcal{T}$ is traversed in backward direction, and where the approximations $\mathfrak{Q}^i_{\mathcal{C}(n)}(\cdot)$ are updated to $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ by constructing additional cuts. For the remainder of this paper, we solely focus on this cut generation step. For more details on the complete algorithmic procedure, we refer to [14, 31].

Whereas Assumption 1 guarantees the existence of a feasible solution for MS-MILP, for some $x_{a(n)}$, the subproblems may become infeasible. In such a case, in addition to the previously mentioned *optimality cuts* also *feasibility cuts* are required, which iteratively approximate $\text{dom}(Q_n)$. Usually, this requirement is avoided by taking an appropriate recourse assumption, such as:

Assumption 3 (Relatively complete recourse)**.** *For all $n \in \mathcal{N}$, for all $x_{a(n)}$ feasible at node $a(n)$, there exist (z_n, x_n, y_n) satisfying the constraints in subproblem* (5).

In the backward pass of iteration i, in each considered node $n \in \overline{\mathcal{N}}$, subproblems for the incumbent $x^i_{a(n)}$ and the updated outer approximation $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ are solved. By a partial epigraph reformulation, we shift $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ to the constraints. Then, the subproblems can be expressed as

$$\begin{aligned} \underline{Q}_n^{i+1}(x^i_{a(n)}) := \min_{x_n, y_n, z_n, \theta_{\mathcal{C}(n)}} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}_n^{i+1} \\ & z_n = x^i_{a(n)}, \end{aligned} \tag{8}$$

where we define

$$\mathcal{M}_n^{i+1} := \Big\{(x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \ : \ z_n \in Z_{a(n)}, \ (x_n, y_n, z_n) \in \mathcal{F}_n, \ \theta_{\mathcal{C}(n)} \geq \mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(x_n)\Big\}. \quad (9)$$

The polyhedral outer approximation $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ is defined as the pointwise maximum of all linear cuts generated so far. To avoid unboundedness of subproblems (8), we initialize each $\mathfrak{Q}_{\mathcal{C}(n)}^{0}(\cdot)$ with a valid lower bound $\underline{\theta}_{\mathcal{C}(n)} > -\infty$. We refer to $\underline{Q}_n^{i+1}(\cdot)$ as the *approximate value function.*

In the same vein as Assumption 2, we impose another requirement.

Assumption 4. *For all $n \in \overline{\mathcal{N}}$ and all iterations i, all linear cuts defining the polyhedral set $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(x_n)$ are defined by rational coefficients.*

Note that in practice where cut coefficients are computed numerically, this is always satisfied.

Our assumptions on rationality of coefficients and MILP-representability yield the following important result, which goes back to [18] and which we need later on.

Lemma 3.4 (Theorem 11.13 in [10])**.** *Under Assumptions 1, 2, 4, the set* $\operatorname{conv}(\mathcal{M}_n^{i+1})$ *is a closed rational polyhedron, and the recession cones of* $\operatorname{conv}(\mathcal{M}_n^{i+1})$ *and* $\widehat{\mathcal{M}}_n^{i+1}$ *coincide, where the latter set denotes the continuous relaxation of $\mathcal{M}_n^{i+1}$.*

Importantly, due to the cut constraints, however, neither $\mathcal{M}_n^{i+1}$ nor $\operatorname{conv}(\mathcal{M}_n^{i+1})$ has to be bounded.

Similarly to Lemma 2.2, we obtain the following properties for the functions $\underline{Q}_n^{i+1}(\cdot)$ by additionally exploiting Assumptions 2 and 4 and that $\theta_{\mathcal{C}(n)}$ is bounded from below:

Lemma 3.5. *Let $n \in \overline{\mathcal{N}}$. If Assumptions 1, 2, 4 are satisfied, then $\underline{Q}_n^{i+1}(\cdot)$ is proper, lsc, and piecewise polyhedral with finitely many pieces. Moreover,* $\operatorname{dom}(\underline{Q}_n^{i+1}) = \operatorname{dom}(Q_n)$*, and thus closed.*

3.3 The Lagrangian dual

In order to derive linear Lagrangian cuts to approximate the non-convex value functions $Q_n(\cdot)$, we consider a *Lagrangian relaxation* in which the copy constraints in subproblem (8) are relaxed. For a given vector of dual multipliers $\pi_n \in \mathbb{R}^{d_{a(n)}}$ for the copy constraints, this yields the problem

$$\begin{aligned} \mathcal{L}_n^{i+1}(\pi_n) := \min_{x_n, y_n, z_n, \theta_{\mathcal{C}_n}} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} - \pi_n^\top z_n \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}_n^{i+1}, \end{aligned} \quad (10)$$

where we omit the constant $\pi_n^\top x_{a(n)}^i$ in the objective. For varying π_n, this relaxation defines the *dual function* $\mathcal{L}_n^{i+1}(\cdot)$. The problem of optimizing the dual function over the dual multipliers π_n is the *Lagrangian dual problem*

$$\underline{Q}_n^{D,i+1}(x_{a(n)}^i) := \max_{\pi_n} \quad \mathcal{L}_n^{i+1}(\pi_n) + \pi_n^\top x_{a(n)}^i. \quad (11)$$

As (10) is a relaxation of the primal subproblem (8), it yields a lower bound for $\underline{Q}_n^{i+1}(\cdot)$ at $x_{a(n)}^i$. Solving the dual problem (11) can be interpreted as finding the tightest Lagrangian relaxation for (8), and thus the tightest such lower bound.

Remark 3.6. *In the light of Remark 2.1, note that unless $Z_{a(n)}$ is bounded, the dual function $\mathcal{L}_n^{i+1}(\pi_n)$ may yield the trivial lower bound $-\infty$ for some π_n. However, based on Assumption 1 and $\theta_{\mathcal{C}(n)}$ being bounded from below, it is finite for $\pi_n = 0$. Therefore, $\underline{Q}_n^{D,i+1}(\cdot)$ is proper. Moreover, we shall see that $\underline{Q}_n^{D,i+1}(x_{a(n)})$ is guaranteed to be finite-valued for all $x_{a(n)} \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$.*

A well-known result on Lagrangian relaxation for MILPs is that under some assumptions the optimal value of the dual (11) is the same as that of the following convexification of the primal subproblem (8)

$$\begin{aligned} \underline{Q}_n^{C,i+1}(x_{a(n)}^i) := \min_{x_n, y_n, z_n, \theta_{\mathcal{C}_n}} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \operatorname{conv}(\mathcal{M}_n^{i+1}) \\ & z_n = x_{a(n)}^i. \end{aligned} \tag{12}$$

Here, the part of the constraints which is not relaxed in (10) is convexified, while the copy constraints keep their original form. We first derive an auxiliary result.

Lemma 3.7. *Under Assumptions 1, 2, 4, the function $\underline{Q}_n^{C,i+1}(\cdot)$ is proper, lsc and convex with* $\operatorname{dom}(\underline{Q}_n^{C,i+1}) = \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$. *Moreover, on its effective domain it is piecewise linear.*

We provide a proof for this result in Appendix C.

Based on this lemma, the equivalence between the primal convexification (12) and the dual (11) is given below.

Theorem 3.8 (Theorem 1 in [15])**.** *Under Assumptions 1, 2, 4, the Lagrangian dual* (11) *and the primal convexified problem* (12) *satisfy*

$$\underline{Q}_n^{D,i+1}(x_{a(n)}^i) = \underline{Q}_n^{C,i+1}(x_{a(n)}^i)$$

for all $x_{a(n)}^i \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$.

The main idea behind this result is that problems (11) and (12) are LP duals of each other. Note that we even have $\underline{Q}_n^{D,i+1}(x_{a(n)}^i) = \underline{Q}_n^{C,i+1}(x_{a(n)}^i)$ for all $x_{a(n)}^i \in \mathbb{R}^{d_{a(n)}}$, since both functions are bounded from below and may only take the value $+\infty$ if non-finite.

This result motivates another interesting property of the Lagrangian dual, which relates to the closed convex envelope of the approximate value function. This result is widely known, but we give a self-contained proof in Appendix D.

Theorem 3.9. *Under Assumption 1, the Lagrangian dual* (11) *satisfies*

$$\underline{Q}_n^{D,i+1}(x_{a(n)}^i) = \overline{\operatorname{co}}(\underline{Q}_n^{i+1})(x_{a(n)}^i)$$

for all $x_{a(n)}^i \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$.

3.4 Lagrangian cuts

We now focus on the generation of Lagrangian cuts at points $x_{a(n)}^i \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$ using problem (11), as introduced in [31]. We first define these cuts formally.

Definition 3.10 (Lagrangian cut)**.** *For all $n \in \overline{\mathcal{N}}$, a Lagrangian cut is given by*

$$\theta_n \geq \mathcal{L}_n^{i+1}(\pi_n^i) + (\pi_n^i)^\top x_{a(n)},$$

where π_n^i denotes optimal dual multipliers in (11) *for node n and some given $x_{a(n)}^i \in$* $\operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$.

Under relatively complete recourse, *i.e.*, Assumption 3, within the decomposition method it is ensured that for all iterations i and all nodes $n \in \mathcal{N}$, the condition $x_{a(n)}^i \in \operatorname{dom}(\underline{Q}_n^{i+1}) \subseteq \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$ is satisfied, so there never occurs an $x_{a(n)}^i$ for which no cut can be computed due to infeasibility.

In general, the Lagrangian cuts have the following important properties [see 31]:

Theorem 3.11. *Under Assumptions 1, the Lagrangian cuts defined in 3.10 are*

(a) valid lower approximations of $Q_n(\cdot)$ for all $x_{a(n)} \in Z_{a(n)}$,

(b) tight for $\overline{\operatorname{co}}(\underline{Q}_n^{i+1})(\cdot)$ at $x_{a(n)}^i$,

(c) finite, i.e., only finitely many different cuts can be generated, if the dual multipliers π_n^i are dual basic solutions.

Proof. Properties (a) and (c) are proven in [31, Theorem 3]. Tightness of Lagrangian cuts is directly proven for $\underline{Q}_n^{i+1}(\cdot)$ in [31]. Property (b) is about $\overline{\operatorname{co}}(\underline{Q}_n^{i+1})(\cdot)$ instead. But it directly follows from Theorem 3.9 and Definition 3.10. □

Property (a) implies that for $Z_{a(n)} = \operatorname{conv}(X_{a(n)})$ Lagrangian cuts underestimate $Q_n(\cdot)$ not only on $X_{a(n)}$, but also on the larger set $\operatorname{conv}(X_{a(n)})$ (see the related Remark 3.2). Note that property (a) even holds if the Lagrangian dual (11) is not solved to optimality, *i.e.*, if suboptimal dual multipliers are used in Definition 3.10.

Since we want to approximate $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ instead of approximating each $Q_m(\cdot), m \in \mathcal{C}(n)$, separately, we construct an aggregated cut from the cuts defined in 3.10. Using these aggregated cuts, we express $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ by

$$\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(x_n) := \min\Big\{\theta_{\mathcal{C}(n)} \in \mathbb{R} \,:\, \theta_{\mathcal{C}(n)} \geq \sum_{m \in \mathcal{C}(n)} p_{nm}\big(\mathcal{L}_m^{i+1}(\pi_m^r) + (\pi_m^r)^\top x_n\big)\ \forall r = 1,\dots,i+1\Big\}.$$

Using Theorem 3.11, the validity of $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ follows immediately.

Corollary 3.12. *Under Assumption 1, $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ is a valid lower approximation of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ for all $x_n \in Z_n$.*

Importantly, we cannot directly generalize the tightness result from Theorem 3.11 to $\overline{\operatorname{co}}(\mathbb{E}[Q_n])$, since in general $\mathbb{E}[\overline{\operatorname{co}}(Q_n)] \neq \overline{\operatorname{co}}(\mathbb{E}[Q_n])$ [27].

3.5 The case of tight Lagrangian cuts

In SDDiP [31], it is assumed that all state variables are binary, *i.e.*, $X_{a(n)} = \{0,1\}^{d_{a(n)}}$, and thus $\operatorname{conv}(X_{a(n)}) = [0,1]^{d_{a(n)}}$. This assumption has two key effects, which ensure the almost sure finite convergence of SDDiP to an optimal policy of the considered MS-MILP. First, $X_{a(n)}$ is finite. Second, the Lagrangian cuts from Definition 3.10 are not only tight for $\overline{\operatorname{co}}(\underline{Q}_n^{i+1})(\cdot)$ at $x_{a(n)}^i$, but in fact for $\underline{Q}_n^{i+1}(\cdot)$. This tightness is directly proven in [31, Theorem 3]. However, our previous analyses allow for a different perspective on this result, which is briefly mentioned in [31], but not used in the proof: It holds because $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$ and $\underline{Q}_n^{i+1}(\cdot)$ coincide at $x_{a(n)}^i$. The main reason for this is that $X_{a(n)}$ is contained in the extreme points of $Z_{a(n)}$. Therefore, this tightness result crucially depends on the choice of $Z_{a(n)}$. This perspective also allows to extend the SDDiP tightness result to more general cases, as also touched upon in Remark 1 in [29].

Theorem 3.13. *Under Assumptions 1, 2, 4, for any iteration i and any node $n \in \overline{\mathcal{N}}$, let $Z_{a(n)}$ be bounded and let $X_{a(n)}$ be contained in the set of extreme points of $Z_{a(n)}$. Then, for all $x_{a(n)} \in X_{a(n)} \cap \operatorname{dom}(\underline{Q}_n^{i+1})$, we have*

$$\overline{\text{co}}(\underline{Q}_n^{i+1})(x_{a(n)}) = \underline{Q}_n^{i+1}(x_{a(n)}).$$

We present a proof in Appendix E.

Combined with the properties of the Lagrangian cuts from Theorem 3.11, Theorem 3.13 directly implies a tightness result for the Lagrangian cuts.

Corollary 3.14 (Theorem 3 in [31])**.** *Under Assumptions 1, 2, 4, for any iteration i and any node $n \in \overline{\mathcal{N}}$, let $Z_{a(n)}$ be bounded and let $X_{a(n)}$ be contained in the set of extreme points of $Z_{a(n)}$. Then the Lagrangian cuts defined in Definition 3.10 are* tight *for $\underline{Q}_n^{i+1}(\cdot)$ at $x_{a(n)}^i$.*

Remark 3.15. *Assume that $X_{a(n)} = \{0,1\}^{d_{a(n)}}$ and that $Z_{a(n)} = X_{a(n)}$ or $Z_{a(n)} = \operatorname{conv}(X_{a(n)})$ as in SDDiP [31] and that we have relatively complete recourse (Assumption 3). Then, the conditions of Theorem 3.13 are satisfied, and for all feasible $x_{a(n)}$ the known tightness result of SDDiP follows.*

We highlight this result using the illustrative problems (6) and (7).

Example 3.16. *Consider the problem* (6) *with $Z = X = \{0,1\}$. Solving the dual* (11) *yields the cut $\theta \geq 2x$. This cut underestimates $Q(\cdot)$ on $\{0,1\}$ and is tight for $\overline{\text{co}}(Q)(\cdot)$ at $x = 1$, see Fig. 3a. We observe that it is even tight for $Q(\cdot)$ at this point.*

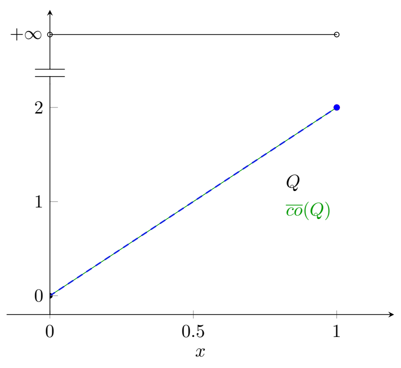
If we choose $Z = \operatorname{conv}(X) = [0,1]$ instead, we obtain the cut $\theta \geq -1 + 3x$ by solving (11)*. This cut underestimates $Q(\cdot)$ on $[0,1]$ and is tight for $\overline{\text{co}}(Q)(\cdot)$ at $x = 1$, see Fig. 3b. Again, we observe tightness for $Q(\cdot)$.*

In contrast, as illustrated by Fig. 3c by the gap between the red square and the blue dot, Theorem 3.13 is not guaranteed to hold if we choose $Z = \mathbb{R}$ or, equivalently, do not introduce copy constraint in the subproblems.

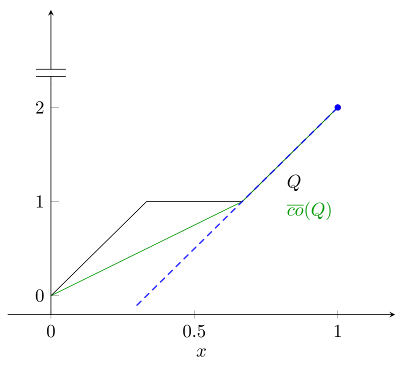
Example 3.17. *Consider the problem* (7) *and the incumbent $x = \frac{6}{5}$. Recall that $X = Z = [0,2]$, so the extreme point condition in Corollary 3.14 is not satisfied. Solving the Lagrangian dual* (11) *yields the cut $\theta \geq \frac{4}{5}x$. As Fig. 4 shows, this cut is tight for $\overline{\text{co}}(Q)(\cdot)$ at $x = \frac{6}{5}$ (blue dot), but not for $Q(\cdot)$ (red square).*

4 Lipschitz regularization and Lagrangian duality

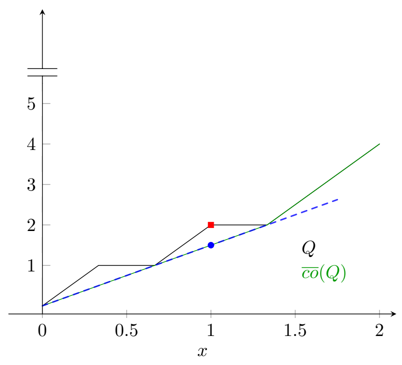
In this section, we address the generation of *linear* Lagrangian cuts for $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ when a Lipschitz regularization of the original MS-MILP is considered. We have shown in the previous section that such cuts can be generated even when no regularization is applied. In fact, regularization is particularly relevant for generating *non-convex* approximations of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, which we consider in Sect. 5. However, it may still be applied in cases where



(a) $Z = \{0, 1\}$.



(b) $Z = [0, 1]$.



(c) $Z = \mathbb{R}$.

Figure 3: Lagrangian cuts for $Q(\cdot)$ and different choices of Z in Example 3.16.

linear cuts are generated, for instance, to ensure feasibility of the subproblems. Moreover, the results for this case prove relevant to derive our results in Sect. 5. Therefore, we address the case of linear cuts first.

4.1 Applying a Lipschitz regularization

First, we formally introduce the considered Lipschitz regularization.

Definition 4.1 (Regularization)**.** *For any $n \in \overline{\mathcal{N}}$, let $\sigma_n > 0$ and fix some norm $\|\cdot\|$. Then we call*

$$\begin{aligned} Q_n^R(x_{a(n)}; \sigma_n \|\cdot\|) := \min_{x_n, y_n, z_n} \quad & f_n(x_n, y_n) + \sigma_n \|x_{a(n)} - z_n\| + \mathcal{Q}_{\mathcal{C}(n)}^R(x_n; \sigma_{\mathcal{C}(n)} \|\cdot\|) \\ s.t. \quad & (z_n, x_n, y_n) \in \mathcal{F}_n \\ & z_n \in Z_{a(n)} \end{aligned} \tag{13}$$

the regularized subproblem *or the* regularized value function *for node n, respectively. The* regularized expected value function *is defined by*

$$\mathcal{Q}_{\mathcal{C}(n)}^R(x_n; \sigma_{\mathcal{C}(n)} \|\cdot\|) := \sum_{m \in \mathcal{C}(n)} p_{nm} Q_m^R(x_n; \sigma_m \|\cdot\|).$$

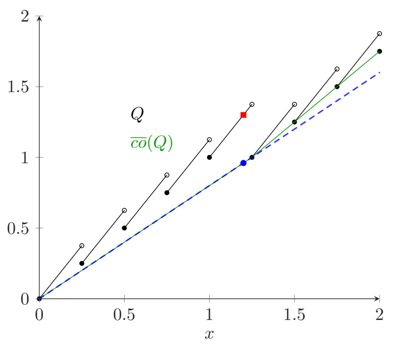


Figure 4: Lagrangian cut for $Q(\cdot)$ in Example 3.17.

By writing $\sigma_{\mathcal{C}(n)}$, we indicate that $\mathcal{Q}^R_{\mathcal{C}(n)}(\cdot;\sigma_{\mathcal{C}(n)}\|\cdot\|)$ depends on σ_m for all $m \in \mathcal{C}(n)$. For the root node, where no regularization is required, we obtain

$$\begin{aligned} v^R := \min_{x_r,y_r} \quad & f_r(x_r,y_r) + \mathcal{Q}^R_{\mathcal{C}(r)}(x_r;\sigma_r\|\cdot\|) \\ s.t. \quad & (x_r,y_r) \in \mathcal{F}_r(x_{a(r)}). \end{aligned} \tag{14}$$

Remark 4.2. *The regularized problem defined by the recursion* (13)-(14) *can be interpreted as applying a special* inf*-convolution $f_n\square(\sigma_n\|\cdot\|)$, called* Lipschitz regularization *or* Pasch-Hausdorff envelope *[2], to the objective function of the original MS-MILP, see also [30, 29].*

This regularization comes with two main advantages. First, it naturally ensures feasibility of the considered subproblems, even if we take no recourse assumption for the original subproblems.

Lemma 4.3. *Under Assumption 1, problem* (13) *is feasible for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$*, i.e., $\mathrm{dom}(Q^R_n;\sigma_n\|\cdot\|) = \mathbb{R}^{d_{a(n)}}$.

In particular, we do not require the recourse assumption (Assumption 3) from Sect. 3.

Second, using a Lipschitz regularization ensures that the considered value functions are σ_n-Lipschitz continuous instead of only lsc. While related results have been shown before, see [29], our assumptions differ a bit, so we provide a self-contained proof in Appendix F.

Lemma 4.4. *Under Assumptions 1, 2, for all $n \in \overline{\mathcal{N}}$, the regularized value function $Q^R_n(\cdot;\sigma_n\|\cdot\|)$ underestimates $Q_n(\cdot)$ and is proper and σ_n-Lipschitz continuous on $\mathbb{R}^{d_{a(n)}}$.*

Considering regularized MS-MILPs comes at the price that we do not necessarily solve the original MS-MILP any longer. In general, $v^R \leq v^*$ [29, Proposition 2]. However, equality can be imposed by using a sufficiently large σ_n for all nodes $n \in \overline{\mathcal{N}}$, as shown in [12, Theorem 5]:

Lemma 4.5. *There exist finite $\bar{\sigma}_n > 0$ for all $n \in \overline{\mathcal{N}}$ such that given $\sigma_n \geq \bar{\sigma}_n$, for all $n \in \mathcal{N}$, the penalty reformulation in* (13) *is exact, i.e., any optimal solution $(x_n, y_n, z_n)_{n\in\mathcal{N}}$ of the regularized MS-MILP* (13)-(14) *satisfies $z_n = x_{a(n)}$ for all $n \in \mathcal{N}$.*

Hence, for sufficiently large, but finite $\sigma_n > 0$, we have $v^R = v^*$. This result also implies that for any optimal solution $(x_n^*, y_n^*)_{n\in\mathcal{N}}$ of the original MS-MILP (1)-(3) we have $Q_n^R(x_{a(n)}^*; \sigma_n\|\cdot\|) = Q_n(x_{a(n)}^*)$ [29, Lemma 1].

Example 4.6 provides an illustration of regularized value functions and their properties.

Example 4.6. *Consider the problem* (7)*. We use the absolute value* $|\cdot|$ *as penalty function in* (13)*. The regularized value functions* $Q^R(\cdot;\sigma|\cdot|)$ *are depicted in Fig. 5 for different values of* σ*. It is visible that all of them underestimate* $Q(\cdot)$ *for all* $x \in [0,2]$*, that all of them are Lipschitz continuous and that they are monotonically increasing in* $\sigma > 0$*. Assume that the optimal first-stage solution is* $x^* = 1$*. Then an exact penalization is achieved for any* $\sigma \geq 1$*.*

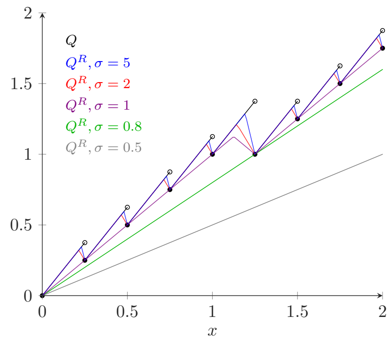


Figure 5: Regularized value functions $Q^R(\cdot;\sigma|\cdot|)$ in Example 4.6 for different σ.

Another effect is that using an arbitrary norm, the regularized subproblems (13) are no longer MILPs, but MINLPs. This is unfavorable from a computational perspective. However, at least for the (weighted) ℓ^1-norm or ℓ^∞-norm, an equivalent MILP reformulation can be achieved, so we do not leave the class of MILP subproblems [1].

Lemma 4.7. *If the norm* $\|\cdot\|$ *used in* (13) *is the (weighted)* ℓ^1*-norm or* ℓ^∞*-norm, the problem remains MILP-representable.*

4.2 Special regularized value functions

We introduce different variations of regularized subproblems and value functions which we require in the next few subsections. Moreover, we cover some of their basic properties.

- Similarly to (8) in the non-regularized case, we define *approximate regularized value functions* for each $n \in \overline{\mathcal{N}}$ as

$$\underline{Q}_n^{R;i+1}(x_{a(n)}^i; \sigma_n\|\cdot\|) := \begin{aligned} \min_{x_n, y_n, z_n, \theta_{\mathcal{C}(n)}} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} + \sigma_n \|x_{a(n)}^i - z_n\| \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}_n^{i+1}. \end{aligned} \tag{15}$$

  We also define $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(x_n; \sigma_{\mathcal{C}(n)}\|\cdot\|) := \sum_{m\in\mathcal{C}(n)} p_{nm} \underline{Q}_m^{R;i+1}(x_n; \sigma_m\|\cdot\|)$ as the *expected approximate regularized value function* for all $x_n \in \mathbb{R}^{d_n}$.

- Similarly to (12) in the non-regularized case, for each $n \in \overline{\mathcal{N}}$ we consider the *convexified regularized value function*

$$\underline{Q}_n^{CR;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|) := \begin{array}{cl} \min\limits_{x_n, y_n, z_n, \theta_{\mathcal{C}(n)}} & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} + \sigma_n \|x_{a(n)}^i - z_n\| \\ \text{s.t.} & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \operatorname{conv}(\mathcal{M}_n^{i+1}). \end{array} \tag{16}$$

- We denote the *closed convex envelope of the approximate regularized value function* by $\overline{\operatorname{co}}(\underline{Q}_n^{R;i+1}; \sigma_n \|\cdot\|)(\cdot)$. This function underestimates $\underline{Q}_n^{R;i+1}(\cdot; \sigma_n \|\cdot\|)$ on $\mathbb{R}^{d_{a(n)}}$.

The following properties are relevant to prove our main results in the next subsections.

Lemma 4.8. *Under Assumptions 1, 2, 4, given some arbitrary norm $\|\cdot\|$ and some $\sigma_n > 0$, for all $n \in \overline{\mathcal{N}}$,*

(a) the function $\underline{Q}_n^{R;i+1}(\cdot; \sigma_n \|\cdot\|)$ is finite-valued and σ_n-Lipschitz continuous on $\mathbb{R}^{d_{a(n)}}$,

(b) the function $\underline{Q}_n^{CR;i+1}(\cdot; \sigma_n \|\cdot\|)$ is finite-valued, convex and σ_n-Lipschitz continuous on $\mathbb{R}^{d_{a(n)}}$,

(c) the function $\overline{\operatorname{co}}(\underline{Q}_n^{R;i+1}; \sigma_n \|\cdot\|)(\cdot)$ is finite-valued and convex on $\mathbb{R}^{d_{a(n)}}$,

(d) the function $\underline{Q}_n^{CR;i+1}(\cdot; \sigma_n \|\cdot\|)$ is equivalent to $(\underline{Q}_n^{CR;i+1}; \sigma_n \|\cdot\|)^{**}(\cdot)$ on $\mathbb{R}^{d_{a(n)}}$.

Moreover, we need the following auxiliary result.

Lemma 4.9. *For all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$ we have*

$$(\underline{Q}_n^{R;i+1}; \sigma_n \|\cdot\|)^{**}(x_{a(n)}) = (\underline{Q}_n^{CR;i+1}; \sigma_n \|\cdot\|)^{**}(x_{a(n)}).$$

Lemma 4.8 and Lemma 4.9 are proven in Appendix G and Appendix H, respectively.

4.3 A primal convexification result

In the remainder of Sect. 4, we focus on the generation of linear Lagrangian cuts in the context of regularized subproblems and value functions (13).

Recall that in the non-regularized case (see Sect. 3.4), these cuts are generated based on a relaxation of the copy constraints. This approach cannot be applied in the regularized case, as the copy constraints are already relaxed and penalized in the regularized subproblems (13). However, we show that still cuts with similar properties can be obtained by considering specific *bounded* Lagrangian dual problems.

As a first key step, we introduce a primal convexification result for bounded Lagrangian dual problems. More precisely, we show that the convexified regularized problem (16) is closely related to the bounded Lagrangian dual problem

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|) := \begin{array}{cl} \max\limits_{\pi_n} & \mathcal{L}_n^{i+1}(\pi_n) + \pi_n^\top x_{a(n)}^i \\ \text{s.t.} & \|\pi_n\|_* \leq \sigma_n, \end{array} \tag{17}$$

where $\|\cdot\|_*$ denotes the dual norm to the norm $\|\cdot\|$ used in the regularized subproblem.

Theorem 4.10. *Under Assumptions 1, 2, and 4, given some arbitrary norm $\|\cdot\|$ and some $\sigma_n > 0$, the bounded Lagrangian dual* (17) *satisfies*

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|) = \underline{Q}_n^{CR;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|).$$

Proof. In this proof, we use sup and inf operators to be rigorous with regard to suprema and infima being attained. For notational simplicity, we set $\lambda_n := (x_n, y_n, \theta_{\mathcal{C}(n)})$ and then define

$$c_n^\top \lambda_n := f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} \tag{18}$$

with an appropriate coefficient vector c_n. Using this notation, the value function to the Lagrangian dual becomes

$$\begin{aligned}
\underline{Q}_n^{DR;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|) &= \sup_{\|\pi_n\|_* \le \sigma_n} \mathcal{L}_n^{i+1}(\pi_n) + \pi_n^\top x_{a(n)}^i \\
&= \sup_{\|\pi_n\|_* \le \sigma_n} \inf_{(z_n, \lambda_n) \in \mathcal{M}_n^{i+1}} \Big\{ c_n^\top \lambda_n + \pi_n^\top (x_{a(n)}^i - z_n) \Big\} \\
&= \sup_{\|\pi_n\|_* \le \sigma_n} \inf_{(z_n, \lambda_n) \in \operatorname{conv}(\mathcal{M}_n^{i+1})} \Big\{ c_n^\top \lambda_n + \pi_n^\top (x_{a(n)}^i - z_n) \Big\}.
\end{aligned} \tag{19}$$

The last line follows since the objective of the inner problem is linear.

We now consider the dual problem where we swap the sup and inf operators. As we discuss below, strong duality holds.

$$\begin{aligned}
&= \inf_{(z_n, \lambda_n) \in \operatorname{conv}(\mathcal{M}_n^{i+1})} \sup_{\|\pi_n\|_* \le \sigma_n} \Big\{ c_n^\top \lambda_n + \pi_n^\top (x_{a(n)}^i - z_n) \Big\} \\
&= \inf_{(z_n, \lambda_n) \in \operatorname{conv}(\mathcal{M}_n^{i+1})} \Big\{ c_n^\top \lambda_n + \sigma_n \sup_{\|\frac{\pi_n}{\sigma_n}\|_* \le 1} \Big\{ \Big(\frac{\pi_n}{\sigma_n}\Big)^\top (x_{a(n)}^i - z_n) \Big\} \Big\} \\
&= \inf_{(z_n, \lambda_n) \in \operatorname{conv}(\mathcal{M}_n^{i+1})} \Big\{ c_n^\top \lambda_n + \sigma_n \|x_{a(n)}^i - z_n\| \Big\}.
\end{aligned} \tag{20}$$

Here, we used the definition of dual norms. As is shown in Appendix G, in problem (16) always a finite infimum is attained, so in the last line we may replace the infimum with a minimum. Substituting c_n and λ_n with their definitions, we obtain exactly the definition of the function $\underline{Q}_n^{CR;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|)$.

It remains to be shown that we have strong duality between problems (19) and (20). First, according to Lemma 3.4, the set $\operatorname{conv}(\mathcal{M}_n^{i+1})$ is a closed polyhedron. By its relaxation property and Assumption 1, it is also non-empty. Therefore, both sets $\operatorname{conv}(\mathcal{M}_n^{i+1})$ and $\big\{\pi_n \in \mathbb{R}^{d_{a(n)}} : \|\pi_n\|_* \le \sigma_n\big\}$ are closed convex and non-empty, with the latter also bounded. Moreover, the objective is linear in π_n for fixed (z_n, λ_n) and vice versa. Hence, we can apply the minimax theorem from [23, Corollary 37.3.2] to infer strong duality. □

Remark 4.11. *While the duality between multiplier bounds and a Lipschitz regularization (based on the duality of norms) is known in the literature on multistage stochastic programming, see for instance [17, Proposition 4.2], [30, Proposition 5], [29, Lemma 2], to our knowledge the result with respect to convexification in Theorem 4.10 has never been discussed and explicitly proven before. Also in the literature on convex analysis, e.g. [2, 4, 23], we are not aware of any mentioning of the above result, as the discussion* *is usually limited to results for* unbounded *Lagrangian dual problems, or exactness of general augmented Lagrangian dual problems with respect to the original primal problem, not its Lipschitz regularization or convexification. A related result to Theorem 4.10 is presented in [29, Proposition 4] for the* true *regularized value function* $Q_n^R(\cdot;\sigma_n\|\cdot\|)$ *instead of its closed convex envelope, given that* $Z_{a(n)} = X_{a(n)}$ *is compact and that* $Q_n(\cdot)$ *is convex. Similar results are proven in [30] in a distributionally robust setting. Another related, but different result is the primal characterization for general augmented Lagrangian dual problems in [12, Theorem 1].*

4.4 Convex envelopes from bounded Lagrangian duals

An important question is whether the primal convexification result from Theorem 4.10 can also be linked to the closed convex envelope $\overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(\cdot)$, as it was the case for the non-regularized case (cf. Theorems 3.8, 3.9). We prove this now.

Corollary 4.12. *Consider the regularized subproblem* (15) *and the corresponding bounded Lagrangian dual* (17) *given some arbitrary norm* $\|\cdot\|$ *and some* $\sigma_n > 0$. *Under Assumptions 1, 2, 4, for all* $x_{a(n)}^i \in \mathbb{R}^{d_{a(n)}}$ *we have*

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|) = \overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(x_{a(n)}^i).$$

Proof. From Lemma 4.8 (d) and Lemma 4.9 we can conclude that for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$ we have

$$\underline{Q}_n^{CR;i+1}(x_{a(n)};\sigma_n\|\cdot\|) = (\underline{Q}_n^{CR;i+1};\sigma_n\|\cdot\|)^{**}(x_{a(n)}) = (\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)^{**}(x_{a(n)}). \tag{21}$$

Furthermore, from Lemma 4.8 (c) we know that $\overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(\cdot)$ is proper. By Proposition 1.6.1 (d) in [4] we then have $\overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(x_{a(n)}) = (\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)^{**}(x_{a(n)})$ for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$. Hence, with (21) it follows that

$$\underline{Q}_n^{CR;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|) = \overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(x_{a(n)}^i)$$

for all $x_{a(n)}^i \in \mathbb{R}^{d_{a(n)}}$. The primal convexification result in Theorem 4.10 yields

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|) = \overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(x_{a(n)}^i).$$

□

Corollary 4.12 directly implies the following result for Lagrangian cuts that are computed using the bounded Lagrangian dual problem.

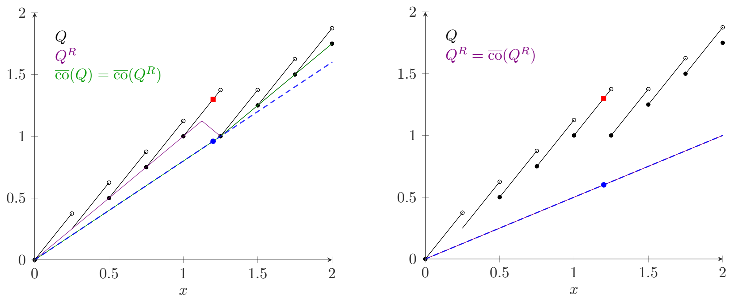
Corollary 4.13. *Consider Lagrangian cuts as defined in Definition 3.10, but with optimal multipliers* π_n^i *for the bounded Lagrangian dual problem* (17). *Then, these cuts are*

(a) valid lower approximations of $Q_n^R(\cdot;\sigma_n\|\cdot\|)$, *and thus also for* $Q_n(\cdot)$ *for all* $x_{a(n)} \in Z_{a(n)}$,

(b) tight for $\overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(\cdot)$ *at* $x_{a(n)}^i$.

We illustrate Corollary 4.13 with an example.

Example 4.14. *Consider the problem* (7) *with incumbent* $x = \frac{6}{5}$*. We use absolute value* $|\cdot|$ *as the penalty function in* (13) *and to bound the dual multipliers in* (17)*. Solving the Lagrangian dual problem for* $\sigma \geq \frac{4}{5}$*, we obtain the cut* $\theta \geq \frac{4}{5}x$*. For* $\sigma < \frac{4}{5}$*, in contrast, the resulting cut is* $\theta \geq \sigma x$*. Fig. 6 displays these cuts (blue broken lines) for* $\sigma = 1$ *and* $\sigma = \frac{1}{2}$*. As we can see, in both cases, the cut is tight for* $\overline{\text{co}}(Q^R; \sigma|\cdot|)(\cdot)$ *at* $x = \frac{6}{5}$ *(blue dots).*



(a) $\sigma = 1$. (b) $\sigma = \frac{1}{2}$.

Figure 6: (Regularized) value functions and Lagrangian cuts derived from (17) for Example 4.14.

4.5 The case of Lipschitz continuous value functions

We consider the special case, where for all $n \in \overline{\mathcal{N}}$, the value functions $Q_n(\cdot)$ are already Lipschitz continuous, *e.g.*, because a strong recourse assumption like complete continuous recourse is satisfied [1, 31]. In that case, a regularization of the MS-MILP is not required. However, we may still state a theoretical relation between the regularized and the non-regularized problem.

Lemma 4.15. *For all* $n \in \overline{\mathcal{N}}$*, let* $Q_n(\cdot)$ *be Lipschitz continuous on* $\text{dom}(Q_n)$ *with respect to some norm* $\|\cdot\|$ *with Lipschitz constant* α_n*. Then for* $\sigma_n \geq \alpha_n$ *we have*

$$Q_n^R(x_{a(n)}; \sigma_n\|\cdot\|) = Q_n(x_{a(n)})$$

for all $x_{a(n)} \in \text{dom}(Q_n)$*. Analogously, for all* $x_{a(n)} \in \text{dom}(\underline{Q}_n^{i+1})$

$$\underline{Q}_n^{R;i+1}(x_{a(n)}; \sigma_n\|\cdot\|) = \underline{Q}_n^{i+1}(x_{a(n)}).$$

For leaf nodes $n \in \mathcal{N}$, this result follows immediately from [2, Corollary 12.18], considering that the regularized value functions are the Pasch-Hausdorff envelopes of the non-regularized ones. For other nodes in $\mathcal{N}$ it can then be shown inductively.

As already noticed in [30], this result shows that computationally regularization may even prove beneficial if the original value functions are already guaranteed to be Lipschitz continuous. First, the Lagrangian dual problem can be bounded. Second, cuts with larger Lipschitz constant than the value function can be excluded from its approximation.

Additionally, Lemma 4.15 has a helpful implication that we use in the next section when discussing the SDDiP setting again. It can be used to show that for sufficiently large σ_n, the lower convex envelopes of the regularized and non-regularized approximate value functions do coincide.

Lemma 4.16. *Under Assumptions 1, 2, and 4, for all $n \in \overline{\mathcal{N}}$, there exists some $\sigma_n > 0$ such that for all $x_{a(n)} \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$*

$$\overline{\operatorname{co}}(\underline{Q}_n^{R;i+1}; \sigma_n \|\cdot\|)(x_{a(n)}) = \overline{\operatorname{co}}(\underline{Q}_n^{i+1})(x_{a(n)}).$$

We provide a proof in Appendix I.

Example 4.17. *As an illustration for Lemma 4.16, see Example 4.6 and Fig. 5. We can see that for $\sigma \geq 1$, $\overline{\operatorname{co}}(Q)(\cdot)$ and $\overline{\operatorname{co}}(Q^R; \sigma|\cdot|)(\cdot)$ do coincide.*

4.6 The case of tight Lagrangian cuts

We consider cases where tightness for function $\underline{Q}_n^{R;i+1}(\cdot; \sigma_n\|\cdot\|)$, and not only its closed convex envelope $\overline{\operatorname{co}}(\underline{Q}_n^{i+1})(\cdot)$, can be obtained using Lagrangian cuts. This includes the case of binary state variables from SDDiP [31].

Note that we already discussed similar cut generation results for the *true* value functions in Sect. 3.5, without the requirement of Lipschitz continuity of $Q_n(\cdot)$ or $\underline{Q}_n^{i+1}(\cdot)$. For this reason, considering a Lipschitz regularization in this case or similar cases may seem superfluous. However, we still briefly discuss it in this section, as some of the results prove beneficial later in Sect. 5 when we deal with *non-convex* approximations of the value functions.

Importantly, compared to Sect. 3.5, the extreme point argument used in the proof of Theorem 3.13 is no longer valid in the regularized setting. The two functions, $\overline{\operatorname{co}}(\underline{Q}_n^{R;i+1}; \sigma_n\|\cdot\|)(\cdot)$ and $\underline{Q}_n^{R;i+1}(\cdot; \sigma_n\|\cdot\|)$ share the effective domain $\mathbb{R}^{d_{a(n)}}$, and hence, it is not clear whether they coincide for all $x_{a(n)} \in X_{a(n)}$. Nonetheless, under some assumptions, the intended tightness result can be established.

Case 1: Using sufficiently large σ_n. For sufficiently large, but finite regularization parameters $\sigma_n > 0$, we immediately obtain the tightness result using Lemma 4.16.

Lemma 4.18. *Under Assumptions 1, 2, 4, for any iteration i and any node $n \in \overline{\mathcal{N}}$, let $Z_{a(n)}$ be bounded and let $X_{a(n)}$ be contained in its extreme points. Let $\|\cdot\|$ be some arbitrary norm and $\sigma_n > 0$ sufficiently large for all $n \in \mathcal{N}$. Then, for all $x_{a(n)}^i \in X_{a(n)} \cap \operatorname{dom}(\underline{Q}_n^{i+1})$ we have*

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i; \sigma_n\|\cdot\|) = \underline{Q}_n^{i+1}(x_{a(n)}^i) = \underline{Q}_n^{R;i+1}(x_{a(n)}^i; \sigma_n\|\cdot\|).$$

We provide a proof in Appendix J.

This result is not surprising considering the exact penalization result for Lipschitz regularization from Lemma 4.5. Interestingly, in the SDDiP case of binary state variables $x_{a(n)}$, tightness for the regularized value function can even be obtained independent of σ_n, as we show next.

Case 2: Regularization with the ℓ^1-norm. Suppose we use the (weighted) ℓ^1-norm in the regularization. Then we can derive the following auxiliary result, which has already been proven in [13, Lemma 3.8] in a slightly different form. The proof is given in Appendix K.

Lemma 4.19. *Let $X_{a(n)} = \{0,1\}^{d_{a(n)}}$ for all $n \in \mathcal{N}$ and $Z_{a(n)} = X_{a(n)}$ or $Z_{a(n)} = \operatorname{conv}(X_{a(n)})$. Then, for any iteration i, any node $n \in \overline{\mathcal{N}}$ and any $\sigma_n > 0$, the regularized subproblem* (15) *and the bounded Lagrangian dual* (17) *for the ℓ^1-norm satisfy*

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i; \sigma_n\|\cdot\|_1) \geq \underline{Q}_n^{R;i+1}(x_{a(n)}^i; \sigma_n\|\cdot\|_1). \tag{22}$$

Using this lemma, we obtain the intended tightness result.

Corollary 4.20. *The inequality in Lemma 4.19 is satisfied with equality.*

Proof. From Lemma 4.19 we have relation (22). On the other hand, from Corollary 4.12 and the definition of the closed convex envelope

$$\underline{Q}_n^{DR;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|_1) = \overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|_1)(x_{a(n)}^i) \leq \underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|_1).$$

□

We illustrate the previous results using problem (6) below.

Example 4.21. *Consider the problem* (6) *with incumbent* $x = 1$ *and* $Z = \text{conv}(X) = [0,1]$*. We use the absolute value* $|\cdot|$ *as penalty function in* (13) *and set* $\sigma = 2$*. Fig. 7 shows that the regularized value function* $Q^R(\cdot;\sigma|\cdot|)$ *is monotonically increasing outside of* Z*, and thus coincides with its convex envelope outside of* Z*. For this reason, both functions also coincide at extreme points of* Z*, such as* $x = 1$*. The obtained Lagrangian cut* $\theta \geq -\frac{1}{3} + 2x$ *is tight for* $Q^R(\cdot;\sigma|\cdot|)$ *at* $x = 1$*, in accordance with Corollary 4.20.*

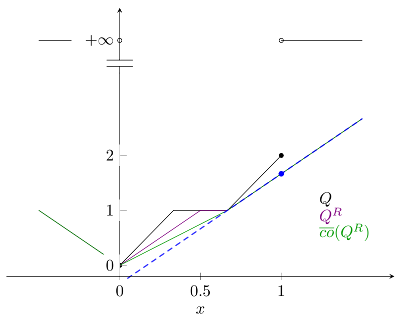


Figure 7: Lagrangian cut for the regularized value function with $\sigma = 2$ in problem (6).

We note that in the multidimensional case for norms different than $\|\cdot\|_1$*, this is not necessarily true. We present an example for this in Example 5.14 (3) in the next section.*

5 Lipschitz regularization and non-convex cuts

In the previous section we pointed out that Lagrangian cuts for regularized value functions can be generated by solving bounded Lagrangian dual problems. These cuts are tight for the closed convex envelopes of the regularized value functions, as long as the dual problem is solved to optimality. Just as in the non-regularized case, in general, these cuts are not guaranteed to be tight for the true value functions, though, and thus cannot guarantee convergence of decomposition methods for MS-MILPs.

In this section, we deal with the alternative approach to generate non-convex approximations $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ of the expected value functions $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ in order to ensure convergence, again by exploiting Lipschitz regularization. We specifically focus on the alternative lift-and-project approach from [13], which is part of the NC-NBD method presented in the same paper. We first give a brief introduction into its main concepts and then present its theoretical backbone in a rigorous way. In particular, we close an open question on how to ensure Lipschitz continuity of the obtained non-convex approximations. This allows us to drop the technical Assumption 4 in [13]. While NC-NBD in [13] assumes a deterministic problem, we enhance its ideas to the stochastic setting here.

5.1 The lift-and-project idea

As a basis to describe the lift-and-project cut generation idea, we consider the approximate regularized value function from Sect. 4 for some node $n \in \overline{\mathcal{N}}$.

$$\begin{aligned} \underline{Q}_n^{R;i+1}(x_{a(n)}^i; \sigma_n \|\cdot\|^\circ) = \min_{x_n, y_n, z_n, \theta_{\mathcal{C}(n)}} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} + \sigma_n \|x_{a(n)}^i - z_n\|^\circ \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}_n^{i+1}. \end{aligned} \tag{23}$$

The incumbents $x_{a(n)}^i$ for all nodes $n \in \overline{\mathcal{N}}$ are computed in a forward pass through the scenario tree, which we do not describe in detail here. The objective function $f_n(\cdot)$ and all but the cut constraints in (23) are still linear. The only difference in (23) compared to Sect. 4 is that we now assume that the approximation $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, which is contained in the set $\mathcal{M}_n^{i+1}$, is non-convex. However, we assume that it can still be approximated by mixed-integer linear constraints, see Sect. 5.7 and [13], so $\mathcal{M}_n^{i+1}$ has the same properties as in Sect. 4. Notation-wise, the notation $\|\cdot\|^\circ$ is introduced to distinguish the norm used for regularization in the original state space from a second, possibly deviating norm $\|\cdot\|^\bullet$ that is used in the lifted space where cuts are generated later on.

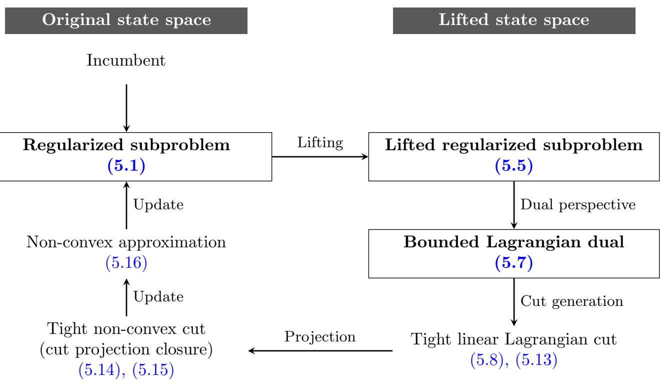

Original state space
Lifted state space
Incumbent
Regularized subproblem (5.1)
Lifting
Lifted regularized subproblem (5.5)
Update
Dual perspective
Non-convex approximation (5.16)
Bounded Lagrangian dual (5.7)
Update
Cut generation
Tight non-convex cut (cut projection closure) (5.14), (5.15)
Projection
Tight linear Lagrangian cut (5.8), (5.13)

Figure 8: The lift-and-project approach used in [13].

The main concept of the cut generation approach from [13] is illustrated in Fig. 8. In each iteration i, instead of directly generating Lagrangian cuts in the original state space, the subproblems and value functions, *e.g.*, problem (23), are first temporarily lifted to a binary state space. According to Corollary 4.20, by solving a bounded Lagrangian dual problem then *tight linear* Lagrangian cuts can be computed for the regularized value functions. However, these cuts are expressed in the lifted state space. In order to use them in the original state space, they are projected back to that space. The pointwise maximum of this projection, which we refer to as the *cut projection closure* (CPC), can be interpreted as a non-convex cut, and, as we show, it is tight for $\underline{Q}_n^{R;i+1}(\cdot; \sigma_n \|\cdot\|^\circ)$.

Another key feature of this cut generation method is that we allow for the construction of cuts at points $x_{\mathcal{B},n}^i$ differing from the current incumbents x_n^i. These points are called *anchor points* in [13]. This also means that the non-convex cuts, *i.e.*, the CPC, is guaranteed to be tight for $\underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|^\circ)$ at $x_{\mathcal{B},n}^i$ only. But as long as the distance between x_n^i and $x_{\mathcal{B},n}^i$ can be controlled, we shall see that the approximation error of the non-convex cuts at x_n^i can be controlled as well.

5.2 Sufficient non-convex approximations.

In order to ensure convergence of decomposition methods employing this lift-and-project cut generation approach, for instance the NC-NBD method in [13], the non-convex approximations $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ have to satisfy three main properties. We call a non-convex approximation satisfying these properties *sufficient.*

Definition 5.1. *Let $n \in \mathcal{N}$ and consider some arbitrary iteration $i \in \mathbb{N}$. Given some anchor point $x_{\mathcal{B},n}^i$, some norm $\|\cdot\|^\circ$ and some $\sigma_n > 0$ used in the regularized subproblem* (23)*, a non-convex approximation $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ is called* sufficient *if it*

(S1) is a valid under-approximation of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, i.e., *for all $x_n \in X_n$:*

$$\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(x_n) \leq \mathcal{Q}_{\mathcal{C}(n)}(x_n),$$

(S2) overestimates the expected approximate regularized value function at the anchor point:

$$\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(x_{\mathcal{B},n}^i) \geq \underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(x_{\mathcal{B},n}^i;\sigma_{\mathcal{C}(n)}\|\cdot\|^\circ),$$

(S3) is Lipschitz continuous with respect to the norm $\|\cdot\|^\circ$ used in subproblem (23) *with a finite Lipschitz constant.*

The reasoning behind these properties is the following: Using similar arguments as in Lemma 4.8, it can be shown that $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(\cdot;\sigma_{\mathcal{C}(n)}\|\cdot\|^\circ)$ is Lipschitz continuous. As $\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(\cdot)$ is also Lipschitz continuous according to (S3), using property (S2), the approximation error at the incumbent x_n^i can be bounded by the Lipschitz constants and the distance between x_n^i and $x_{\mathcal{B},n}^i$ [see 13, Lemma 4.1]. In our case, the anchor points $x_{\mathcal{B},n}^i$ are determined using a binary approximation of x_n^i which goes along with the lifting to the binary state space, see Sect. 5.3. Thus, the distance between x_n^i and $x_{\mathcal{B},n}^i$ can be controlled by refining the approximation precision β_n if required. As a result, also the cut approximation error at x_n^i can be controlled, and reduced sufficiently [see 13, Lemma 4.2]. Together with the validity (S1), this ensures exactness and finite convergence of the decomposition method [see 13, Theorem 4.3]. For more details on NC-NBD and its convergence proof, we refer to [13].

In the remainder of this section, we focus on showing how sufficient non-convex approximations can be obtained in the lift-and-project framework. In [13], it is already shown how properties (S1) and (S2) can be achieved for the deterministic case, but we extend these results to the scenario tree setting. Ensuring property (S3), on the other hand, is more sophisticated. Whereas it can be easily shown that it holds for a fixed precision β_n of the binary approximation [13], we have to make sure that the Lipschitz constant does not diverge for $\beta_n \to 0$. Otherwise, the reduction in distance between x_n^i and $x_{\mathcal{B},n}^i$ may be redeemed by an increasing Lipschitz constant. In other words, we have to bound the Lipschitz constant in (S3) independently of β_n (and by that i). Instead of showing how this can be achieved, in [13] a technical assumption is taken (Assumption

(A4) in [13]). We close this theoretical gap in this section. The key idea is to use a tailor-made norm $\|\cdot\|^{\bullet}$ for the regularization in the lifted space.

5.3 Lifting to the binary space

We lift the subproblems and value functions to a different space by temporarily applying a binary approximation of the state $x_{a(n)}$ [16]. For simplicity, we assume that all components of $x_{a(n)}$ satisfy bounded box constraints with a zero lower bound. Then, any component $x_{a(n),j} \in [0, U_j], j = 1, \dots, d_{a(n)}$, can be approximated by

$$x_{a(n),j} = \beta_{a(n),j} \sum_{\kappa=1}^{K_{a(n),j}} 2^{\kappa-1} \lambda_{a(n),\kappa j} + r_{a(n),j}, \tag{24}$$

with a discretization precision $\beta_{a(n),j} \in (0,1)$ if $x_{a(n),j}$ is continuous and $\beta_{a(n),j} = 1$ if it is integer. $r_{a(n),j} \in \left[-\frac{\beta_{a(n),j}}{2}, \frac{\beta_{a(n),j}}{2}\right]$ denotes the approximation error. For some vector $x_{a(n)}$, this requires $K_{a(n)} = \sum_{j=1}^{d_{a(n)}} K_{a(n),j}$ binary variables $\lambda_{a(n),\kappa j}$, with $K_{a(n),j} = \lfloor \log_2 \left(\frac{U_j}{\beta_{a(n),j}}\right) \rfloor + 1$.

We define a $(d_{a(n)} \times K_{a(n)})$-matrix $\mathcal{B}_{a(n)}$ containing all the coefficients of the binary approximation and collect all binary variables in one large vector $\lambda_{a(n)} \in \{0,1\}^{K_{a(n)}}$. Then, the binary expansion can be written compactly as

$$x_{a(n)} = \mathcal{B}_{a(n)} \lambda_{a(n)} + r_{a(n)}. \tag{25}$$

For some index $k \in K_{a(n)}$, let $j(k)$ denote the component in the original space associated with k. Then we define $\kappa(k) := k - \sum_{\ell=1}^{j(k)} K_{a(n),\ell}$ to access the correct κ in (24).

By applying (25) to a trial point $x^i_{a(n)}$ and omitting the error term, we define the anchor point as the approximation

$$x^i_{\mathcal{B},a(n)} := \mathcal{B}_{a(n)} \lambda^i_{a(n)}. \tag{26}$$

Example 5.2. *Again, we consider problem* (7) *with incumbent* $x = \frac{6}{5}$. *For different values of* β *or* K, *respectively, we obtain the anchor points*

$$\begin{aligned}
K = 2, \beta = \frac{2}{3}: \qquad & x_{\mathcal{B}} = \frac{2}{3}(2^0 \cdot 0 + 2^1 \cdot 1) = \frac{4}{3}, \\
K = 3, \beta = \frac{2}{7}: \qquad & x_{\mathcal{B}} = \frac{2}{7}(2^0 \cdot 0 + 2^1 \cdot 0 + 2^2 \cdot 1) = \frac{8}{7}, \\
K = 4, \beta = \frac{2}{15}: \qquad & x_{\mathcal{B}} = \frac{2}{15}(2^0 \cdot 1 + 2^1 \cdot 0 + 2^2 \cdot 0 + 2^3 \cdot 1) = \frac{6}{5}.
\end{aligned}$$

That is, for $K = 4$ *and* $\beta = \frac{2}{15}$, *the approximation of the incumbent is exact.*

Instead of problem (23) we now consider $\underline{Q}_n^{R;i+1}(x^i_{\mathcal{B},a(n)}; \sigma_n \|\cdot\|^{\circ})$, *i.e.*, we do not consider the approximate regularized value function at the incumbent $x^i_{a(n)}$, but at the anchor point $x^i_{\mathcal{B},a(n)}$.

Using relation (26) we can interpret $\lambda^i_{a(n),j}, j = 1, \dots, K_{a(n)}$, as the state variables in a lifted binary state space. We can thus express the function $\underline{Q}_n^{R;i+1}(x^i_{\mathcal{B},a(n)}; \sigma_n \|\cdot\|^{\circ})$

in terms of these binary state variables. To this end, we also set

$$z_n = \mathcal{B}_{a(n)} \mathfrak{z}_n \tag{27}$$

with additional variables $\mathfrak{z}_n \in [0,1]^{K_{a(n)}}$. In view of (26), this can be interpreted as first introducing and then relaxing copy constraints for each $\lambda^i_{a(n),j}, j = 1, \dots, K_{a(n)}$, separately. Additionally, we define the norm $\|\lambda_n\|_{\mathcal{B}} := \|\mathcal{B}_{a(n)}\lambda_n\|$. This yields the reformulation of the regularized subproblem (23):

$$\begin{aligned} \underline{Q}^{R;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n \|\cdot\|^\circ_{\mathcal{B}}) := \min_{x_n, y_n, z_n, \theta_{\mathcal{C}(n)}, \mathfrak{z}_n} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} + \sigma_n \|(\lambda^i_{a(n)} - \mathfrak{z}_n)\|^\circ_{\mathcal{B}} \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}^{i+1}_n \\ & z_n = \mathcal{B}_{a(n)} \mathfrak{z}_n \\ & \mathfrak{z}_n \in [0,1]^{K_{a(n)}}. \end{aligned} \tag{28}$$

This reformulation is exact in the sense that

$$\underline{Q}^{R;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n \|\cdot\|^\circ_{\mathcal{B}}) = \underline{Q}^{R;i+1}_{n}(x^i_{\mathcal{B},a(n)}; \sigma_n \|\cdot\|^\circ). \tag{29}$$

In the same vein, we may define $Q_{\mathcal{B};n}(\cdot)$ as the true value function $Q_n(\cdot)$ expressed as a function in the lifted state space.

Remark 5.3. *Recall our discussion on different choices of $Z_{a(n)}$ and their impact in Sect. 3.1. While the choice of $Z_{a(n)}$ in the original state space is not particularly relevant in our lift-and-project setting now, the choice of bounding $\mathfrak{z}_n$ in the lifted space using the convex hull $[0,1]^{K_{a(n)}}$ instead of $\{0,1\}^{K_{a(n)}}$ as the accompanying set is crucial if some components of $x_{a(n)}$ are continuous. First, in contrast to (26) the reformulation (27) of z_n is exact given this choice, even for continuous variables. Second, it ensures that linear cuts generated in the lifted binary state space are valid for the non-convex expected value functions on the whole set $[0,1]^{K_{a(n)}}$. This is inevitable in order to obtain non-convex cuts in the original state space which are valid underestimators of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ even for points in $X_{a(n)}$ that cannot be exactly represented by the current binary approximation.*

5.4 Generating Lagrangian cuts in the lifted space

To generate Lagrangian cuts in the lifted binary state space, we follow Sect. 4 and consider a bounded Lagrangian dual problem

$$\underline{Q}^{DR;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n \|\cdot\|^\bullet) := \max_{\|\pi_n\|^\bullet_* \leq \sigma_n} \quad \mathcal{L}^{i+1}_{\mathcal{B};n}(\pi_n) + \pi_n^\top \lambda^i_{a(n)}. \tag{30}$$

The dual function $\mathcal{L}^{i+1}_{\mathcal{B};n}(\cdot)$ is defined by

$$\begin{aligned} \mathcal{L}^{i+1}_{\mathcal{B};n}(\pi_n) := \min_{x_n, y_n, z_n, \theta_{\mathcal{C}(n)}, \mathfrak{z}_n} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} - \pi_n^\top \mathfrak{z}_n \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}^{i+1}_n \\ & z_n = \mathcal{B}_{a(n)} \mathfrak{z}_n \\ & \mathfrak{z}_n \in [0,1]^{K_{a(n)}}. \end{aligned}$$

Solving the dual problem (30) we obtain optimal multipliers π^i_n. We can then build the function

$$\phi_{\mathcal{B};n}(\lambda_{a(n)}) := \mathcal{L}^{i+1}_{\mathcal{B};n}(\pi^i_n) + (\pi^i_n)^\top \lambda_{a(n)}, \tag{31}$$

which defines a linear Lagrangian cut in the binary space $\{0,1\}^{K_{a(n)}}$.

The crucial part, and a new contribution compared to [13], is how we choose the norm $\|\cdot\|^\bullet_*$ in problem (30) to bound the dual multipliers. Let $\|\cdot\|$ be an arbitrary norm and $\|\cdot\|_*$ its dual norm. Furthermore, let W and $\widehat{W}$ be some diagonal weight matrices whose diagonal entries at row k and column k satisfy the relation $\widehat{w}_{kk} = w^{-1}_{kk}$ (for simplicity, we omit indices for W and $\widehat{W}$). Then, $\|x\|_w := \|Wx\|$ defines the *weighted* norm and $\|x\|_{w*} := \|\widehat{W}x\|_*$ defines the dual weighted norm for some vector x.

For each component $k = 1, \dots, K_{a(n)}$ in the binary state space, we now define weights

$$w_{kk} = 2^{\kappa(k)-1}\beta_{a(n),j(k)},$$

and choose $\|\cdot\|^\bullet = \|\cdot\|_w$ given some norm $\|\cdot\|$. The motivation behind this choice is to bound the dual multipliers in such a way that the effects of the binary approximation are compensated by the weights. Note that these bounds are tighter than the ones originally proposed in [13] where no weighted norms are used.

With this construction, we observe that the matrix W and the matrix $\mathcal{B}_{a(n)}$ are closely related. Both matrices contain the same non-negative entries, but W is a $(K_{a(n)} \times K_{a(n)})$-diagonal matrix, whereas $\mathcal{B}_{a(n)}$ is a $(d_{a(n)} \times K_{a(n)})$-matrix where non-negative entries corresponding to the same component j of the original state space occur in the same row. Hence, we can define a matrix G such that $\mathcal{B}_{a(n)} = GW$. This matrix contains only ones and zeros, with several ones in each row, but only a single one in each column.

For some matrix A, let $\|A\|$ be the matrix norm induced by $\|\cdot\|$. Then, the consistency of matrix norms and the inducing vector norm yields the relation

$$\begin{aligned} \|\mathcal{B}_{a(n)}(\lambda^i_{a(n)} - \mathfrak{z}_n)\| = \|GW(\lambda^i_{a(n)} - \mathfrak{z}_n)\| &\le \|G\| \; \|W(\lambda^i_{a(n)} - \mathfrak{z}_n)\| \\ &= \|G\| \; \|\lambda^i_{a(n)} - \mathfrak{z}_n\|_w. \end{aligned} \tag{32}$$

5.5 Properties of Lagrangian cuts in the lifted space

Recall Remark 5.3. As shown in [13], by choosing $\mathfrak{z}_n \in [0,1]^{K_{a(n)}}$, the function $\phi_{\mathcal{B};n}(\cdot)$ defined in (31) provides a valid underestimator for the true value function in the binary state space, but also everywhere in the original state space. This is crucial to prove property (S1) in the next section.

Lemma 5.4 (Lemma 3.7 in [13]). *The function $\phi_{\mathcal{B};n}(\cdot)$ satisfies*

$$Q_{\mathcal{B};n}(\lambda_{a(n)}) \ge \phi_{\mathcal{B};n}(\lambda_{a(n)}),$$

for all $\lambda_{a(n)} \in [0,1]^{K_{a(n)}}$, and

$$Q_n(x_{a(n)}) \ge \phi_{\mathcal{B};n}(\lambda_{a(n)})$$

for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$ and any $\lambda_{a(n)} \in [0,1]^{K_{a(n)}}$, such that $x_{a(n)} = \mathcal{B}_{a(n)}\lambda_{a(n)}$.

Next, we use the results from Sect. 4.6 to obtain some overestimation results with respect to property (S2). Recall that one way to achieve tightness presented in Sect. 4.6

is to choose some sufficiently large but finite $\sigma_n > 0$. While this is sufficient to derive cuts that satisfy properties (S2) and (S3) for some fixed binary precision $\beta_{a(n),j}$, this is not the case if we consider binary refinements. In that case, after each refinement we may require a larger σ_n, such that the sequence of these values diverges. This is detrimental in ensuring property (S3). Therefore, we directly focus on the second case in Sect. 4.6 and choose the ℓ^1-norm. Importantly, Lemma 4.19 and Corollary 4.20 still hold if we use a weighted ℓ^1-norm.

Corollary 5.5. *Choosing* $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ *in* (30) *it follows*

$$\phi_{\mathcal{B};n}(\lambda^i_{a(n)}) = \underline{Q}^{R;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n\|\cdot\|^\bullet). \tag{33}$$

Equation (33) is sufficient to achieve the overestimation property (S2) in Definition 30 if $\|\cdot\|^\circ$ is any ℓ_p-norm, as we show now.

Lemma 5.6. *Let* $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ *in problem* (30) *and let* $\|\cdot\|^\circ$ *in problem* (23) *be any* ℓ_p*-norm. Then,*

$$\phi_{\mathcal{B};n}(\lambda^i_{a(n)}) \geq \underline{Q}^{R;i+1}_{n}(x^i_{\mathcal{B},a(n)}; \sigma_n\|\cdot\|^\circ).$$

Proof. For the maximum absolute column sum norm we have $\|G\|_1 = 1$, since G contains at most a single one in each column. Hence, from (32) it follows

$$\|\mathcal{B}_{a(n)}(\lambda^i_{a(n)} - \mathfrak{z}_n)\|_1 \leq \|G\|_1 \; \|\lambda^i_{a(n)} - \mathfrak{z}_n\|_{1,w} = \|\lambda^i_{a(n)} - \mathfrak{z}_n\|_{1,w}. \tag{34}$$

Moreover, we have

$$\|\mathcal{B}_{a(n)}(\lambda^i_{a(n)} - \mathfrak{z}_n)\|_1 \geq \|\mathcal{B}_{a(n)}(\lambda^i_{a(n)} - \mathfrak{z}_n)\|_p \tag{35}$$

for any ℓ_p-norm. Combining some of our previous results and exploiting that $\|\cdot\|^\circ$ is an ℓ_p-norm, we obtain

$$\begin{aligned} \phi_{\mathcal{B};n}(\lambda^i_{a(n)}) &\overset{(33)}{=} \underline{Q}^{R;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n\|\cdot\|_{1,w}) \overset{(34)}{\geq} \underline{Q}^{R;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n\|\cdot\|_{1,\mathcal{B}}) \\ &\overset{(35)}{\geq} \underline{Q}^{R;i+1}_{\mathcal{B};n}(\lambda^i_{a(n)}; \sigma_n\|\cdot\|^\circ_{\mathcal{B}}) \overset{(29)}{=} \underline{Q}^{R;i+1}_{n}(x^i_{\mathcal{B},a(n)}; \sigma_n\|\cdot\|^\circ). \end{aligned}$$

□

To derive a Lagrangian cut for $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, we aggregate the functions $\phi_{\mathcal{B};m}(\cdot)$ for all $m \in \mathcal{C}(n)$:

$$\begin{aligned} \phi_{\mathcal{B};\mathcal{C}(n)}(\lambda_n) &:= \sum_{m\in\mathcal{C}(n)} p_{nm}\big(\mathcal{L}^{i+1}_{\mathcal{B};m}(\pi^i_m) + (\pi^i_m)^\top \lambda_n\big) \\ &= \underbrace{\sum_{m\in\mathcal{C}(n)} p_{nm}\mathcal{L}^{i+1}_{\mathcal{B};m}(\pi^i_m)}_{=:\gamma^i_{\mathcal{C}(n)}} + \underbrace{\Big(\sum_{m\in\mathcal{C}(n)} p_{nm}\pi^i_m\Big)}_{=:\pi^i_{\mathcal{C}(n)}}^\top \lambda_n \end{aligned} \tag{36}$$

The previous validity and overestimation results naturally extend to this aggregated function.

Corollary 5.7. *The function* $\phi_{\mathcal{B};\mathcal{C}(n)}(\cdot)$ *satisfies*

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \phi_{\mathcal{B};\mathcal{C}(n)}(\lambda_n)$$

for all $x_n \in \mathbb{R}^{d_n}$ and any $\lambda_n \in [0,1]^{K_n}$, such that $x_n = \mathcal{B}_n \lambda_n$.

Corollary 5.8. *Let $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ in* (30) *and let $\|\cdot\|^\circ$ in problem* (23) *be any ℓ_p-norm. Then,*

$$\phi_{\mathcal{B};\mathcal{C}(n)}(\lambda_n^i) \geq \underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(x_{\mathcal{B},n}^i; \sigma_n \|\cdot\|^\circ).$$

We now address the projection of these linear cuts back to the original state space.

5.6 The cut projection closure

As explained in Sect. 5.1, the lifting to the binary state space is carried out only temporarily to generate *tight* linear Lagrangian cuts. Importantly, according to Corollary 5.7, these cuts also allow us to obtain valid underapproximations of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ for all points $x_n \in X_n$, even those which cannot be exactly represented by the current binary approximation. More precisely, for some given $x_n \in X_n$, each $\lambda_n \in [0,1]^{K_n}$ such that $x_n = \mathcal{B}_n \lambda_n$ provides a valid underestimator for $\mathcal{Q}_{\mathcal{C}(n)}(x_n)$. For some given $x_n \in X_n$ there may exist infinitely many such configurations for λ_n, and thus infinitely many underestimators. We are interested in the pointwise supremum of all these underestimators, that is, the tightest underestimating function that can be gained from projecting the cut to x_n. We refer to this supremum as the cut projection closure.

Definition 5.9 (Cut projection closure). *Let $\phi_{\mathcal{B};\mathcal{C}(n)} : [0,1]^{K_n} \to \mathbb{R}$ be a cut-defining linear function given in* (36)*. Then, the cut projection closure (CPC) $\phi_{\mathcal{C}(n)} : \mathbb{R}^{d_n} \to \mathbb{R}$ is defined as*

$$\phi_{\mathcal{C}(n)}(x_n) := \max_{\lambda_n} \left\{ \gamma_{\mathcal{C}(n)} + \pi_{\mathcal{C}(n)}^\top \lambda_n \ : \ \mathcal{B}_n \lambda_n = x_n, \lambda_n \leq e, \lambda_n \geq 0 \right\}. \tag{37}$$

Here, e is a unit vector of dimension K_n.

By strong duality of linear programs, the CPC can be equivalently expressed as

$$\phi_{\mathcal{C}(n)}(x_n) = \min_{\eta_n, \mu_n} \left\{ \gamma_{\mathcal{C}(n)} + x_n^\top \eta_n + e^\top \mu_n \ : \ \mathcal{B}_n^\top \eta_n + \mu_n \geq \pi_{\mathcal{C}(n)}, \mu_n \geq 0 \right\}. \tag{38}$$

Importantly, the dual feasible region in (38) does not depend on x_n and has a finite number of extreme points for a given binary precision. Therefore, we can conclude:

Lemma 5.10. *The CPC $\phi_{\mathcal{C}(n)}(\cdot)$ is a piecewise linear and concave function in $\mathbb{R}^{d_{a(n)}}$.*

The CPC is a piecewise linear function, and the slope of each piece is determined by the value of η_n in an extreme point of (38). Therefore, we analyze these extreme points in more detail. Based on our findings from the previous section, we choose $\|\cdot\|^\bullet$ as the weighted ℓ^1-norm again. Additionally, we define $\sigma_n^{\max} := \max_{m \in \mathcal{C}(n)} \sigma_m$. Then, as proven in Appendix L, we obtain:

Lemma 5.11. *Let $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ in problem* (30)*. Then, for any binary precision $\beta_{n,j} \in (0,1), j = 1, \ldots, d_n$, any extreme point of problem* (38) *satisfies $\|\eta_n\|_\infty \leq \sigma_n^{\max}$.*

The crucial idea here is that by a careful choice of the weighted norm to bound the Lagrangian dual (30), effects of the binary expansion are compensated, such that each component of η_n can be bounded independently of the current binary precision $\beta_{n,j} \in (0,1), j = 1, \ldots, d_n$, and the number K_n of binary variables. Therefore, this bound remains valid for any refinement of the binary precision in NC-NBD [13], and even with these refinements the CPC is prevented from becoming infinitely steep. This is stated in the following lemma, which is proven in Appendix M.

Lemma 5.12. *Let $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ in problem* (30)*. Then, for any norm $\|\cdot\|^\circ$, the CPC is a $\tilde{\sigma}_{\mathcal{C}(n)}$-Lipschitz continuous function with $\tilde{\sigma}_{\mathcal{C}(n)} > 0$ independent of the binary precision $\beta_{n,j} \in (0,1), j = 1, \dots, d_n$.*

5.7 Main result

For any node $n \in \mathcal{N}$, using the CPC, we can now determine the non-convex outer approximation of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ as

$$\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(x_n) = \min\Big\{\theta_{\mathcal{C}(n)} \in \mathbb{R} \;:\; \theta_{\mathcal{C}(n)} \geq \phi^r_{\mathcal{C}(n)}(x_n) \;\forall r = 1, \dots, i+1 \Big\}. \tag{39}$$

Based on our previous findings we can now state conditions under which this non-convex approximation is sufficient in the sense of Definition 5.1, which is the main result of this section. We provide a proof in Appendix N.

Theorem 5.13. *Let $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ in problem* (30)*, let $\|\cdot\|^\circ$ in problem* (23) *be any ℓ_p-norm and let $\sigma_n > 0$. Then, $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ as defined in* (39) *is a sufficient non-convex approximation of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$.*

As pointed out in Sect. 5.2, a sufficient non-convex approximation of $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ (with appropriately chosen set $Z_{a(n)} \supseteq X_{a(n)}$) is sufficient to guarantee convergence of the NC-NBD inner loop, and by that of NC-NBD in total, without the requirement of the technical assumption (A4) in [13]. The condition of setting $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ in problem (30) in the lifted space to achieve such approximation is not really strict, since it still allows to choose any ℓ_p-norm for the regularization in problem (23) in the original state space.

Finally, let us emphasize that the CPC is a non-convex function outer approximating $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$, and defined by the linear programs (37) and (38). Therefore, directly incorporating it into $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ in subproblem (8) leads to a non-convex bilevel problem. In order to resolve this issue, in [13] it is proposed to first express the CPC by its KKT constraints. Using SOS-1 constraints or a Big-M formulation, this can be achieved without leaving the class of mixed-integer linear programs.

5.8 Illustrative example

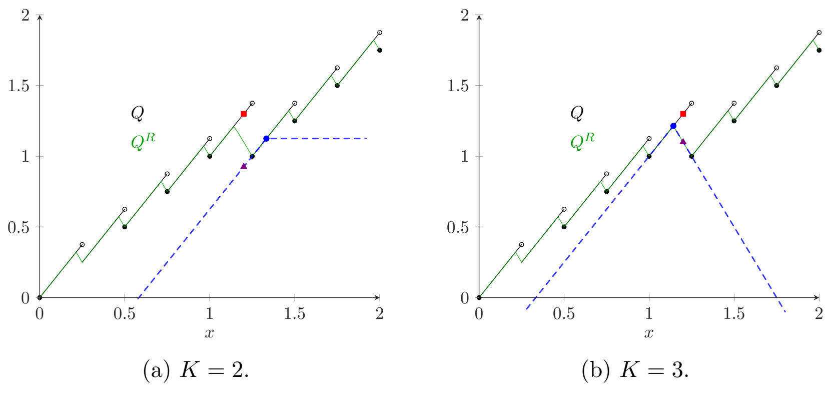
We highlight the key take-aways from this section with an illustrative example.

Example 5.14. *Consider the problem* (7) *again.*

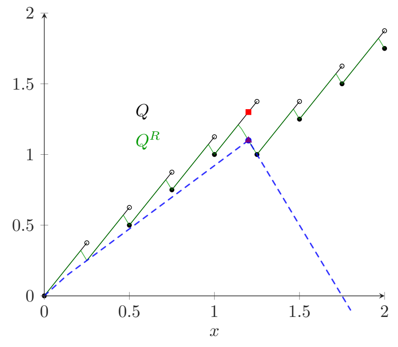
(1) ***CPC for fixed σ.*** *Let the incumbent be $x = \frac{6}{5}$. For the regularization, let $\|\cdot\|^\circ = |\cdot|$ and $\sigma = 2$. For the cut generation, we choose $K = 3$ ($\beta = \frac{2}{7}$) and $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ as proposed. According to Example 5.2, the anchor point becomes $x_{\mathcal{B}} = \frac{8}{7}$. By solving the Lagrangian dual problem* (30) *at that point, we obtain the CPC*

$$\phi(x) := \max_{\lambda} \Big\{ -\frac{1}{2} - \frac{4}{7}\lambda_1 - \frac{8}{7}\lambda_2 + \frac{12}{7}\lambda_3 \;:\; \frac{2}{7}(\lambda_1 + 2\lambda_2 + 4\lambda_3) = x,\; \lambda \in [0,1]^3 \Big\}.$$

We can apply the same procedure for $K = 2$ and $K = 4$. For all three cases, the CPC is visualized in Fig. 9. Its value at $x_{\mathcal{B}}$ is highlighted by a blue dot, the *value at $x = \frac{6}{5}$ by a violet triangle and the true value $Q(\frac{6}{5})$ by a red square. We see that in all three cases, the CPC is valid and the regularized value function $Q^R(\cdot;\sigma|\cdot|)$ is overestimated at $x_{\mathcal{B}}$. In fact, the overestimation is exact for the given example. Moreover, the anchor point $x_{\mathcal{B}}$ gets closer to $x = \frac{6}{5}$ with increasing the binary precision. For $K = 4$ both points coincide. This does not guarantee to monotonically improve the approximation at $x = \frac{6}{5}$, though, as Fig. 9b and Fig. 9c show.*



(a) $K = 2$. (b) $K = 3$.



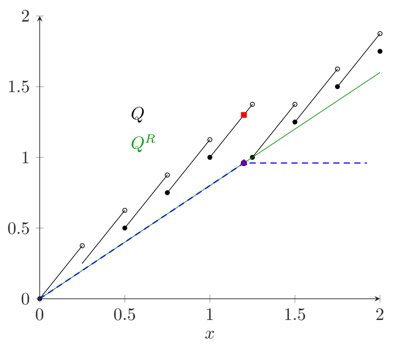
(c) $K = 4$.

Figure 9: CPC at $x = \frac{6}{5}$ for $\sigma = 2$, $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ and different K in Example 5.14.

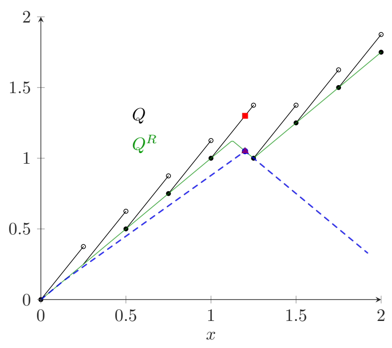
(2) ***CPC for increasing*** $\boldsymbol{\sigma}$***.*** *Let the incumbent be $x = \frac{6}{5}$ again. As for case (1), we choose $\|\cdot\|^\circ = |\cdot|$ and $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$. We fix the binary precision to $K = 4$ ($\beta = \frac{2}{15}$) and consider different values for σ. The obtained CPCs are visualized in Fig. 10. We observe that in each case, the CPC is valid and the corresponding regularized value function is (exactly) overestimated at $x = \frac{6}{5}$. Additionally, the slope of the CPC is bounded by σ. For increasing values of σ, the approximation of the true value function at $x = \frac{6}{5}$ is improved.*

(3) ***Using*** $\|\cdot\|^\bullet = \|\cdot\|_{\infty,w}$***.*** *Consider the same setting as for case (1), but with $\|\cdot\|^\bullet = \|\cdot\|_{\infty,w}$ instead of the weighted 1-norm. In this case, we obtain the CPC*

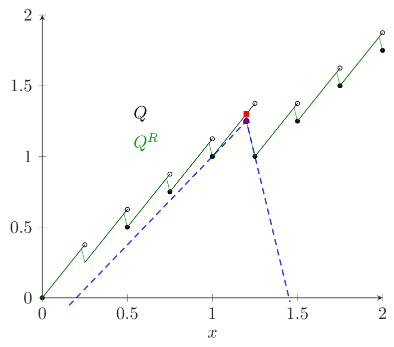
$$\phi(x) := \max_{\lambda} \left\{ -\frac{2}{7}\lambda_1 - \frac{4}{7}\lambda_2 + 1.10714\lambda_3 \ : \ \frac{2}{7}(\lambda_1 + 2\lambda_2 + 4\lambda_3) = x, \ \lambda \in [0,1]^3 \right\}.$$



(a) $\sigma = \frac{4}{5}$.



(b) $\sigma = 1$.



(c) $\sigma = 5$.

Figure 10: CPC at $x = \frac{6}{5}$ for $K = 4$, $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ and different values of σ in Example 5.14.

As Fig. 11a shows, in this case, the overestimation property is not satisfied.

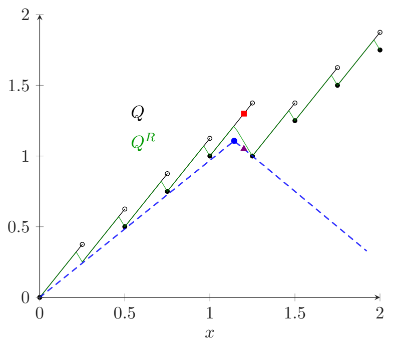
Setting $\sigma = 1$, we even observe a case in which $Q^R_{\mathcal{B}}(\cdot; \sigma\|\cdot\|_{\infty,w})$ and its closed convex envelope $\overline{\text{co}}(Q^R_{\mathcal{B}}; \sigma\|\cdot\|_{\infty,w})(\cdot)$ do not coincide at $\lambda = (0,0,1)$, which is the binary representation of the anchor point $x_{\mathcal{B}}$.

(4) ***Using $\mathfrak{z} \in \{0,1\}^K$.*** *Consider the same setting as for case (1), but with choosing $\mathfrak{z} \in \{0,1\}^K$ instead of $\mathfrak{z} \in [0,1]^K$. The CPC can be computed as*

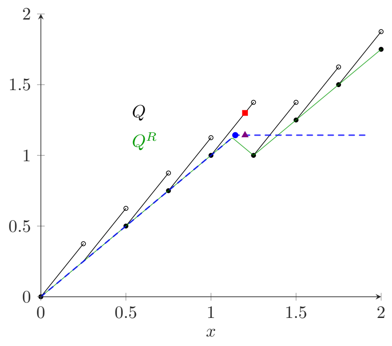
$$\phi(x) := \max_{\lambda} \left\{ 0.914286\lambda_3 \;:\; \frac{2}{7}(\lambda_1 + 2\lambda_2 + 4\lambda_3) = x,\; \lambda \in [0,1]^3 \right\}.$$

As Fig. 11b shows, this is not a valid cut for the value function $Q(\cdot)$. The CPC is only guaranteed to be valid for points which can be exactly represented in the lifted state space.

(5) ***Bounding the slope of the CPC.*** *Let $x = 1.249$ now, i.e., very close to a point of discontinuity of $Q(\cdot)$. Let $\|\cdot\|^\circ = |\cdot|$, $\sigma = 5$, $K = 8$ for a sufficiently close approximation, and $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$ as proposed. As Fig. 12a illustrates, in this case, we obtain a CPC that overestimates $Q^R(\cdot; \sigma|\cdot|)$, but is bounded in slope by $\sigma = 5$ (blue dotted line).*

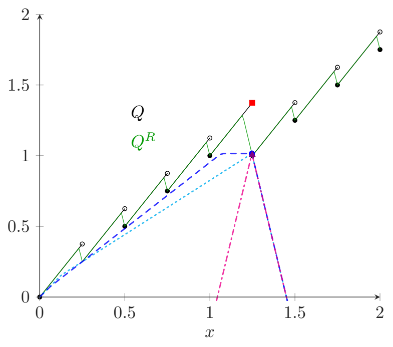


(a) $\sigma = 2$ and $\|\cdot\|^\bullet = \|\cdot\|_{\infty,w}$.

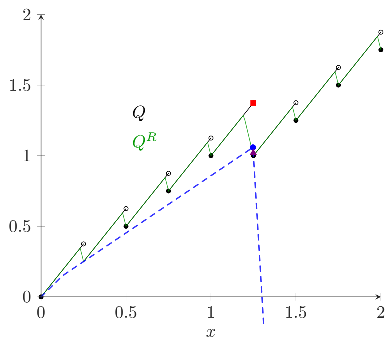


(b) $\sigma = 1$, $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$, $\mathfrak{z} \in \{0,1\}^K$.

Figure 11: Non-sufficient CPCs for $K = 3$ at $x = \frac{6}{5}$ in Example 5.14.



(a) Using $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$.



(b) Using $\|\cdot\|^\bullet = \|\cdot\|_1$.

Figure 12: CPC at $x = 1.249$ for $\sigma = 5$ and $K = 8$ in Example 5.14.

In contrast, consider the case where we do not use the weighted norm $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$, but the unweighted one $\|\cdot\|^\bullet = \|\cdot\|_1$ instead. As proven in [13] we can then bound the dual multipliers in (30) *by $\sigma \max_j U_j$ to achieve the intended overestimation. However, the CPC becomes extremely steep for $K = 8$ and is not bounded by $\sigma = 5$, see Fig. 12b. In general, its Lipschitz constant may diverge for $\beta \to 0$.*

For completeness, we should mention that Lagrangian dual problems are often degenerate with infinitely many optimal solutions. Therefore, given a trial point $x^i_{a(n)}$, there may exist an infinite number of tight linear Lagrangian cuts (satisfying Corollary 3.14) or tight CPCs (satisfying Theorem 5.13) with varying approximation quality at $x_{a(n)} \neq x^i_{a(n)}$. For illustration, see the blue dashed, cyan dotted, and magenta dash-dotted CPCs for problem (7) shown in Fig. 12a.

6 Conclusion

We provide new theoretical insight on the generation of linear and non-convex cuts for value functions of MS-MILPs based on Lagrangian duality, and the effects that copy constraints and a Lipschitz regularization of the subproblems have in this context. In particular, we point out the relation between bounded Lagrangian dual problems and the convex envelope of the regularized value functions. We further show that by a careful choice of the regularization, this relation can be exploited to generate non-convex cuts with favorable properties.

As future work directions, a computational comparison of generating linear Lagrangian cuts using non-regularized and regularized problems could be of interest. While the approximation quality is better in the first case, bounding the Lagrangian duals in the latter might accelerate the cut generation process. For non-convex approximations, the CPC from our lift-and-project approach could be compared in detail with the augmented Lagrangian cuts proposed in [1]. Another challenge is to efficiently incorporate the non-convex CPC into the subproblems within SDDP-like methods.

Acknowledgments

Andy Sun's research is partially funded by the National Science Foundation CAREER award 2316675. Christian Füllner's research is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)–445857709. The authors thank Filipe Cabral and Shixuan Zhang for the fruitful discussions on this topic during Christian Füllner's research visit at Georgia Institute of Technology. This visit was funded by the Karlsruhe House of Young Scientists (KHYS).

A Proof of Lemma 2.2

Proof. Consider a leaf node n of $\mathcal{N}$. Since $f_n(\cdot)$ is linear and $F_n(x_{a(n)})$ is bounded for all $x_{a(n)}$ by Assumption 1 (A1), we conclude that $Q_n(\cdot)$ is bounded from below. Moreover, by feasibility assumption (A3), we have $\operatorname{dom}(Q_n) \neq \emptyset$. Hence, $Q_n(\cdot)$ is proper. For all other nodes in $\mathcal{N}$, a similar reasoning can be applied inductively.

The lsc of $Q_n(\cdot)$ and the closedness of $\operatorname{dom}(Q_n)$ follow from [19, Theorem 3.1] under Assumption 1 (A2), based on the observation that, apart from the integrality requirements, X_n and Y_n are representable by polyhedral constraints. The piecewise polyhedrality follows from the mixed-integer linear character of the subproblems, see also [19].

By taking expectations the assertion also holds for $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$. □

B Proof of Lemma 2.3

Proof. Since $Q_n(\cdot)$ is bounded from below (see Appendix A), also $\overline{\operatorname{co}}(Q_n)(\cdot)$ is bounded from below (there exists a constant convex function underestimating $Q_n(\cdot)$). By Assumption 1 (A3), $\operatorname{dom}(Q_n) \neq \emptyset$. As $\overline{\operatorname{co}}(Q_n)(\cdot)$ underestimates $Q_n(\cdot)$ on $\operatorname{dom}(Q_n)$, this implies that $\overline{\operatorname{co}}(Q_n)(\cdot)$ is proper. Then the assertion follows from [4, Proposition 1.6.1]. □

C Proof of Lemma 3.7

Proof. Recall from Lemma 3.5 that the objective function of subproblem (8) is bounded from below on $\mathcal{M}_n^{i+1}$. The objective function is the same for (12), and by Lemma 3.4 also the recession cones of $\mathcal{M}_n^{i+1}$ and $\operatorname{conv}(\mathcal{M}_n^{i+1})$ do coincide. Therefore, the objective function is also bounded from below on $\operatorname{conv}(\mathcal{M}_n^{i+1})$.

The result $\operatorname{dom}(\underline{Q}_n^{C,i+1}) = \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$ follows by standard convexity arguments considering the constraint set $\operatorname{conv}(\mathcal{M}_n^{i+1})$ instead of $\mathcal{M}_n^{i+1}$. By Assumption 1 (A3), we have $\operatorname{dom}(\underline{Q}_n^{i+1}) \neq \emptyset$, and thus $\operatorname{dom}(\underline{Q}_n^{C,i+1}) \neq \emptyset$. Together with the boundedness from below, the properness follows.

Due to Lemma 3.4, problem (12) can be rewritten as a linear program, and therefore the finite minimum is attained on $\operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$. Moreover, for linear programs, the lower semicontinuity and convexity on $\mathbb{R}^{d_{a(n)}}$ (with $\underline{Q}_n^{C,i+1}(x_{a(n)}) = +\infty$ for all $x_{a(n)} \notin \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1})))$ and the piecewise linearity on $\operatorname{dom}(\underline{Q}_n^{C,i+1})$ follow from standard results in stochastic programming, see [6, 19] for instance. □

D Proof of Theorem 3.9

Proof. Applying the arguments from Lemma 2.3 to $\underline{Q}_n^{i+1}(\cdot)$, it follows that $(\underline{Q}_n^{i+1})^{**}(x_{a(n)}) = \overline{\operatorname{co}}(\underline{Q}_n^{i+1})(x_{a(n)})$ for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$, so it is sufficient for us to consider the biconjugate.

For any $\pi_n \in \mathbb{R}^{d_{a(n)}}$, the conjugate of $\underline{Q}_n^{i+1}(\cdot)$ is defined as

$$(\underline{Q}_n^{i+1})^*(\pi_n) = \max_{u_n} \big\{ \pi_n^\top u_n - \underline{Q}_n^{i+1}(u_n) \big\} = -\min_{u_n} \big\{ -\pi_n^\top u_n + \underline{Q}_n^{i+1}(u_n) \big\},$$

with $u_n, \pi_n \in \mathbb{R}^{d_{a(n)}}$ [4]. Then, the corresponding biconjugate is equal to $\underline{Q}_n^{D,i+1}(x_{a(n)}^i)$, since

$$(\underline{Q}_n^{i+1})^{**}(x_{a(n)}^i) = \max_{\pi_n} \left\{\pi_n^\top x_{a(n)}^i - (\underline{Q}_n^{i+1})^*(\pi_n)\right\} = \max_{\pi_n} \min_{u_n} \left\{\pi_n^\top (x_{a(n)}^i - u_n) + \underline{Q}_n^{i+1}(u_n)\right\}.$$

Inserting the definition of $\underline{Q}_n^{i+1}(\cdot)$ and utilizing the copy constraint to replace u_n with z_n, we obtain the Lagrangian dual (11). In particular, this holds true for the case where finite values are obtained, proving the assertion. □

E Proof of Theorem 3.13

Proof. First, we notice that $\mathrm{dom}(\underline{Q}_n^{i+1}) \subseteq Z_{a(n)}$ and bounded by assumption. By Lemma 3.5 it is also closed, thus compact. Under this condition, according to Remark 2.4, $\overline{\mathrm{co}}(\underline{Q}_n^{i+1})(\cdot)$ and $\mathrm{co}(\underline{Q}_n^{i+1})(\cdot)$ do coincide on $\mathrm{dom}(\underline{Q}_n^{i+1})$. Therefore, it remains to be shown that

$$\mathrm{co}(\underline{Q}_n^{i+1})(x_{a(n)}) = \underline{Q}_n^{i+1}(x_{a(n)})$$

for all $x_{a(n)} \in X_{a(n)} \cap \mathrm{dom}(\underline{Q}_n^{i+1})$.

Now choose $x_{a(n)}$ arbitrarily from this set. Since $x_{a(n)} \in X_{a(n)}$, by assumption $x_{a(n)}$ is an extreme point of $Z_{a(n)}$, meaning it cannot be expressed as a convex combination of points in $Z_{a(n)}$ different from itself [26]. However, we also have $x_{a(n)} \in \mathrm{dom}(\underline{Q}_n^{i+1}) \subseteq Z_{a(n)}$, so it cannot be expressed as a convex combination of points in $\mathrm{dom}(\underline{Q}_n^{i+1})$ different from itself either. Therefore, $x_{a(n)}$ is an extreme point of $\mathrm{dom}(\underline{Q}_n^{i+1})$. According to [26, Proposition 2.1], this implies that $\mathrm{co}(\underline{Q}_n^{i+1})(x_{a(n)}) = \underline{Q}_n^{i+1}(x_{a(n)})$ for all $x_{a(n)} \in X_{a(n)}$. □

F Proof of Lemma 4.4

Proof. We prove all results for leaf nodes $n \in \mathcal{N}$. For other nodes, the same reasoning can be applied inductively, using that $\mathcal{Q}_{\mathcal{C}(n)}^R(\cdot; \sigma_{\mathcal{C}(n)} \|\cdot\|)$ is Lipschitz and underestimating $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$.

First, we show that the minimum in subproblem (13) is well-defined. The feasible set is

$$\mathcal{M}_n^{R;i+1} := \Big\{(x_n, y_n, z_n) \ : \ z_n \in Z_{a(n)}, x_n \in X_n, y_n \in Y_n, A_n z_n + B_n x_n + C_n y_n \geq b_n\Big\}.$$

Under Assumptions 1 and 2, this set is closed as the intersection of closed sets.

By definition, $\mathcal{Q}_{\mathcal{C}(n)}^R(\cdot; \sigma_{\mathcal{C}(n)} \|\cdot\|) \equiv 0$ is Lipschitz continuous. Therefore, the objective function $g_n(x_n, y_n, z_n) := f_n(x_n, y_n) + \sigma_n \|x_{a(n)} - z_n\| + \mathcal{Q}_{\mathcal{C}(n)}^R(x_n; \sigma_{\mathcal{C}(n)} \|\cdot\|)$ is lsc. Together with the closedness of $\mathcal{M}_n^{R;i+1}$, it follows that for any $\alpha \in \mathbb{R}$, the level set

$$\mathrm{lev}_\alpha(g_n) = \Big\{(x_n, y_n, z_n) \in \mathcal{M}_n^{R;i+1} \ : \ g_n(x_n, y_n, z_n) \leq \alpha\Big\}$$

is closed. Moreover, by Assumption 1, x_n and y_n are bounded, and by that $f_n(x_n, y_n) + \mathcal{Q}_{\mathcal{C}(n)}^R(x_n; \sigma_{\mathcal{C}(n)} \|\cdot\|)$ is bounded from below by some finite constant $\tilde{\alpha}$. The remaining term $\sigma_n \|x_{a(n)} - z_n\|$ is bounded from below by 0 and bounded from above by $\alpha - \tilde{\alpha}$. Therefore, within $\mathrm{lev}_\alpha(g_n)$, z_n is bounded as well. In total, $\mathrm{lev}_\alpha(g_n)$ is compact. For α sufficiently large, it is also non-empty. Then, by extensions of the Theorem of

Weierstraß, a finite minimum is attained in subproblem (13). This immediately implies properness of $Q_n^R(\cdot;\sigma_n\|\cdot\|)$.

Second, assuming that it exists, let (x_n^*, y_n^*, z_n^*) be an optimal solution to the original subproblem (5). Clearly, this point is feasible, but not necessarily optimal for subproblem (13). Due to the copy constraint in subproblem (5), the term $\sigma_n\|x_{a(n)} - z_n^*\|$ vanishes in the objective, and the underestimation property follows. In contrast, if subproblem (5) has no feasible solution given $x_{a(n)}$, then we have $Q_n(x_{a(n)}) = +\infty$ and the assertion is trivial.

Finally, we notice that $Q_n^R(\cdot;\sigma_n\|\cdot\|)$ can be interpreted as the Pasch-Hausdorff envelope (or Lipschitz regularization) of $Q_n(\cdot)$ and $\sigma_n\|\cdot\|$:

$$Q_n^R(x_{a(n)};\sigma_n\|\cdot\|) = Q_n\Box(\sigma_n\|\cdot\|)(x_{a(n)}) = \min_{z_n\in Z_{a(n)}} Q_n(z_n) + \sigma_n\|x_{a(n)} - z_n\|.$$

By a similar reasoning as above, a finite minimum is attained. Since $Q_n(\cdot)$ is proper and lsc according to Lemma 2.2 and since $Q_n^R(\cdot;\sigma_n\|\cdot\|)$ is proper as well, it follows that $Q_n^R(\cdot;\sigma_n\|\cdot\|)$ is σ_n-Lipschitz by a general property of the Pasch-Hausdorff envelope, see [2, Proposition 12.17]. □

G Proof of Lemma 4.8

Proof. (a) Problem (15) is a relaxation of problem (13). Hence, by Lemma 4.3, it is feasible for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$. Additionally, x_n, y_n are bounded, whereas $\theta_{\mathcal{C}(n)}$ and $\|\cdot\|$ are at least bounded from below. Therefore, $\underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|)$ is finite-valued, and by that also proper. Using the same reasoning as in Appendix F, we can also show that finite infima are attained.

The Lipschitz continuity can be shown by exploiting that $\underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|)$ is the Pasch-Hausdorff envelope (or Lipschitz regularization) of $\underline{Q}_n^{i+1}(\cdot)$ and $\sigma_n\|\cdot\|$ for all $n \in \mathcal{N}$. Since $\underline{Q}_n^{i+1}(\cdot)$ is proper and lsc according to Lemma 3.5 and since $\underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|)$ is proper as well, it follows that $\underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|)$ is σ_n-Lipschitz continuous by a general property of the Pasch-Hausdorff envelope, see [2, Proposition 12.17].

(b) The convexity can be shown in a straightforward way given that the feasible set and the objective are convex. As problem (16) is a relaxation of problem (15), feasibility and boundedness of $\underline{Q}_n^{CR;i+1}(\cdot;\sigma_n\|\cdot\|)$ from above follow from (a). Recall that x_n, y_n are bounded and $\theta_{\mathcal{C}(n)}$ is bounded from below in the set $\mathcal{M}_n^{i+1}$. By convexity properties, it can be shown that these properties also must hold for elements in $\text{conv}(\mathcal{M}_n^{i+1})$. Therefore, the objective function is bounded from below. It follows that $\underline{Q}_n^{CR;i+1}(\cdot;\sigma_n\|\cdot\|)$ is finite-valued, and by that also proper.

Using the same arguments as in (a) for $\underline{Q}_n^{CR;i+1}(\cdot;\sigma_n\|\cdot\|)$ and $\underline{Q}_n^{C,i+1}(\cdot)$ together with Lemma 3.7 we can conclude that $\underline{Q}_n^{CR;i+1}(\cdot;\sigma_n\|\cdot\|)$ is σ_n-Lipschitz continuous on $\mathbb{R}^{d_{a(n)}}$, and thus also closed.

(c) The function $\overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(\cdot)$ is convex by definition. As it underestimates $\underline{Q}_n^{R;i+1}(\cdot;\sigma_n\|\cdot\|)$, which is finite-valued, it is finite-valued as well.

(d) As shown in part (b), $\underline{Q}_n^{CR;i+1}(\cdot;\sigma_n\|\cdot\|)$ is closed proper convex. Therefore, by [4, Proposition 1.6.1 (c)] it coincides with its biconjugate $(\underline{Q}_n^{CR;i+1};\sigma_n\|\cdot\|)^{**}(\cdot)$ on $\mathbb{R}^{d_{a(n)}}$. □

H Proof of Lemma 4.9

Proof. We define c_n and λ_n as in (18). First, we consider problem (15), but introduce an additional variable and copy constraint to obtain

$$\underline{Q}_n^{R;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|) = \min_{\lambda_n,z_n,u_n} \left\{c_n^\top \lambda_n + \sigma_n\|u_n\| \;:\; (\lambda_n,z_n)\in\mathcal{M}_n^{i+1}, u_n = x_{a(n)}^i - z_n\right\}.$$

We relax the equality constraint, which yields the dual function

$$\begin{aligned}\Phi(\pi_n) &:= \min_{\lambda_n,z_n,u_n,w_n} \left\{c_n^\top\lambda_n + \sigma_n\|u_n\| + \pi_n^\top(u_n - (x_{a(n)}^i - z_n)) \;:\; (\lambda_n,z_n)\in\mathcal{M}_n^{i+1}\right\}\\ &= \min_{(\lambda_n,z_n)\in\mathcal{M}_n^{i+1}} c_n^\top\lambda_n - \pi_n^\top(x_{a(n)}^i - z_n) + \min_{u_n}\sigma_n\|u_n\| + \pi_n^\top u_n\\ &= \min_{(\lambda_n,z_n)\in\operatorname{conv}(\mathcal{M}_n^{i+1})} c_n^\top\lambda_n - \pi_n^\top(x_{a(n)}^i - z_n) + \min_{u_n}\sigma_n\|u_n\| + \pi_n^\top u_n.\end{aligned}$$

The first equation follows from separability, while the second one follows from the linearity of the objective in the first minimization problem.

Using the same steps for the convexified problem (16), we obtain the dual function $\Phi^C(\pi_n)$, which satisfies $\Phi(\pi_n) = \Phi^C(\pi_n)$ for all $\pi_n \in \mathbb{R}^{d_{a(n)}}$. As they are defined by taking the supremum of $\Phi(\pi_n)$ or $\Phi^C(\pi_n)$ over all π_n respectively, also the biconjugates $(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)^{**}(\cdot)$ and $(\underline{Q}_n^{CR;i+1};\sigma_n\|\cdot\|)^{**}(\cdot)$ are equivalent. □

I Proof of Lemma 4.16

Proof. We consider some arbitrary node $n \in \overline{\mathcal{N}}$. By Theorem 4.10 and Corollary 4.12 we have

$$\underline{Q}_n^{CR;i+1}(x_{a(n)};\sigma_n\|\cdot\|) = \overline{\operatorname{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(x_{a(n)}) \tag{40}$$

for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$. Analogously, by Theorem 3.8 and Theorem 3.9 for all $x_{a(n)} \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$ we have

$$\underline{Q}_n^{C,i+1}(x_{a(n)}) = \overline{\operatorname{co}}(\underline{Q}_n^{i+1})(x_{a(n)}). \tag{41}$$

We now show that $\underline{Q}_n^{CR;i+1}(x_{a(n)};\sigma_n\|\cdot\|) = \underline{Q}_n^{C,i+1}(x_{a(n)})$ is satisfied for all $x_{a(n)} \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$ to prove the assertion. For that, consider the definition of $\underline{Q}_n^{C,i+1}(\cdot)$ in (12). According to Lemma 3.7, $\underline{Q}_n^{C,i+1}(\cdot)$ is a piecewise linear convex function, which implies that it is Lipschitz continuous on $\operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$.

Finally, we notice that $\underline{Q}_n^{CR;i+1}(\cdot;\sigma_n\|\cdot\|)$ is exactly the regularized function that we obtain if we Lipschitz-regularize $\underline{Q}_n^{C,i+1}(\cdot)$ using parameter σ_n. Therefore, we may apply the reasoning from Lemma 4.15 to these two functions. We conclude that if σ_n is chosen sufficiently large, then

$$\underline{Q}_n^{C,i+1}(x_{a(n)}) = \underline{Q}_n^{CR;i+1}(x_{a(n)};\sigma_n\|\cdot\|) \tag{42}$$

for all $x_{a(n)} \in \operatorname{conv}(\operatorname{dom}(\underline{Q}_n^{i+1}))$. Combining this with (40) and (41) proves the assertion. □

J Proof of Lemma 4.18

Proof. For sufficiently large $\sigma_n > 0$, combining Corollary 4.12, Lemma 4.16 and the definition of the closed convex envelope yields

$$\underline{Q}_n^{DR;i+1}(x^i_{a(n)};\sigma_n\|\cdot\|) = \overline{\text{co}}(\underline{Q}_n^{R;i+1};\sigma_n\|\cdot\|)(x^i_{a(n)}) = \overline{\text{co}}(\underline{Q}_n^{i+1})(x^i_{a(n)}) \leq \underline{Q}_n^{R;i+1}(x^i_{a(n)};\sigma_n\|\cdot\|)$$

for all $x_{a(n)} \in \text{conv}(\text{dom}(\underline{Q}_n^{i+1}))$. Additionally, given the taken assumptions, by Theorem 3.13, we have $\overline{\text{co}}(\underline{Q}_n^{i+1})(x^i_{a(n)}) = \underline{Q}_n^{i+1}(x^i_{a(n)})$ for all $x_{a(n)} \in X_{a(n)} \cap \text{dom}(\underline{Q}_n^{i+1})$. Hence, the first equality in the assertion follows for $x^i_{a(n)} \in X_{a(n)} \cap \text{dom}(\underline{Q}_n^{i+1})$. This directly implies

$$\underline{Q}_n^{i+1}(x^i_{a(n)}) \leq \underline{Q}_n^{R;i+1}(x^i_{a(n)};\sigma_n\|\cdot\|).$$

However, since we also have the opposite result for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$ by definition (cf. Lemma 4.4), the second equality in the assertion follows. $\square$

K Proof of Lemma 4.19

Proof. We construct a special feasible solution for the dual problem (17) to prove the assertion. Suppose $\widehat{\pi}_n \in \mathbb{R}^{d_{a(n)}}$ is a vector, for which each component j is defined by

$$\widehat{\pi}_{nj} := \begin{cases} \sigma_n & \text{if } x^i_{a(n),j} = 1 \\ -\sigma_n & \text{if } x^i_{a(n),j} = 0. \end{cases} \tag{43}$$

Such construction is always possible, since $x^i_{a(n)} \in \{0,1\}^{d_{a(n)}}$. The vector $\widehat{\pi}_n$ is feasible for (17) with ℓ^∞-norm. Therefore, we obtain

$$\begin{aligned} \underline{Q}_n^{DR;i+1}(x^i_{a(n)};\sigma_n\|\cdot\|_1) &= \max_{\|\pi_n\|_*\leq\sigma_n} \min_{z_n\in Z_{a(n)}} \underline{Q}_n^{i+1}(z_n) + \pi_n^\top(x^i_{a(n)} - z_n) \\ &\geq \min_{z_n\in Z_{a(n)}} \underline{Q}_n^{i+1}(z_n) + \widehat{\pi}_n^\top(x^i_{a(n)} - z_n) \\ &= \min_{z_n\in Z_{a(n)}} \underline{Q}_n^{i+1}(z_n) + \sum_{j=1}^{d_{a(n)}} \widehat{\pi}_{tj}(x^i_{a(n),j} - z_{nj}). \end{aligned} \tag{44}$$

Now we exploit the binary nature of $x_{a(n)}$. Note that if $x^i_{a(n),j} = 1$, from (43) it follows

$$\widehat{\pi}_{tj}(x^i_{a(n),j} - z_{nj}) = \sigma_n(x^i_{a(n),j} - z_{nj}) = \sigma_n|x^i_{a(n),j} - z_{nj}|.$$

The last equality holds, since for $x^i_{a(n),j} = 1$ and $z_{nj} \in [0,1]$ (or $z_{nj} \in \{0,1\}$), the term $x^i_{a(n),j} - z_{nj}$ is always non-negative. Analogously, for $x^i_{a(n),j} = 0$, from (43) it follows

$$\widehat{\pi}_{tj}(x^i_{a(n),j} - z_{nj}) = -\sigma_n(x^i_{a(n),j} - z_{nj}) = \sigma_n|x^i_{a(n),j} - z_{nj}|.$$

Inserting this result in (44) yields

$$\begin{aligned}\underline{Q}_n^{DR;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|_1) &\geq \min_{z_n\in Z_{a(n)}} \underline{Q}_n^{i+1}(z_n) + \sum_{j=1}^{d_{a(n)}} \sigma_n|x_{a(n),j}^i - z_{nj}| \\ &= \underline{Q}_n^{R;i+1}(x_{a(n)}^i;\sigma_n\|\cdot\|_1).\end{aligned} \tag{45}$$

□

L Proof of Lemma 5.11

Proof. First we notice that the dual CPC problem (38) is separable in the dimensions $j = 1,\dots,d_n$ of the original state space. Hence, we may analyze each case separately, which yields

$$\min_{\eta_{nj},\mu_{nj}} \left\{x_{nj}\eta_{nj} + e^\top \mu_{nj} \;:\; \eta_{nj}\mathfrak{b}_{nj} + \mu_{nj} \geq \pi_{\mathcal{C}(n),j}, \mu_{nj} \geq 0\right\}.$$

$\mathfrak{b}_{nj} \in \mathbb{R}^{K_{nj}}$ is a vector which contains the non-zero entries from the j-th column of $\mathcal{B}_n^\top$. The variable η_{nj} is one-dimensional, and $\pi_{\mathcal{C}(n),j} \in \mathbb{R}^{K_{nj}}$ contains all entries from $\pi_{\mathcal{C}(n)}$ referring to component j.

We introduce slack variables and split up η_{nj} to reformulate the constraints as:

$$\begin{aligned}\eta_{nj}^+\mathfrak{b}_{nj} - \eta_{nj}^-\mathfrak{b}_{nj} + \mu_{nj} - \nu_{nj} &= \pi_{\mathcal{C}(n),j} \\ \eta_{nj}^+, \eta_{nj}^-, \mu_{nj}, \nu_{nj} &\geq 0.\end{aligned} \tag{46}$$

The set defined by (46) has $2+2K_{nj}$ variables. In a basic solution, $2+K_{nj}$ variables have to be zero and the K_{nj} columns associated with the remaining variables have to be linearly independent. We observe that for each row $k = 1,\dots,K_{nj}$, the variables μ_{njk} and ν_{njk} cannot be in the basis together, because otherwise the basic columns are not linearly independent. With the same reasoning, η_{nj}^+ and η_{nj}^- cannot be in the basis together. Moreover, for $K_{nj} > 1$, it is not sufficient to have only η_{nj}^+ or η_{nj}^- in the basis.

We now consider different cases of basic solutions.

Case 1. η_{nj}^+ and η_{nj}^- both in the basis. Then $\eta_{nj}^+ = \eta_{nj}^- = 0$, so obviously $\eta_{nj} = \eta_{nj}^+ - \eta_{nj}^- = 0 \leq \sigma_n^{\max}$.

Case 2. η_{nj}^+ in the basis. This implies $\eta_{nj}^- = 0$. The equation

$$\eta_{nj}^+ = \frac{\pi_{\mathcal{C}(n),jk} - \mu_{njk} + \nu_{njk}}{2^{k-1}\beta_{nj}}$$

has to be satisfied for all $k = 1,\dots,K_{nj}$ simultaneously. However, since η_{nj}^+ is in the basis, for some $\bar{k}$, both $\mu_{nj\bar{k}}$ and $\nu_{nj\bar{k}}$ have to be zero. Therefore, this $\bar{k}$ determines the value of η_{nj}^+. The largest possible value that $\eta_{nj}^+ \geq 0$ may take can be obtained by maximizing over k:

$$\eta_{nj}^+ \leq \max_{k=1,\dots,K_{nj}} \max\left\{\frac{\pi_{\mathcal{C}(n),jk}}{2^{k-1}\beta_{nj}}, 0\right\}.$$

If all $\pi_{\mathcal{C}(n),jk} \leq 0$, then $\eta^+_{nj} = 0 \leq \sigma^{\max}_n$. Otherwise,

$$\eta^+_{nj} \leq \max_{k=1,\dots,K_{nj}} \frac{\pi_{\mathcal{C}(n),jk}}{2^{k-1}\beta_{nj}}. \tag{47}$$

We now exploit the bounds in the Lagrangian dual problem (30). We choose $\|\cdot\|^\bullet = \|\cdot\|_{1,w}$, hence for each $m \in \mathcal{C}(n)$, the dual multipliers are bounded by $\|\pi_m\|_{\infty,w} \leq \sigma_m$. Recall that by our choice of the weight matrix W, this is equivalent to $|\pi_{mk}| \leq \sigma_m 2^{\kappa(k)-1}\beta_{n,j(k)}$ for all $k = 1,\dots,K_n$. Restricting to some component j, we have $|\pi_{mjk}| \leq \sigma_m 2^{k-1}\beta_{nj}$ for all $k = 1,\dots,K_{nj}$, and thus $|\pi_{\mathcal{C}(n),jk}| \leq \sigma^{\max}_n 2^{k-1}\beta_{nj}$ for all $k = 1,\dots,K_{nj}$. Consequently, in (47) it follows $\eta^+_{nj} \leq \sigma^{\max}_n$.

Case 3. η^-_{nj} in the basis. We can prove $\eta^-_{nj} \leq \sigma^{\max}_n$ by using the same reasoning as for Case 2.

Since $\eta_{nj} = \eta^+_{nj} - \eta^-_{nj}$, but only one of both variables can be non-zero, we conclude $\eta_{nj} \leq \sigma^{\max}_n$.

Due to separability, the above reasoning can be applied for each $j = 1,\dots,d_n$ separately, so it follows $\eta_{nj} \leq \sigma^{\max}_n$ for all j. Hence, $\|\eta_n\|_\infty \leq \sigma^{\max}_n$. Note that the above reasoning is completely independent of the values of β_{nj} or K_{nj}. □

M Proof of Lemma 5.12

Proof. The CPC is defined as the minimum of finitely many linear functions, which we enumerate by $\ell = 1,\dots,L$. Each such function $\psi^\ell_{\mathcal{C}(n)}(\cdot)$ is determined by an extreme point $(\mu^\ell_n, \nu^\ell_n, \eta^\ell_n)$ of (38). Consider two arbitrary points $x^1_n, x^2_n \in \mathbb{R}_{d_n}$. Using the Hölder inequality and Lemma 5.11, we obtain

$$|\psi^\ell_{\mathcal{C}(n)}(x^2_n) - \psi^\ell_{\mathcal{C}(n)}(x^1_n)| = |(\eta^\ell_n)^\top (x^2_n - x^1_n)| \leq \|\eta^\ell_n\|_\infty \|x^2_n - x^1_n\|_1 \leq \sigma^{\max}_n \|x^2_n - x^1_n\|_1.$$

Hence, each $\psi^\ell_{\mathcal{C}(n)}(\cdot)$ is Lipschitz continuous w.r.t. $\|\cdot\|_1$ with Lipschitz constant $\sigma^{\max}_n$. Taking the maximum of all Lipschitz constants over $\ell = 1,\dots,L$, we obtain a Lipschitz constant for the CPC, which is again $\sigma^{\max}_n$. By equivalence of norms in $\mathbb{R}_{d_n}$, for any other norm than $\|\cdot\|_1$, we can obtain a Lipschitz constant $\tilde{\sigma}_{\mathcal{C}(n)} > 0$ by multiplying $\sigma^{\max}_n$ with an appropriate positive constant. □

N Proof of Theorem 5.13

Proof. We first prove validity property (S1). From Corollary 5.7 we have $\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \phi_{\mathcal{B};\mathcal{C}(n)}(\lambda_n)$ for all $x_n \in \mathbb{R}^{d_n}$ and any $\lambda_n \in [0,1]^{K_n}$, such that $x_n = \mathcal{B}_n\lambda_n$. Hence

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \max_{\lambda_n} \Big\{ \phi_{\mathcal{B};\mathcal{C}(n)}(\lambda_n) \ : \ \lambda_n \in [0,1]^{K_n}, \mathcal{B}_n\lambda_n = x_n \Big\} = \phi_{\mathcal{C}(n)}(x_n)$$

for all $x_n \in \mathbb{R}^{d_n}$, where the last equality applies the definition of the CPC in (37). Since this result is true for all $\phi^r_{\mathcal{C}(n)}(\cdot), r = 1,\dots,i+1$, it also holds for their pointwise maximum $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$.

Next, we prove the overestimation property (S2). Note that $\phi^i_{\mathcal{B};\mathcal{C}(n)}(\lambda^i_n) = \phi^i_{\mathcal{C}(n)}(x^i_{\mathcal{B},n})$ by the definitions of $x^i_{\mathcal{B},n}$ and the CPC. Hence, under our assumptions, Corollary 5.8

yields

$$\phi^i_{\mathcal{C}(n)}(x^i_{\mathcal{B},n}) \geq \underline{\mathcal{Q}}^{R;i+1}_{\mathcal{C}(n)}(x^i_{\mathcal{B},n}; \sigma_{\mathcal{C}(n)} \|\cdot\|^\circ).$$

By definition of $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ in (39) it directly follows

$$\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(x^i_{\mathcal{B},n}) \geq \underline{\mathcal{Q}}^{R;i+1}_{\mathcal{C}(n)}(x^i_{\mathcal{B},n}; \sigma_{\mathcal{C}(n)} \|\cdot\|^\circ).$$

Finally, we prove Lipschitz property (S3) using Lemma 5.12. Under our assumptions, for any $\|\cdot\|^\circ$, each $\phi^r_{\mathcal{C}(n)}(\cdot), r = 1, \dots, i+1$, is $\tilde{\sigma}_{\mathcal{C}(n)}$-Lipschitz continuous with $\tilde{\sigma}_{\mathcal{C}(n)} > 0$, finite and independent of $\beta_{n,j} \in (0,1), j = 1, \dots, d_n$. The pointwise maximum of Lipschitz continuous functions is Lipschitz continuous with its Lipschitz constant the maximum of the individual constants. Therefore, $\mathfrak{Q}^{i+1}_{\mathcal{C}(n)}(\cdot)$ is $\tilde{\sigma}_{\mathcal{C}(n)}$-Lipschitz continuous w.r.t. $\|\cdot\|^\circ$. □

References

[1] S. Ahmed, F. G. Cabral, and B. Freitas Paulo da Costa. Stochastic Lipschitz dynamic programming. *Mathematical Programming*, 191:755–793, 2022.

[2] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces.* Springer, 2nd edition, 2017.

[3] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[4] D. P. Bertsekas. *Convex Optimization Theory.* Athena Scientific, 2009.

[5] J. R. Birge. Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, 33(5):989–1007, 1985.

[6] J. R. Birge and F. Louveaux. *Introduction to stochastic programming.* Springer Series in Operations Research and Financial Engineering. Springer, New York, 2nd edition, 2011.

[7] C.C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters*, 24(1):37–45, 1999.

[8] S. Cerisola, J. M. Latorre, and A. Ramos. Stochastic dual dynamic programming applied to nonconvex hydrothermal models. *European Journal of Operational Research*, 218(3): 687–697, 2012.

[9] R. Chen and J. Luedtke. On generating Lagrangian cuts for two-stage stochastic integer programs. *INFORMS Journal on Computing*, 34(4):2332–2349, 2022.

[10] M. Conforti, G. Cornuéjols, and G. Zambelli. *Polyhedral approaches to mixed integer linear programming*, pages 343–385. 50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art. Springer Berlin Heidelberg, 2010.

[11] J. E. Falk. Lagrange multipliers and nonconvex programs. *SIAM Journal on Control*, 7(4): 534–545, 1969.

[12] M.J. Feizollahi, S. Ahmed, and A. Sun. Exact augmented Lagrangian duality for mixed integer linear programming. *Mathematical Programming*, 161(1-2):365–387, 2017.

[13] C. Füllner and S. Rebennack. Non-convex nested Benders decomposition. *Mathematical Programming*, 196:987–1024, 2022.

[14] C. Füllner and S. Rebennack. Stochastic dual dynamic programming and its variants - a review. Preprint, available at `http://www.optimization-online.org/DB_FILE/2021/01/8217.pdf`, 2023.

[15] A. M. Geoffrion. Lagrangean relaxation for integer programming. In M.L. Balinski, editor, *Approaches to integer programming*, pages 82–114. Springer Berlin Heidelberg, Berlin, Heidelberg, 1974.

[16] F. Glover. Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):455–460, 1975.

[17] V. Leclère, P. Carpentier, J.-P. Chancelier, A. Lenoir, and F. Pacaud. Exact converging bounds for stochastic dual dynamic programming via Fenchel duality. *SIAM Journal on Optimization*, 30(2):1223–1250, 2020.

[18] R. R. Meyer. On the existence of optimal solutions to integer and mixed-integer programming problems. *Mathematical Programming*, 7:223–235, 1974.

[19] R. R. Meyer. Integer and mixed-integer programming models: general properties. *Journal of Optimization Theory and Applications*, 3(4), 1975.

[20] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375, 1991.

[21] A. B. Philpott, F. Wahid, and F. Bonnans. MIDAS: A mixed integer dynamic approximation scheme. *Mathematical Programming*, 181:19–50, 2020.

[22] R. Rahmaniani, S. Ahmed, T. G. Crainic, M. Gendreau, and W. Rei. The Benders dual decomposition method. *Operations Research*, 68(3):878–895, 2020.

[23] R. T. Rockafellar. *Convex analysis.* Princeton university press, 1970.

[24] G. Steeger and S. Rebennack. Dynamic convexification within nested Benders decomposition using Lagrangian relaxation: an application to the strategic bidding problem. *European Journal of Operational Research*, 257(2):669–686, 2017.

[25] G. Steeger, T. Lohmann, and S. Rebennack. Strategic bidding for a price-maker hydroelectric producer: stochastic dual dynamic programming and Lagrangian relaxation. *IISE Transactions*, 47:1–14, 2018.

[26] F. Tardella. On the existence of polyhedral convex envelopes. In C. A. Floudas and P. M. Pardalos, editors, *Frontiers in global optimization*, pages 563–573. Springer, 2004.

[27] N. van der Laan and W. Romeijnders. A converging Benders' decomposition algorithm for two-stage mixed-integer recourse models. *Operations Research*, 2023.

[28] R. M. van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

[29] S. Zhang and X. A. Sun. Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 196:935–985, 2022.

[30] S. Zhang and X. A. Sun. On distributionally robust multistage convex optimization: new algorithms and complexity analysis. available at `https://arxiv.org/pdf/2010.06759.pdf`, 2022.

[31] J. Zou, S. Ahmed, and X. A. Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.

C.1 Supplementary material

We provide some supplementary material that was part of our research on regularization and Lagrangian cuts, but omitted in the submitted publication for reasons of space.

Supplementary Material “On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming”

Christian Füllner, X. Andy Sun, Steffen Rebennack

1 The role of copy constraints – Two-dimensional example

Example 1.1. *Consider the value function*

$$\begin{aligned} Q(x) = \min_{y,z} \quad & -16y_1 - 19y_2 - 23y_3 - 28y_4 \\ s.t. \quad & 2y_1 + 3y_2 + 4y_3 + 5y_4 \leq 10 - \frac{1}{3}x_1 - \frac{2}{3}x_2 \\ & 6y_1 + y_2 + 3y_3 + 2y_4 \leq 10 - \frac{2}{3}x_1 - \frac{1}{3}x_2 \\ & y_1, y_2, y_3, y_4 \in \{0,1\} \end{aligned}$$

taken from [2], together with the two-dimensional state space

$$X = \Big\{(x_1,x_2) \in \mathbb{Z}^2 \ : \ x_1, x_2 \in [0,5], \ x_2 \leq \frac{9}{2} - x_1\Big\}.$$

To generate Lagrangian cuts, we introduce copy constraints $z = x$ together with constraint $z \in Z$. We analyze the effect of different choices for Z on the cut tightness. We consider $X, \mathbb{R}^2$ and the following sets in between as choices for Z:

$$\begin{aligned} \operatorname{conv}(X) &= \Big\{(x_1,x_2) \in \mathbb{R}^2 \ : \ x_1, x_2 \in [0,5], \ x_2 \leq 4 - x_1\Big\}, \\ \bar{X} &= \Big\{(x_1,x_2) \in \mathbb{R}^2 \ : \ x_1, x_2 \in [0,5], \ x_2 \leq \frac{9}{2} - x_1\Big\}, \\ \tilde{X} &= \Big\{(x_1,x_2) \in \mathbb{R}^2 \ : \ x_2 \leq \frac{9}{2} - x_1\Big\}, \\ X' &= \Big\{(x_1,x_2) \in \mathbb{Z}^2 \ : \ x_1, x_2 \in [0,5]\Big\}, \\ \operatorname{conv}(X') = \bar{X}' &= \Big\{(x_1,x_2) \in \mathbb{R}^2 \ : \ x_1, x_2 \in [0,5]\Big\}. \end{aligned}$$

These sets are illustrated in Figure 1.

We generate cuts at three different incumbents: $\bar{x} = (0,4)^\top, \bar{x} = (3,1)^\top, \bar{x} = (2,1)^\top$ which are highlighted with red circles. Depending on the choice of Z, these incumbents may be located at an extreme point of Z (thus satisfying the sufficient condition for tightness in Theorem 3.13), on the boundary of Z, or in the interior of Z (if Z is discrete, we define $\operatorname{ext}(Z) := \operatorname{ext}(\operatorname{conv}(Z))$ and $\operatorname{bd}(Z), \operatorname{int}(Z)$ analogously). These relations and the values of the obtained Lagrangian cuts at $\bar{x}$ are summarized in Table 1.

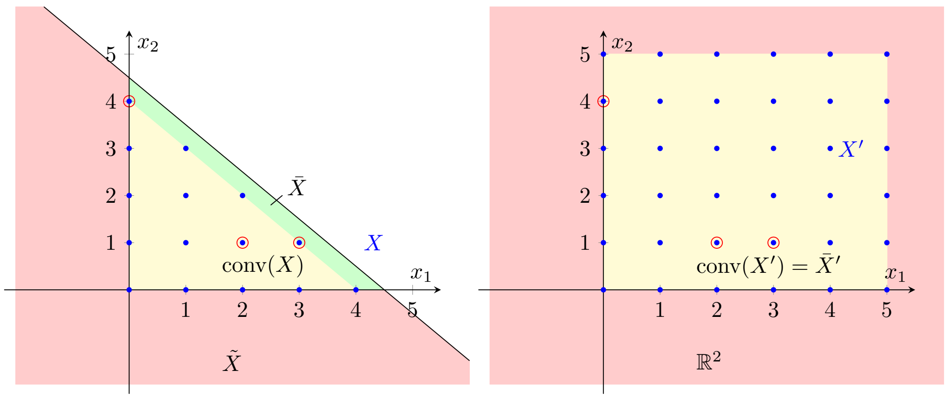


Figure 1: Different choices for Z in Example 1.1.

Table 1: Cut value at $\bar{x}$ for different choices of Z in Example 1.1.

	$\bar{x} = (0,4)^\top$ $Q(\bar{x}) =$ **-44.0** rel. to Z	cut value	$\bar{x} = (3,1)^\top$ $Q(\bar{x}) =$ **-47.0** rel. to Z	cut value	$\bar{x} = (2,1)^\top$ $Q(\bar{x}) =$ **-47.0** rel. to Z	cut value
X	$\in \text{ext}(Z)$	**-44.0**	$\in \text{bd}(Z)$	**-47.0**	$\in \text{int}(Z)$	-51.0
$\text{conv}(X)$	$\in \text{ext}(Z)$	**-44.0**	$\in \text{bd}(Z)$	**-47.0**	$\in \text{int}(Z)$	-51.0
$\bar{X}$	$\in \text{bd}(Z)$	-46.1	$\in \text{int}(Z)$	-48.8	$\in \text{int}(Z)$	-52.3
$\tilde{X}$	$\in \text{int}(Z)$	-46.3	$\in \text{int}(Z)$	-50.4	$\in \text{int}(Z)$	-53.8
X'	$\in \text{bd}(Z)$	**-44.0**	$\in \text{bd}(Z)$	-50.9	$\in \text{int}(Z)$	-53.8
$\text{conv}(X') = \bar{X}'$	$\in \text{bd}(Z)$	-46.1	$\in \text{bd}(Z)$	-51.1	$\in \text{int}(Z)$	-53.8
$\mathbb{R}^2$	$\in \text{int}(Z)$	-48.4	$\in \text{int}(Z)$	-53.1	$\in \text{int}(Z)$	-55.4

We can see that $\bar{x} \in \text{ext}(Z)$ *guarantees tight cuts with respect to* $Q(\cdot)$*, whereas for* $\bar{x} \in \text{int}(Z)$ *tightness is never achieved. For* $\bar{x} \in \text{bd}(Z)$*, tightness is achieved sometimes, however, not guaranteed in general. This observation is in accordance with the sufficient condition provided in Theorem 3.13.*

In particular, we notice that computing cuts as tight as possible requires not only to consider box and integrality constraints, but also the additional linear constraint $x_2 \leq \frac{9}{2} - x_1$ *defining* X*, when imposing constraints on the copy variable* z*.* □

2 CPC and augmented Lagrangian cuts

We compare the non-convex Lagrangian cuts derived in Sect. 5, referred to as CPCs, with the non-convex augmented Lagrangian cuts proposed in [1] and enhanced in [4].

2.1 Augmented Lagrangian cuts

To compute augmented Lagrangian cuts, given some some norm $\|\cdot\|$ and some multipliers $\pi_n \in \mathbb{R}^{d_{a(n)}}$ and $\rho_n \geq 0$, we consider the inner problem

$$\begin{aligned} \mathcal{AL}_n^{i+1}(\pi_n, \rho_n; x_{a(n)}^i) := \min_{x_n, y_n, z_n, \theta_{\mathcal{C}_n}} \quad & f_n(x_n, y_n) + \theta_{\mathcal{C}(n)} - \pi_n^\top (z_n - x_{a(n)}^i) \\ & + \rho_n \|z_n - x_{a(n)}^i\| \\ \text{s.t.} \quad & (x_n, y_n, z_n, \theta_{\mathcal{C}(n)}) \in \mathcal{M}_n^{i+1}, \end{aligned} \tag{48}$$

which defines the augmented dual function $\mathcal{AL}_n^{i+1}(\cdot)$. Apart from some constant in the objective, this problem differs from the inner problem (10) to the classical Lagrangian dual by the penalization term $\rho_n\|z_n - x_{a(n)}^i\|$ in the objective. Note that this penalization term resembles the one that we use for regularization in Sect. 4.

Similar to the classical outer problem (11), the augmented dual problem optimizes $\mathcal{AL}_n^{i+1}(\cdot)$ over the multipliers π_n and ρ_n.

$$\begin{aligned} \gamma_n^{AD,i+1} := \underline{Q}_n^{AD,i+1}(x_{a(n)}^i) := \max_{\pi_n, \rho_n} \quad & \mathcal{AL}_n^{i+1}(\pi_n, \rho_n; x_{a(n)}^i) \\ \text{s.t.} \quad & \rho_n \geq 0. \end{aligned} \tag{49}$$

For some optimal solution (π_n^i, ρ_n^i) of problem (49) with optimal value $\gamma_n^{AD,i+1}$, an augmented Lagrangian cut (ALD cut) can be defined by function

$$\phi_n^{AD}(x_{a(n)}) := \gamma_n^{AD,i+1} + (\pi_n^i)^\top (z_n - x_{a(n)}^i) - \rho_n \|z_n - x_{a(n)}^i\|.$$

Given some $n \in \tilde{N}$ and some trial point x_n^i, by taking expectations of $\gamma_m^{AD,i+1}, \pi_m^i$ and ρ_m^i over all $m \in \mathcal{C}(n)$, we obtain coefficients $\gamma_{\mathcal{C}(n)}^{AD,i+1}, \pi_{\mathcal{C}(n)}^i$ and $\rho_{\mathcal{C}(n)}^i$ and can define the function

$$\phi_{\mathcal{C}(n)}^{AD}(x_n) := \gamma_{\mathcal{C}(n)}^{AD,i+1} + (\pi_{\mathcal{C}(n)}^i)^\top (x_n - x_n^i) - \rho_n \|x_n - x_n^i\|, \tag{50}$$

which defines an aggregated augmented Lagrangian cut. This cut is valid, *i.e.*,

$$\mathcal{Q}_{\mathcal{C}(n)}(x_n) \geq \phi_{\mathcal{C}(n)}^{AD}(x_n)$$

for all $x_n \in X_n$ (this is true even for suboptimal multipliers π_m, ρ_m, $m \in \mathcal{C}(n)$), and tight [1], *i.e.*,

$$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(x_n^i) := \sum_{m \in \mathcal{C}(n)} p_{nm} \underline{Q}_m^{i+1}(x_n^i) = \phi_{\mathcal{C}(n)}^{AD}(x_n^i).$$

2.2 Comparison of CPC and ALD cuts

We summarize some important similarities and differences between the computation and application of CPCs and ALD cuts.

In both cases, Lipschitz continuity and tightness of the cuts are required for convergence. This in turn, requires to compute cuts for Lipschitz continuous value functions, as otherwise arbitrarily steep cuts might be generated close to discontinuities or infinitely steep sections. Lipschitz continuity can be guaranteed by Lipschitz regularization, see Sect. 4, or a complete continuous recourse (CCR) assumption [5]. In Sect. 5, for the CPC, we focus on Lipschitz regularization, while for the ALD cuts in [1] CCR is assumed. Reverse choices are possible, though, see for instance Sect. 4.5. Therefore, we consider both cases for both approaches.

2.2.1 Case 1: Lipschitz continuous value functions

Consider the case of Lipschitz continuous value functions. Let α_n denote the Lipschitz constant of $Q_n(\cdot)$. Then, according to Lemma 4.15, for any $n \in \bar{\mathcal{N}}$ and for $\sigma_n \geq \alpha_n$, the regularized value function $Q_n^R(\cdot; \sigma_n \|\cdot\|)$ and the value function $Q_n(\cdot)$ coincide on $\operatorname{dom}(Q_n)$. The same holds true for the approximate value functions $\underline{Q}_n^{R;i+1}(\cdot; \sigma_n \|\cdot\|)$ and $\underline{Q}_n^{i+1}(\cdot)$. Therefore, for any $n \in \mathcal{N}$, some norm $\|\cdot\|$ and some $\sigma_n \geq \alpha_n$, any sufficient CPC also satisfies the tightness condition

$$\mathfrak{Q}_{\mathcal{C}(n)}^{i+1}(x_{\mathcal{B},n}^i) = \underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(x_{\mathcal{B},n}^i)$$

at the anchor point $x_{\mathcal{B},n}^i$. This follows by combining properties (S1) and (S2) in Definition 5.1. Crucially, this tightness result requires a sufficiently large choice of σ_n. At best, the Lipschitz constant α_n is known, so that tightness is achieved while simultaneously avoiding overly steep cuts.

In comparison, for any $n \in \mathcal{N}$, some norm $\|\cdot\|$ and some sufficiently large $\rho_n \geq \alpha_n$, the ALD cut (50) is tight at $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(x_n^i)$ (it can be shown that $\rho_n = \alpha_n$ is sufficient for this result to hold [1]). Importantly, as no binarization is required, here tightness is achieved at x_n^i and not only at $x_{\mathcal{B}n}^i$.

2.2.2 Case 2: Non-Lipschitz continuous value functions

Suppose that the CCR property cannot be satisfied, so the value functions are not guaranteed to be Lipschitz continuous.

In our proposed approach, we apply a Lipschitz regularization. According to Definition 5.1, for any $n \in \mathcal{N}$, some given norm $\|\cdot\|^\circ$ and some $\sigma_n \geq \alpha_n$, any sufficient CPC overestimates $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(x_{\mathcal{B},n}^i; \sigma_n \|\cdot\|^\circ)$ and is Lipschitz continuous.

Non-bounded ALD cuts. For ALD cuts, things are a bit more intricate. If we follow the ALD approach from above, without Lipschitz regularization and dual bounds, the obtained ALD cuts may still become tight at $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(x_n^i)$ if we choose ρ_n sufficiently large, however, at the cost of being very steep at trial points x_n^i close to discontinuities or infinitely steep sections. The same is true if we apply a Lipschitz regularization in the forward pass, but do not impose appropriate dual bounds in the backward pass.

Generalized conjugacy cuts. However, if we additionally bound the dual variables by $\|\pi_n\|_* \leq \sigma_n, 0 \leq \rho_n \leq \sigma_n$, in the augmented dual problem, then the obtained ALD cuts are guaranteed to overestimate $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(x_n^i; \sigma_n \|\cdot\|^\circ)$ [4, Proposition 3], but are also guaranteed to be $2\sigma_{\mathcal{C}(n)}$-Lipschitz continuous [4, Proposition 5]. This means that the ALD cuts are sufficient in the sense of Definition 5.1. The guaranteed Lipschitz constant differs from that for the CPC, though. Again, as no temporary lifting is required, the overestimation result holds at x_n^i instead of $x_{\mathcal{B}n}^i$. This type of ALD cut is formally introduced in [4] and referred to as a *generalized conjugacy* (GC) cut, as it is a special case of cuts that can be derived using generalized conjugacy theory.

2.2.3 Overall comparison

We provide an overview of this comparison in Table 2.

The previous comparison seems to indicate that ALD cuts are superior to our proposed CPC, as they achieve tightness without the state space lifting and the projection back to the original state space. This seems to be particularly true for GC cuts, which also ensure Lipschitz continuity without relying on a strong recourse assumption.

In fact, it is true that our proposed non-convex approximations suffer from some noteworthy drawbacks:

- They require to solve a Lagrangian dual problem in a lifted, *i.e.*, higher-dimensional space, which is computationally detrimental.
- The obtained CPCs are tight at $x^i_{\mathcal{B},n}$ instead of x^i_n, with the approximation error depending on the precision of the state space binarization.
- Compared to ALD cuts, the non-convex CPC requires more (binary) variables and constraints to be represented by MILP constraints and to be included in the SDDiP subproblems.
- The Lagrangian dual problem often exhibits degeneracy, see Sect. 5.8. With a second optimization step minimizing the norm of all optimal solutions, usually cuts with suitable space-filling behavior can be computed, though.

However, we may also observe some challenges for ALD cuts.

- As mentioned before, for an unbounded dual problem, the cuts may become extremely steep close to discontinuities of $Q_n(\cdot)$. Therefore, convergence may not be guaranteed.
- Augmented Lagrangian dual problems are in general more expensive to solve than classical Lagrangian dual problems. One reason is that in contrast to problem (10), the inner problem (48) is nonlinear in general due to the norm in the objective function, even though for $\|\cdot\|_1$ or $\|\cdot\|_\infty$, it can be reformulated as an MILP. Due to the computational challenges, instead of optimizing over both π_n and ρ_n, in practice often a fixed value $\rho_n > 0$ is considered and iteratively increased (similarly to σ_n in [3]).
- For a fixed $\rho_n \geq \sigma_n$, we may end up with $\pi_n = 0$, which yields a trivial reverse-norm cut that is tight for $\underline{\mathcal{Q}}^{R;i+1}_{\mathcal{C}(n)}(\cdot;\sigma_n\|\cdot\|)$, but also very steep. The reason is that it only takes the Lipschitz constant of $\underline{\mathcal{Q}}^{R;i+1}_{\mathcal{C}(n)}(\cdot;\sigma_n\|\cdot\|)$ into account, but not its actual shape. This may result in a suboptimal space-filling, *i.e.*, the approximation is very loose outside of x^i_n. Even minimizing the norm of all dual optimal solutions does not help in this case, as $\pi_n = 0$ implies $\|\pi_n\| = 0$.

  In order to obtain better space-filling cuts, then ρ_n has to be reduced, at the risk of losing tightness. Ahmed et al. [1] propose the following mechanism: Compute π_n, either by solving the augmented Lagrangian dual problem for some fixed ρ_n or by solving the dual of the LP relaxation, and then use a bisection procedure to find the smallest ρ_n for which tightness is still ensured. This approach is heuristic, though.

We further investigate these theoretical differences using an illustrative example.

Table 2: Comparison of different Lagrangian cut approaches and their properties.

Cut type	Dual problem		Tightness/Overestimation			Lipschitz
	type	bounds	for function	at	given that	
Case 1: $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ is α_n-Lipschitz						
LC	LD	yes	$\sum_{m \in \mathcal{C}(n)} p_{nm} \overline{\text{co}}(\underline{Q}_m^{i+1})(\cdot)$	x_n^i	$\sigma_n \geq \alpha_n$	yes
CPC	lifted LD	yes	$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(\cdot)$	$x_{\mathcal{B},n}^i$	$\sigma_n \geq \alpha_n$	yes
ALD	ALD	no	$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(\cdot)$	x_n^i	-	(yes)
GC	ALD	yes	$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(\cdot)$	x_n^i	$\sigma_n \geq \alpha_n$	yes
Case 2: $\mathcal{Q}_{\mathcal{C}(n)}(\cdot)$ is not Lipschitz						
LC	LD	yes	$\sum_{m \in \mathcal{C}(n)} p_{nm} \overline{\text{co}}(\underline{Q}_m^{R;i+1}; \sigma_n \|\cdot\|)(\cdot)$	x_n^i	-	yes
CPC	lifted LD	yes	$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(\cdot; \sigma_n \|\cdot\|)$	$x_{\mathcal{B},n}^i$	-	yes
ALD	ALD	no	$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(\cdot)$	x_n^i	-	no
GC	ALD	yes	$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{R;i+1}(\cdot; \sigma_n \|\cdot\|)$	x_n^i	-	yes

LC/LD = classical Lagrangian cut/dual

Example 2.1. *Consider the problem* (7).

Let the incumbent be $x = \frac{6}{5}$. *We compare three different non-convex cuts for different values of* σ. *First, the CPC obtained in Example 5.14 for fixed binary precision* ($K = 4$). *Second, ALD cuts obtained by solving an augmented Lagrangian dual problem for fixed* $\rho = \sigma$. *For this example, the obtained cuts are the same, no matter if we introduce dual bounds or not. Third, the trivial ALD cuts defined by* $\rho = \sigma$ *and* $\pi = 0$. *The cuts are displayed in Fig. 2 (CPC: blue dashed line, best ALD cut: cyan dotted line, trivial ALD cut: magenta dash-dotted line).*

We observe that the ALD cuts may be tighter at the trial point, especially since they are constructed at this point and not at a deviating anchor point. On the other hand, the CPC yields much better space-filling approximations than the trivial ALD cut. Compared to the best obtained ALD cut, at least for larger values of σ, *the space-filling is better as well. For smaller values of* σ *this is not always the case.*

We compare the same types of cuts for the incumbent $x = 1.249$, *which is close to a discontinuity. Again, the ALD cuts are the same, no matter if we introduce dual bounds or not. The CPC is as tight as the trivial ALD cut at* $x = 1.249$, *but exhibits better approximation quality elsewhere. It is also slightly less steep than the best obtained ALD cut.* □

We should note that the ALD cuts could be further improved in Example 2.1, for instance by applying the bisection heuristic from [1] and optimizing over ρ. In general, the quality of the obtained cuts seems to rather depend on fine-tuning and parameter choices, *e.g.*, selecting appropriate σ or ρ, or in the case of dual degeneracy, obtaining the best possible space-filling without compromising tightness, and less on whether our proposed lift-and-project or an augmented Lagrangian approach is used.

References

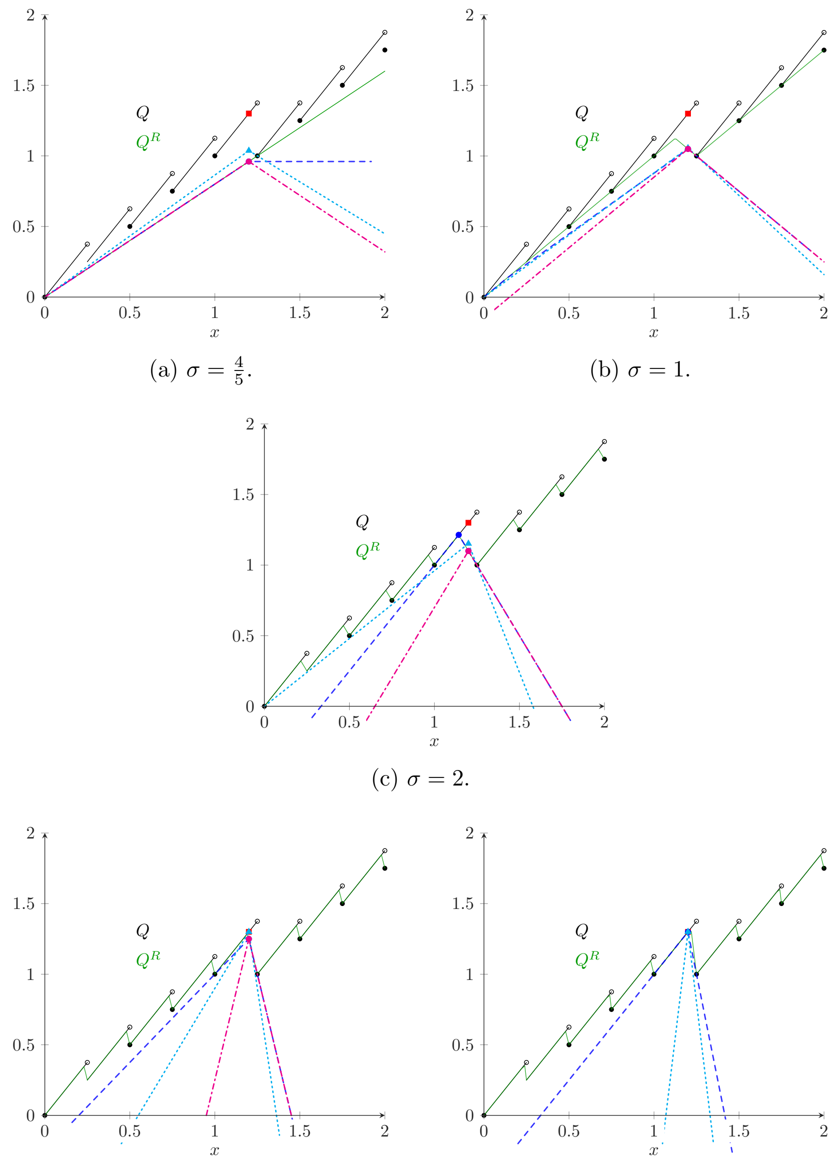
[1] S. Ahmed, F. G. Cabral, and B. Freitas Paulo da Costa. Stochastic Lipschitz dynamic programming. *Mathematical Programming*, 191:755–793, 2022.

[2] S. Ahmed, M. Tawarmalani, and N.V. Sahinidis. A finite branch-and-bound algorithm for two-stage stochastic integer programs. *Mathematical Programming*, 100:355–377, 2004.

[3] C. Füllner and S. Rebennack. Non-convex nested Benders decomposition. *Mathematical Programming*, 196:987–1024, 2022.

[4] S. Zhang and X. A. Sun. Stochastic dual dynamic programming for multi-stage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 196:935–985, 2022.

[5] J. Zou, S. Ahmed, and X. A. Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.



(a) $\sigma = \frac{4}{5}$.

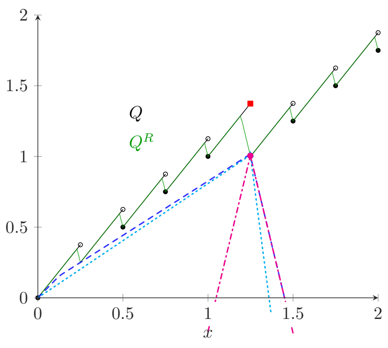
(b) $\sigma = 1$.

(c) $\sigma = 2$.

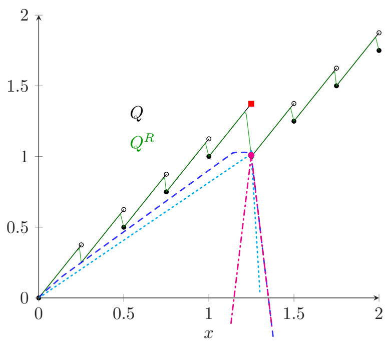
(d) $\sigma = 5$.

(e) $\sigma = 10$.

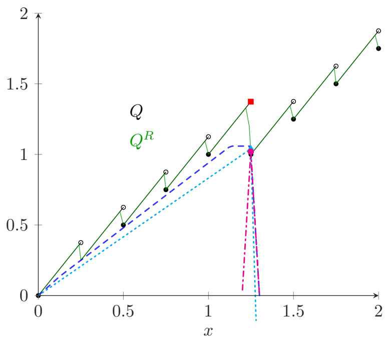
Figure 2: CPC and ALD cuts at $x = \frac{6}{5}$ for different values of σ in Example 2.1.



(a) $\sigma = \frac{4}{5}$.



(b) $\sigma = 1$.



(c) $\sigma = 2$.

Figure 3: CPC and ALD cuts at $x = 1.249$ for different values of σ in Example 2.1.

Paper D

A new framework to generate Lagrangian cuts in multistage stochastic mixed-integer programming

Christian Füllner[a], X. Andy Sun[b], Steffen Rebennack[a]

[a] *Karlsruhe Institute of Technology (KIT), Institute for Operations Research, Stochastic Optimization, Karlsruhe, Germany*

[b] *Sloan School of Management, Massachusetts Institute of Technology, USA*

working paper, suggested citation:
Füllner C., Sun X. A., Rebennack S. (2024), A new framework to generate Lagrangian cuts in multistage stochastic mixed-integer programming.

A new framework to generate Lagrangian cuts in multistage stochastic mixed-integer programming

Christian Füllner[1*], X. Andy Sun[2], and Steffen Rebennack[1]

[1]Institute for Operations Research (IOR), Stochastic Optimization (SOP), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[*]Address correspondence to: christian.fuellner@kit.edu
[2]Sloan School of Management, Massachusetts Institute of Technology, USA

Abstract

Based on recent advances in Benders decomposition and two-stage stochastic integer programming we present a new generalized framework to generate Lagrangian cuts in multistage stochastic mixed-integer linear programming (MS-MILP). This framework can be incorporated into decomposition methods for MS-MILPs, such as the stochastic dual dynamic integer programming (SDDiP) algorithm. We show how different normalization techniques can be applied in order to generate cuts satisfying specific properties with respect to the convex hull of the epigraph of the value functions, e.g. having a maximum depth or being facet-defining. We provide computational results to evaluate the efficacy and performance of different normalizations in our new framework and compare them with existing techniques from the literature.

1 Introduction

In this paper, we study cut generation strategies that can be applied in decomposition methods for solving multi-stage stochastic mixed-integer linear programs (MS-MILP). More precisely, we present an alternative framework for the generation of Lagrangian cuts as they are used for instance in stochastic dual dynamic integer programming (SDDiP) proposed by [31].

1.1 Motivation and Prior Work

Multistage stochastic programs are very relevant to model decision-making processes in practice because often sequential decisions have to be made over a finite number of stages and under uncertainty considering the problem data of the following stages. For a large number of considered scenarios, solving these problems with standard solvers is computationally intractable, as their size grows exponentially in the number of stages. For this reason, decomposition methods exploiting their sequential and block-diagonal structure have been established as predominant solution techniques, among the most prominent ones nested Benders decomposition (BD) [6] and stochastic dual dynamic programming (SDDP) [24]. These methods decompose the large-scale problem into stage- and scenario-specific subproblems, coupled by state variables and so-called *value functions*, denoted $Q_n(\cdot)$. For multistage stochastic linear programming problems (MS-LPs) these functions are convex polyhedral and can be exactly represented by finitely many affine functions called *cuts* [7]. However, in many applications some of the decision variables have to be integer or binary to model more complicated constraints. In this case, we obtain an MS-MILP and the value functions are in general non-convex and discontinuous.

A key challenge is that in this case, linear under-estimators are in general not tight. They may at best yield the closed convex envelope $\overline{\text{co}}(Q_n)(\cdot)$ of $Q_n(\cdot)$, which is the pointwise supremum of all affine functions majorized by $Q_n(\cdot)$ [4]. Even this property is not achieved by classical Benders cuts in general. Therefore, more focus has been put on Lagrangian cuts lately, which are constructed by solving special Lagrangian dual problems, as introduced in the original SDDiP paper [31]. These Lagrangian cuts have useful properties: They are valid under-estimators of $Q_n(\cdot)$ and they can be used to recover $\overline{\text{co}}(Q_n)(\cdot)$. As shown in [31], if all state variables of the MS-MILP are binary, this even ensures tightness for $Q_n(\cdot)$, which is sufficient to establish almost sure finite convergence of SDDiP.

Nonetheless, applying Lagrangian cuts computationally in practice comes with some considerable challenges: First, Lagrangian dual problems are often degenerate with multiple optimal solutions. Even if all the cuts associated with these solutions are tight, their approximation quality may differ significantly, as illustrated in Fig. 1. This issue is especially common for the binary state space required in SDDiP because all cuts are constructed at extreme points of the state space.

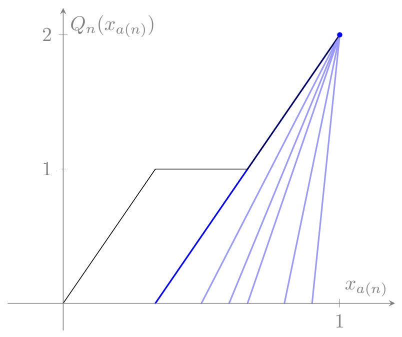


Figure 1: Tight Lagrangian cuts with different approximation quality.

Second, even if the Lagrangian dual is a convex optimization problem, it may be very costly to solve repeatedly due to its non-smooth objective function. This is only aggravated if the state space is artificially increased by a binary expansion, as it is proposed in SDDiP to ensure cut tightness in the case of non-binary state variables [31]. This computational drawback of Lagrangian cuts is already identified in the original SDDiP work [31]. It is concluded that performance-wise the improvement in cut quality is often not worth the significant increase in solution time.

Finally, tight Lagrangian cuts are crucial to ensure theoretical convergence of SDDiP, but this convergence may be quite slow. In the worst case, a complete enumeration of the binary state space is required. It is conceivable that there exist alternative, possibly non-tight, cuts that may significantly speed-up the convergence process.

For the aforementioned reasons, in this paper we address the question of how to improve and accelerate the generation and usage of Lagrangian cuts in decomposition methods for MS-MILP such as SDDiP.

Some first attempts with this aim have been made recently. In the original SDDiP paper it is proposed to combine tight cuts with strengthened Benders cuts that are not tight in general, but outperform classical Benders cuts and are efficient to compute [31]. Rahmaniani et al. [25] present a heuristic to generate Lagrangian cuts more efficiently using inner approximations or partial relaxations. Chen and Luedtke [11] suggest to restrict the feasible set of dual multipliers to the span of Benders cut coefficients of previous iterations (without convergence guarantees for the multi-stage case).

In addition, there has been a lot of research on alternative cut generation techniques for BD, which may be applicable to the stochastic and Lagrangian setting as well. To address degeneracy and dominated cuts in BD, in their seminal paper [22] present a two-step approach to generate Pareto-optimal cuts. Their ideas are improved by [23] who shows that it is sufficient to solve a single optimization problem to generate Pareto-optimal cuts. Sherali and Lunday [27] propose to generate certain maximal non-dominated Benders cuts by solving a perturbation of the original subproblem.

A novel framework to generate Benders cuts is introduced by [17]. Its starting point is the observation that in classical BD there exists an unfavorable bias towards feasibility cuts over optimality cuts. As an alternative, the proposed framework allows to generate optimality and feasibility cuts in a unified way using the same cut generation problem. Additionally, based on this unified framework, several different cut generation techniques can be explored. More precisely, applying the framework initially leads to an unbounded separation problem. For the actual cut generation, unbounded rays have to be identified, which allows for a lot of methodological flexibility.

Fischetti et al. [17] show that using a special normalization of the cut generation problem, the obtained cuts can be proven to correspond to minimal infeasible subsystems. Hosseini and Turner [21] use a different normalization to generate *deep* cuts in BD, which are characterized by their property to maximize the distance between the separating hyperplane and the point to separate. They report considerable performance gains compared to classical BD. The idea of deep cuts is not new, but priorly discussed in context of disjunctive programming [10, 13]. Brandenberg and Stursberg [9] show how facet-defining and Pareto-optimal cuts can be generated in BD using the unified framework and the so-called reverse polar set [see also 28]. It is shown that the performance improvement using this approach is significant. The same cut generation procedure is put forward by [26] in their recent paper, but motivated from a different angle, that is geometrically. A different approach to separate facet-defining cuts is presented by [12] for disjunctive programming. To our knowledge, only the work by Fischetti et al. has been applied to the stochastic setting and to Lagrangian cuts so far [11]. The authors use the alternative framework and a specific normalization technique to generate Lagrangian cuts for two-stage stochastic MILPs.

In this paper, we provide a more general framework for the multistage case and compare various different normalization techniques from a theoretical and computational perspective. In particular, we analyze the Lagrangian cuts that are obtained if the Lagrangian dual is normalized using norm constraints or linear constraints. We show that under some assumptions, these cuts are deep, facet-defining or Pareto-optimal.

1.2 Contribution

The key contributions of this paper are summarized below.

(1) We show how the alternative cut generation framework proposed by [17] for BD can be applied to the generation of Lagrangian cuts for MS-MILPs. This idea has already been used by [11] in two-stage stochastic MILPs, but to our knowledge has not been extended to the multistage case yet.

(2) As the Lagrangian dual problems in this framework are unbounded, some normalization is required to select cut coefficients in a reasonable way. We draw on recent concepts for cut selection in BD, such as optimizing over the reverse polar set [9] or generating deep cuts [21], and extend them to the stochastic and Lagrangian setting. This way, we obtain a variety of different normalization techniques and by that generalize the cut generation approach from [11]. We show that depending on the chosen normalization, cuts satisfying different quality criteria can be obtained, *e.g.*, deep cuts, facet-defining cuts or Pareto-optimal cuts. Moreover, we investigate in detail the geometrical ideas and relations behind these normalizations.

(3) We show that linear normalizations are closely related to the identification of core points in the epigraphs $\mathrm{epi}(Q_n)$, which can be challenging for multistage stochastic problems. Therefore, we propose five heuristic approaches for the computation of core point candidates, and thus setting up linear normalizations for the generation of Lagrangian cuts.

(4) The proposed framework for Lagrangian cut generation can be incorporated into NBD or SDDiP. We prove that under some assumptions, still (almost sure) finite convergence of these methods is guaranteed.

(5) We perform extensive computational tests for SDDiP incorporating the new cut generation framework on a capacitated lot-sizing problem from the literature. We show that the obtained lower bounds in SDDiP are majorly improved using cuts from our proposed generation framework compared to classical Lagrangian cuts or Benders cuts. We also observe that this does not necessarily guarantee an improvement of the in-sample performance of the obtained policies, though.

1.3 Structure

This paper is structured as follows. In Sect. 2 we introduce MS-MILPs formally together with our notation. In Sect. 3 we introduce the new cut generation framework for Lagrangian cuts in general. We then present different types of Lagrangian cuts that can be obtained by using special normalizations. In Sect. 4 we discuss convergence of NBD and SDDiP if these cuts are incorporated. After that, in Sect. 5, we present computational experiments for SDDiP with these new Lagrangian cuts for a capacitated lot-sizing problem from the literature. We finish with a conclusion in Sect. 6. For reasons of space, some technical proofs are shifted to Appendix B.

2 Problem Formulation

We start by introducing MS-MILPs and their decomposition formally, mostly following the notation from [31]. We consider MS-MILPS with a finite number $T \in \mathbb{N}$ of stages, where some of the problem data is uncertain and evolves according to a known stochastic process $\xi := (\xi_1, \dots, \xi_T)$ with deterministic ξ_1. We assume that the random data vectors

$\xi_t, t = 1, \dots, T$, are discrete and finite, such that the uncertainty can be modeled by a finite scenario tree. Let $\mathcal{N}$ denote the set of nodes of this tree. For each node $n \in \mathcal{N}$, the unique ancestor node is denoted by $a(n)$ and the set of child nodes is denoted by $\mathcal{C}(n)$. The probability for some node n is $p_n > 0$ and assumed to be known. The transition probabilities between adjacent nodes $n, m \in \mathcal{N}$ can then be determined as $p_{nm} := \frac{p_m}{p_n}$. For the root node r, we assume $a(r) = \emptyset$ and $p_r = 1$. We define $\overline{\mathcal{N}} := \mathcal{N} \setminus \{r\}$ to address the set of nodes without the root node, $\widetilde{\mathcal{N}}$ to address the set of nodes without leaf nodes and denote by $\mathcal{N}(t)$ the nodes at stage t.

For each node $n \in \mathcal{N}$, we distinguish state variables $x_n \in \mathbb{R}^{d_n}$, which also appear in child nodes of n, and local variables $y_n \in \mathbb{R}^{\tilde{d}_n}$. $f_n(\cdot)$ denotes the objective function of node n and $\mathcal{F}_n(x_{a(n)})$ denotes the feasible set of node n, which depends on the state variable $x_{a(n)}$ from the ancestor node. We assume that $f_n(\cdot)$ is a linear function in x_n and y_n, and that $\mathcal{F}_n(\cdot)$ is a mixed-integer polyhedral set for all $x_{a(n)}$. More precisely, we assume it to be defined by

$$\begin{aligned}\mathcal{F}_n(x_{a(n)}) := \Big\{(x_n, y_n) \in \mathbb{R}^{d_n} \times \mathbb{R}^{\tilde{d}_n} \ : x_n \in X_n,\ y_n \in Y_n, \\ A_n x_{a(n)} + B_n x_n + C_n y_n \geq b_n\Big\}.\end{aligned} \tag{1}$$

Here, A_n, B_n, C_n, b_n denote appropriately defined data matrices and vectors. The sets X_n and Y_n comprise constraints only associated with x_n or y_n, *e.g.*, box constraints or non-negativity constraints. More precisely, we assume that both sets are intersections of polyhedral sets $\bar{X}_n, \bar{Y}_n$ and possible integrality constraints. In the following, we also refer to X_n as the *state space*.

An MS-MILP can then be expressed by its dynamic programming equations. For the root node, we obtain

$$v^* := \min_{x_r, y_r} \Big\{ f_r(x_r, y_r) + \sum_{m \in \mathcal{C}(r)} p_{rm} Q_m(x_r) \ : \ (x_r, y_r) \in \mathcal{F}_r(x_{a(r)}) \Big\} \tag{2}$$

with $x_{a(r)} = 0$, and v^* is the optimal value of the original problem. Let $\overline{\mathbb{R}} := \mathbb{R} \cup \{+\infty\}$. For all $n \in \overline{\mathcal{N}}$, the value function $Q_n : \mathbb{R}^{d_{a(n)}} \to \overline{\mathbb{R}}$ is defined by

$$Q_n(x_{a(n)}) := \min_{x_n, y_n} \Big\{ f_n(x_n, y_n) + \sum_{m \in \mathcal{C}(n)} p_{nm} Q_m(x_n) \ : \ (x_n, y_n) \in \mathcal{F}_n(x_{a(n)}) \Big\}.$$

For the leaf nodes $n \in \mathcal{N} \setminus \widetilde{\mathcal{N}}$, we set $\sum_{m \in \mathcal{C}(n)} p_{nm} Q_m(x_n) \equiv 0$. Moreover, we set $Q_n(x_{a(n)}) = +\infty$ if $\mathcal{F}_n(x_{a(n)}) = \emptyset$, and denote by $\mathrm{dom}(Q_n)$ the *effective domain* of $Q_n(\cdot)$.

Remark 2.1. *Note that regarding $Q_n(\cdot)$ as a function on $\mathbb{R}^{d_{a(n)}}$ is not standard in stochastic programming. Often it is (implicitly) assumed to be defined only on the domain $X_{a(n)}$. However, from our view, allowing $Q_n(\cdot)$ to be defined on $\mathbb{R}^{d_{a(n)}}$ with extended real values appears more suitable for the following steps.*

As this proves beneficial in the cut generation process, we introduce local variables z_n and accompany them with copy constraints $x_{a(n)} = z_n$ and constraints $z_n \in Z_{a(n)}$, with $Z_{a(n)} \supseteq X_{a(n)}$. The most natural choice is $Z_{a(n)} = X_{a(n)}$, but also other choices are possible, *e.g.*, $Z_{a(n)} = \mathrm{conv}(X_{a(n)})$. For more details we refer to [18]. This reformulation yields the equivalent subproblems

$$Q_n(x_{a(n)}) = \min_{x_n, y_n, z_n} \Big\{ f_n(x_n, y_n) + \sum_{m \in \mathcal{C}(n)} p_{nm} Q_m(x_n) \ : (z_n, x_n, y_n) \in \mathcal{F}_n, \\ z_n = x_{a(n)}, \ z_n \in Z_{a(n)} \Big\}, \tag{3}$$

where, $\mathcal{F}_n := \Big\{ (x_n, y_n, z_n) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R}^{d_n} \times \mathbb{R}^{\tilde{d}_n} \ : \ (x_n, y_n) \in \mathcal{F}_n(z_n) \Big\}$.

For the remainder of this article, we make some basic assumptions.

Assumption 1. *The following conditions are satisfied by* (1)-(3)*:*

(A1) For all $n \in \mathcal{N}$, the sets X_n and Y_n are compact.

(A2) For all $n \in \mathcal{N}$, all coefficients in A_n, B_n, C_n, b_n, f_n, $\bar{X}_n$ and $\bar{Y}_n$ are rational.

(A3) For all $n \in \overline{\mathcal{N}}$, $Z_{a(n)}$ is compact, rational MILP-representable and $Z_{a(n)} = \mathrm{dom}(Q_n)$.

Note that the boundedness in (A1) immediately implies that $F_n(x_{a(n)})$ is bounded for all $x_{a(n)} \in \mathbb{R}^{d_{a(n)}}$ and $n \in \mathcal{N}$. Property (A3) implies *relatively complete recourse*, which is a standard assumption in multistage stochastic programming. It allows us to focus on optimality cuts approximating $Q_n(\cdot)$ without requiring feasibility cuts that approximate $\mathrm{dom}(Q_n)$.

We obtain the following properties for the value functions.

Lemma 2.2. *Under Assumption 1, for all $n \in \overline{\mathcal{N}}$, the value functions $Q_n(\cdot)$ are proper, l.sc. (lower semicontinuous) and piecewise polyhedral with finitely many pieces.*

By applying the properness reasoning to the root node, we conclude that v^* is finite.

3 A New Cut Generation Framework

In this section, we present a novel framework for the generation of Lagrangian cuts in multistage stochastic mixed-integer programming, which serves as an alternative to the classical Lagrangian cut generation framework from SDDiP [31], which we state in Appendix A for comparison. The new framework is based on ideas that go back to [17], and in one specific form has been applied to two-stage stochastic programs in [11].

3.1 An Epigraph Perspective on Cut Generation

Recall the definition of the epigraph of the value functions $Q_n(\cdot), n \in \overline{\mathcal{N}}$:

$$\mathrm{epi}(Q_n) = \Big\{ (x_{a(n)}, \theta_n) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R} \ : \ \theta_n \geq Q_n(x_{a(n)}), \ x_{a(n)} \in \mathrm{dom}(Q_n) \Big\}. \tag{4}$$

We can use this definition to reformulate the subproblems (3) to

$$Q_n(x_{a(n)}) = \min_{x_n, y_n, z_n, (\theta_m)} \Big\{ f_n(x_n, y_n) + \sum_{m \in \mathcal{C}(n)} p_{nm} \theta_m \ : (z_n, x_n, y_n) \in \mathcal{F}_n, \\ z_n = x_{a(n)}, \ z_n \in Z_{a(n)} \\ (x_n, \theta_m) \in \mathrm{epi}(Q_m), \ m \in \mathcal{C}(n) \Big\}. \tag{5}$$

Here, and in the following, we use (θ_m) as a shortened notation for $(\theta_m)_{m\in\mathcal{C}(n)}$.

Remark 3.1. *The condition $x_n \in \operatorname{dom}(Q_m)$ from the definition in* (4) *is always satisfied implicitly in problem* (5)*, since* $\operatorname{dom}(Q_m) = Z_n$ *by Assumption 1 and $x_n \in X_n \subseteq Z_n$ by construction.*

In classical Benders-like decomposition methods, such as SDDiP, iteratively *optimality cuts* are constructed to approximate $Q_n(\cdot)$ and, if required, *feasibility cuts* are constructed to approximate $\operatorname{dom}(Q_n)$. This is done by solving distinct cut generation problems. However, from (4) it is evident that both types of cuts actually approximate $\operatorname{epi}(Q_n)$. Therefore, we may as well consider a unified cut generation problem to obtain polyhedral approximations Ψ_m of the sets $\operatorname{epi}(Q_m)$ [17]. Whereas we solely focus on optimality cuts in this paper, see Assumption 1, the resulting cut generation framework still proves itself valuable, as we shall see.

Remark 3.2. *In multistage stochastic programming the cuts are often aggregated to obtain cuts for the expected value functions $\sum_{m\in\mathcal{C}(n)} p_{nm} Q_m(x_n)$ (*single-cut *approach), as this reduces the total number of cuts in the subproblems. In this paper, instead, we consider a separate set of cuts,* i.e.*, separate approximations Ψ_m, for each* $\operatorname{epi}(Q_m)$ *(*multi-cut *approach). This approach is better suited to our cut generation framework.*

As the value functions $Q_n(\cdot)$ are not known explicitly within our decomposition method, we may not use subproblems (5) directly in the cut generation process. However, we can replace each occurrence of $\operatorname{epi}(Q_m)$ with its current approximation Ψ_m^{i+1} (with iteration index i). We refer to the associated value functions $\underline{Q}_m^{i+1}(\cdot)$ as *approximate value functions.* Note that this way, we actually generate cuts approximating $\operatorname{epi}(\underline{Q}_m^{i+1})$. However, by construction these cuts do also yield outer approximations of $\operatorname{epi}(Q_m)$. To avoid unboundedness, each set is initialized with a valid outer approximation $\Psi_m^0, m \in \mathcal{C}(n)$.

For notational simplicity, for the remainder of this paper we define the set

$$\mathcal{W}_n^{i+1} := \Big\{ \big(x_n, y_n, z_n, (\theta_m)\big) \;:\; (x_n, y_n, z_n) \in \mathcal{F}_n,\; z_n \in Z_{a(n)},\; (x_n, \theta_m) \in \Psi_m^{i+1}, m \in \mathcal{C}(n) \Big\},$$

and further define $\lambda_n := \big(x_n, y_n, (\theta_m)\big)$ and $c_n^\top \lambda_n := f_n(x_n, y_n) + \sum_{m\in\mathcal{C}(n)} p_{nm}\theta_m$ (recall that $f(\cdot)$ is linear). Then, the approximate subproblems and approximate value functions associated with problem (5) can be compactly written as

$$\underline{Q}_n^{i+1}(x_{a(n)}^i) = \min_{\lambda_n, z_n} \left\{ c_n^\top \lambda_n \;:\; (\lambda_n, z_n) \in \mathcal{W}_n^{i+1},\; z_n = x_{a(n)}^i \right\}. \tag{6}$$

We make another assumption for the remainder of this paper.

Assumption 2. *For all $n \in \overline{\mathcal{N}}$ and all iterations i, all linear cuts defining the polyhedral set Ψ_m^{i+1} are defined by rational coefficients.*

Furthermore, in the next sections, we often require the convex hull $\operatorname{conv}(\mathcal{W}_n^{i+1})$ of a set $\mathcal{W}_n^{i+1}$. It has the following important property.

Remark 3.3. *Under Assumptions 1 and 2, the set* $\operatorname{conv}(\mathcal{W}_n^{i+1})$ *is a closed convex polyhedron. That means that there exist matrices $\tilde{A}_n, \tilde{B}_n$ and a vector $\tilde{d}_n$ such that*

$$\operatorname{conv}(\mathcal{W}_n^{i+1}) = \Big\{ (\lambda_n, z_n) \;:\; \tilde{A}_n \lambda_n + \tilde{B}_n z_n \geq \tilde{d}_n \Big\}.$$

3.2 A Feasibility Problem for the Epigraph

We can now start to address the actual cut generation process in the proposed unified framework. Given a point $(x^i_{a(n)}, \theta^i_n)$, we consider the feasibility problem

$$v_n^{f,i+1}(x^i_{a(n)}, \theta^i_n) := \min_{\lambda_n, z_n} \Big\{ 0 \ : \ (\lambda_n, z_n) \in \mathcal{W}_n^{i+1}, \ z_n = x^i_{a(n)}, \ \theta^i_n \geq c_n^\top \lambda_n \Big\}, \tag{7}$$

which can be shown to verify if $(x^i_{a(n)}, \theta^i_n) \in \operatorname{epi}(\underline{Q}_n^{i+1})$.

Lemma 3.4. *Under Assumptions 1 and 2 and given a point* $(x^i_{a(n)}, \theta^i_n)$*, problem* (7) *is a feasibility problem for* $\operatorname{epi}(\underline{Q}_n^{i+1})$*, that is,*

$$v_n^{f,i+1}(x^i_{a(n)}, \theta^i_n) = \begin{cases} 0, & \textit{if } (x^i_{a(n)}, \theta^i_n) \in \operatorname{epi}(\underline{Q}_n^{i+1}) \\ +\infty, & \textit{else.} \end{cases}$$

Proof. Let $(x^i_{a(n)}, \theta^i_n) \in \operatorname{epi}(\underline{Q}_n^{i+1})$. Then according to (4) we have

$$\theta^i_n \geq \min_{\lambda_n, z_n} \Big\{ c_n^\top \lambda_n \ : \ (\lambda_n, z_n) \in \mathcal{W}_n^{i+1}, \ z_n = x^i_{a(n)} \Big\}. \tag{8}$$

This implies that there exists some (λ_n, z_n) such that for $(x^i_{a(n)}, \theta^i_n)$ all constraints of (7) are satisfied. Hence, $v_n^{f,i+1}(x^i_{a(n)}, \theta^i_n) = 0$.

Let $v_n^{f,i+1}(x^i_{a(n)}, \theta^i_n) = 0$. Then, there exist (λ_n, z_n) such that for $(x^i_{a(n)}, \theta^i_n)$ all constraints of (7) are satisfied. However, this implies (8), and thus $(x^i_{a(n)}, \theta^i_n) \in \operatorname{epi}(\underline{Q}_n^{i+1})$. □

3.3 Lagrangian Cuts in the New Framework

To generate Lagrangian cuts, we apply a Lagrangian relaxation to problem (7). A key difference to the classical Lagrangian relaxation from SDDiP (see Appendix A) is that not only $x^i_{a(n)}$, but $(x^i_{a(n)}, \theta^i_n)$ is regarded as a fixed incumbent for the cut generation process. Therefore, we relax all constraints containing either $x^i_{a(n)}$ or θ^i_n. For given multipliers $(\pi_n, \pi_{n0}) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R}^+$ the dual function is then given by

$$\mathscr{L}_n^{i+1}(\pi_n, \pi_{n0}) := \min_{\lambda_n, z_n} \Big\{ \pi_n^\top z_n + \pi_{n0} c_n^\top \lambda_n \ : \ (\lambda_n, z_n) \in \mathcal{W}_n^{i+1} \Big\}. \tag{9}$$

The corresponding Lagrangian dual problem is

$$\max_{\pi_n, \pi_{n0}} \Big\{ \mathscr{L}_n^{i+1}(\pi_n, \pi_{n0}) - \pi_n^\top x^i_{a(n)} - \pi_{n0} \theta^i_n \ : \ \pi_{n0} \geq 0 \Big\}. \tag{10}$$

We state some important properties of this dual problem.

Theorem 3.5. *Under Assumptions 1 and 2, for the Lagrangian dual* (10) *it holds:*

(i) The dual function $\mathscr{L}_n^{i+1}(\cdot)$ *is piecewise linear concave in* (π_n, π_{n0})*.*

(ii) Its optimal value $\widehat{v}_n^{D,i+1}(x^i_{a(n)}, \theta^i_n)$ *satisfies*

$$\widehat{v}_n^{D,i+1}(x^i_{a(n)}, \theta^i_n) = \begin{cases} 0, & \textit{if } (x^i_{a(n)}, \theta^i_n) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) \\ +\infty, & \textit{else.} \end{cases}$$

Proof. (i) is a standard result on Lagrangian relaxation [see 20]. Another well-known property of the Lagrangian dual is that it is equivalent to a primal convexification of the original subproblem [19]. In our case, this convexification is given by

$$\min_{\lambda_n, z_n} \Big\{ 0 \ : \ (\lambda_n, z_n) \in \operatorname{conv}(\mathcal{W}_n^{i+1}),\ z_n = x_{a(n)}^i,\ \theta_n^i \geq c_n^\top \lambda_n \Big\}. \tag{11}$$

The closed convex envelope $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$ can be expressed through the convex problem

$$\overline{\text{co}}(\underline{Q}_n^{i+1})(x_{a(n)}^i) = \min_{\lambda_n, z_n} \Big\{ c_n^\top \lambda_n \ : \ (\lambda_n, z_n) \in \operatorname{conv}(\mathcal{W}_n^{i+1}),\ z_n = x_{a(n)}^i \Big\}, \tag{12}$$

see for instance [18, Theorems 3.8 and 3.9] for a formal proof. Therefore, by the same reasoning as in Lemma 3.4, problem (11) is a feasibility problem for $\operatorname{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$. $\square$

Theorem 3.5 directly implies that the Lagrangian dual is unbounded whenever $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$. Therefore, there exists an unbounded ray (π_n^i, π_{n0}^i) such that

$$\mathscr{L}_n^{i+1}(\pi_n^i, \pi_{n0}^i) - (\pi_n^i)^\top x_{a(n)}^i - \pi_{n0}^i \theta_n^i > 0,$$

and $(x_{a(n)}^i, \theta_n^i)$ violates the following Lagrangian cut:

Definition 3.6. *For all $n \in \overline{\mathcal{N}}$ and some multipliers (π_n^i, π_{n0}^i), a Lagrangian cut is given by*

$$\pi_{n0}^i \theta_n + (\pi_n^i)^\top x_{a(n)} \geq \mathscr{L}_n^{i+1}(\pi_n^i, \pi_{n0}^i). \tag{13}$$

This type of cut is valid for any feasible (π_n^i, π_{n0}^i) in (10). We provide a proof in Appendix B.1.

Lemma 3.7. *Under Assumptions 1 and 2, for any $(\pi_n^i, \pi_{n0}^i) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R}^+$ the Lagrangian cut* (13) *is satisfied by all $(x_{a(n)}, \theta_n) \in \operatorname{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$, and thus by all $(x_{a(n)}, \theta_n) \in \operatorname{epi}(Q_n)$.*

We analyze the relation between the Lagrangian cuts (13) and the classical ones (22). We restrict to $\pi_{n0} > 0$ because $\pi_{n0} = 0$ leads to feasibility cuts that by assumption are not required in our case.

Remark 3.8. *Let $\pi_{n0}^i > 0$ in cut* (13)*. As shown in Proposition 1 in [11], with division by π_{n0}^i, it follows*

$$\begin{aligned}
\theta_n &\geq \frac{1}{\pi_{n0}^i} \mathscr{L}_n^{i+1}(\pi_n^i, \pi_{n0}^i) - \left(\frac{\pi_n^i}{\pi_{n0}^i} \right)^\top x_{a(n)} \\
&= \mathscr{L}_n^{i+1}(\widehat{\pi}_n, 1) - \widehat{\pi}_n^\top x_{a(n)} \\
&= \mathcal{L}_n^{i+1}(\widehat{\pi}_n) - \widehat{\pi}_n^\top x_{a(n)}
\end{aligned}$$

with $\widehat{\pi}_n := \frac{\pi_n^i}{\pi_{n0}^i}$. This is an equivalent representation of (13) *in the form of a classical Lagrangian optimality cut* (22)*, see Appendix A.*

We should emphasize that despite the scaling relation shown in Remark 3.8, the new cut generation framework may yield different cuts than the classical one because the choice of dual multipliers is based on a different dual problem.

3.4 Cut Selection Criteria

It is not immediately clear how to select cut coefficients (π_n^i, π_{n0}^i) from the unbounded Lagrangian dual (10) in the most reasonable way. On the one hand, computationally we aspire to determine coefficients (π_n^i, π_{n0}^i) by solving a bounded and feasible optimization problem instead of dealing with an unbounded one. On the other hand, we want to make sure that the obtained cuts are not only separating the incumbent $(x_{a(n)}^i, \theta_n^i)$ from $\text{epi}(Q_n)$, but also of good approximation quality. For instance, they should not be unnecessarily steep (see Sect. 1.1) and they should be supporting $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$.

The first aim can be achieved by bounding problem (10) artificially, *e.g.*, by introducing bounds on the dual multipliers. Another common approach is to fix its unbounded objective to 1. Combined with Remark 3.3 this allows to identify unbounded rays by analyzing a compact polyhedron called *alternative polyhedron* [17]. Finally, we may introduce a normalizing constraint to the Lagrangian dual (10).

With regard to the second aim, various quality criteria for cutting-planes have been put forward in the literature, see [14] for an overview. Many of these criteria have been applied in the context of BD or disjunctive programming before, as shown in Table 1, but our paper is the first one applying them to Lagrangian cuts, and incorporating most of them at once. We focus on three important criteria:

- *Facet-defining cuts.* These cuts reproduce facets of a convex polyhedral set, in our case $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$, and thus may be helpful in ensuring finite convergence.

- *Pareto-optimal cuts.* For $\pi_{n0} > 0$, Pareto-optimal cuts (13) are *non-dominated* in the sense that there exists no other cut

$$\tilde{\pi}_{n0}\theta_n + (\tilde{\pi}_n)^\top x_{a(n)} \geq \mathscr{L}_n^{i+1}(\tilde{\pi}_n, \tilde{\pi}_{n0})$$

  such that

$$\frac{\mathscr{L}_n^{i+1}(\tilde{\pi}_n, \tilde{\pi}_{n0}) - (\tilde{\pi}_n)^\top x_{a(n)}}{\tilde{\pi}_{n0}} \geq \frac{\mathscr{L}_n^{i+1}(\pi_n^i, \pi_{n0}^i) - (\pi_n^i)^\top x_{a(n)}}{\pi_{n0}^i}$$

  for all $(x_{a(n)}, \theta_n) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$. Note that Pareto-optimality with respect to $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ does not necessarily imply Pareto-optimality with respect to $\text{epi}(\underline{Q}_n^{i+1})$, but is easier to achieve. The idea of Pareto-optimal cuts was first put forward by [22].

- *Deep cuts.* The concept of deep cuts goes back to [8]. These cuts are *deep* in the sense that a maximum distance between the incumbent $(x_{a(n)}^i, \theta_n^i)$ and the separating hyperplane is realized, *i.e.*, they cut as deep as possible into the suboptimal region.

As shown in the literature, especially in [9], many of these criteria can be satisfied by optimizing over the so-called *reverse polar set* of $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ shifted by the incumbent $(x_{a(n)}^i, \theta_n^i)$. The reverse polar set is an important tool in the theory on cut generation, as it is directly linked to the support function of $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$, and thus provides a characterization of normal vectors of $(x_{a(n)}^i, \theta_n^i)$-separating hyperplanes [9, 13].

The reverse polar set was first introduced by [3] and can be defined as follows.

Table 1: Examination of cut quality criteria in the literature on BD and disjunctive programming.

Paper	MIS	Max. depth	Facet-def.	Pareto-opt.
Magnanti and Wong [22]				✓
Cornuéjols and Lemaréchal [13]		✓	✓	
Papadakos [23]				✓
Cadoux [10]		✓	✓	
Fischetti et al. [17]	✓			
Sherali and Lunday [27]				✓
Conforti and Wolsey [12]			✓	
Brandenberg and Stursberg [9]	✓		✓	✓
Hosseini and Turner [21]		✓		
Seo et al. [26]			✓	
This paper		✓	✓	✓

MIS: Cuts that correspond to minimal infeasible subsytems of the feasibility subproblem.

Definition 3.9. *The reverse polar set of a set $S \subset \mathbb{R}^n$ is defined as*

$$S^- := \Big\{ d \in \mathbb{R}^n \ : \ d^\top x \leq -1 \quad \forall x \in S \Big\}.$$

To simplify notation, we set $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i) := \big(\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)\big)^-$ for the reverse polar set of $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ shifted by $(x_{a(n)}^i, \theta_n^i)$. Using Remark 3.3, it can be reformulated.

Lemma 3.10. *The reverse polar set $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ can be expressed as*

$$\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i) = \left\{ (\gamma_n, \gamma_{n0}) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R} \ : \ \exists \mu_n \geq 0 \ : \ \begin{array}{l} \gamma_{n0} \leq 0 \\ -\tilde{A}_n^\top \mu_n - \gamma_{n0} c_n = 0 \\ -\tilde{B}_n^\top \mu_n - \gamma_n = 0 \\ \tilde{d}_n^\top \mu_n + \gamma_n^\top x_{a(n)}^i + \gamma_{n0} \theta_n^i \geq 1 \end{array} \right\}.$$

We provide a proof in Appendix B.2.

Remark 3.11. *Even with the above reformulation of $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$, an explicit formulation is usually not readily available due to the existence quantor and due to $\tilde{A}_n, \tilde{B}_n$ and $\tilde{d}_n$ not being known.*

Based on the existing work for BD, in the next sections, we present and investigate different strategies to generate Lagrangian cuts satisfying the above quality criteria. In the light of Remark 3.11, $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ may not be used without further ado to generate such cuts computationally. Still, it proves useful in the derivation of Lagrangian cuts with favorable properties. In particular, as we shall see, optimizing over $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ is closely linked to solving normalized Lagrangian dual problems. So in fact, our two perspectives to approach cut selection are intertwined and boil down to considering specific (bounded) normalizations of problem (10).

We first define the normalized Lagrangian dual in a general form and state some important properties. For simplicity, from now on, we assume that Assumptions 1 and 2

Table 2: Examination of normalized cut generation problems and different perspectives on it in the literature.

	Norm normalization			Linear normalization		
Paper	Prim	Proj	RP	Prim	Proj	RP
Cornuéjols and Lemaréchal [13]	✓	✓		✓	✓	✓
Cadoux [10]	✓	✓	✓			
Fischetti et al. [17]				(✓)		
Brandenberg and Stursberg [9]				✓		✓
Hosseini and Turner [21]	✓	✓		✓		
Chen and Luedtke [11]	(✓*)					
Seo et al. [26]					✓	
This paper	✓*	✓*	✓*	✓*	✓*	✓*

Prim: Primal perspective. Proj: Projection perspective. RP: Reverse polar perspective.
(✓): Perspective is applied, but not further explored. ✓*: Examination for Lagrangian cuts.

are satisfied, even if not explicitly stated.

Definition 3.12. *For some homogeneous normalization function $g_n : \mathbb{R}^{d_n} \times \mathbb{R}^+ \to \mathbb{R}$, the normalized Lagrangian dual is defined as*

$$\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) := \max_{\pi_n, \pi_{n0}} \Big\{ \mathscr{L}_n^{i+1}(\pi_n, \pi_{n0}) - \pi_n^\top x_{a(n)}^i - \pi_{n0}\theta_n^i \;: \\ g_n(\pi_n, \pi_{n0}) \leq 1, \pi_{n0} \geq 0 \Big\}. \tag{14}$$

Remark 3.13. *As long as the normalization constraint $g_n(\pi_n, \pi_{n0}) \leq 1$ is satisfied by some neighborhood N of the origin, we do not exclude any potential cuts due to the scaling property of π_{n0}, see Remark 3.8. In fact, Chen and Luedtke [11] prove that restricting the dual multipliers to $N \cap (\mathbb{R}^{d_{a(n)}} \times \mathbb{R}^+)$ yields a family of possible Lagrangian cuts that is satisfied by the same set of points $(x_{a(n)}, \theta_n)$ as the family of classical Lagrangian (optimality and feasibility) cuts from Appendix A.*

Lemma 3.14. *If $(x_{a(n)}^i, \theta_n^i) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$, then $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) = 0$, and vice versa.*

We provide a proof in Appendix B.3. Lemma 3.14 allows us to solely focus on the case where $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ for the remainder of this section.

In the next two subsections, we consider two different types of normalization: by norm constraints and by linear constraints. As we show, both types of normalization can be viewed from three different perspectives (a primal perspective, a projection perspective and a reverse polar perspective). These perspectives have been analyzed in the literature before, as shown in Table 2, but have not been linked all together in a generalized framework and have not been applied to Lagrangian cuts.

3.5 Normalization by Norm and Deep Cuts

We consider the normalized Lagrangian dual (14) and the associated Lagrangian cuts if some norm is used as the normalization function. We start by formally defining this type of cut.

Definition 3.15. *Let $\|\cdot\|$ be some arbitrary norm. The Lagrangian cut* (13) *defined by the solution (π_n, π_{n0}) to the normalized Lagrangian dual* (14) *with $g_n(\pi_n, \pi_{n0}) = \|\pi_n, \pi_{n0}\|$ is called* $\|\cdot\|$-deep Lagrangian cut*. For ℓ^p-norms we may also use the term* ℓ^p-deep Lagrangian cuts*.*

If appropriate norms are used, *e.g.*, the ℓ^1-norm or the ℓ^∞-norm, then the normalization can be expressed by linear constraints. However, due to the nonlinearity of $\mathscr{L}_n^{i+1}(\cdot)$, the cut generation subproblem is still not an LP, and therefore may be difficult to solve.

For the special choice of the ℓ^1-norm, this normalization is used by Chen and Luedtke [11] to generate Lagrangian cuts in two-stage stochastic programs, however without discussing the conceptual idea behind it in detail. Deep cuts allow for three theoretical and geometrical interpretations (cf. Table 2), which also explain why they are called *deep*. As the existing results from the literature can be applied to the multistage and Lagrangian setting in a straightforward way, we do not provide proofs here.

(1) **Maximizing cut depth.** Deep cuts maximize the distance between the incumbent $(x_{a(n)}^i, \theta_n^i)$ and the hyperplane associated with this cut in the dual norm $\|\cdot\|_*$ of $\|\cdot\|$, which means that they cut as deep as possible into the suboptimal region. Therefore, this distance can be interpreted as the *depth* or a scaled violation of this cut.

Lemma 3.16 (based on Hosseini and Turner [21])**.** *Let $\|\cdot\|$ be some norm and $\|\cdot\|_*$ its dual norm. Further, let $d_n\big((x_{a(n)}^i, \theta_n^i); (\pi_n, \pi_{n0})\big)$ denote the distance between the hyperplane defined by (π_n, π_{n0}) and the point $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ measured in $\|\cdot\|_*$. Then, the optimal value $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i)$ of problem* (14) *with $g_n(\pi_n, \pi_{n0}) = \|\pi_n, \pi_{n0}\|$ equals*

$$\max_{\pi_n, \pi_{n0}} \Big\{ d_n\big((x_{a(n)}^i, \theta_n^i); (\pi_n, \pi_{n0})\big) \ : \ \pi_{n0} \geq 0 \Big\}.$$

(2) **Projection onto the epigraph.** From a primal perspective, generating $\|\cdot\|$-deep cuts is in some sense equivalent to minimizing the distance in $\|\cdot\|_*$ between the incumbent $(x_{a(n)}^i, \theta_n^i)$ and the epigraph $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$, *i.e.*, related to projecting $(x_{a(n)}^i, \theta_n^i)$ onto the epigraph.

Lemma 3.17 (based on Lemma 2.5 in [10])**.** *Let $\|\cdot\|$ be some norm and $\|\cdot\|_*$ its dual norm. Then, the optimal value $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i)$ of problem* (14) *with $g_n(\pi_n, \pi_{n0}) = \|\pi_n, \pi_{n0}\|$ equals that of the projection problem*

$$\min_{x_{a(n)}, \theta_n} \Big\{ \|x_{a(n)} - x_{a(n)}^i, \theta_n - \theta_n^i\|_* \ : \ (x_{a(n)}, \theta_n) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) \Big\}. \qquad (15)$$

Lemma 3.17 implies that $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) > 0$ if $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$, whereas $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) = 0$ if not, so it confirms Lemma 3.14. Therefore, as for the non-normalized case, we have a unique flag for cases where no separating cut has to be constructed. However, in contrast to the non-normalized case, the dual problem is bounded.

We can also conclude from Lemma 3.17 that a deep cut supports $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$.

Corollary 3.18 (based on Proposition 3 in Hosseini and Turner [21])**.** *Suppose $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ and let $(\widehat{x}_{a(n)}, \widehat{\theta}_{a(n)}) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ be a solution* *to* (15)*. Then, any* $\|\cdot\|$*-deep cut separating* $(x^i_{a(n)}, \theta^i_n)$ *from* $\text{epi}(\overline{\text{co}}(\underline{Q}^{i+1}_n))$ *supports* $\text{epi}(\overline{\text{co}}(\underline{Q}^{i+1}_n))$ *at* $(\widehat{x}_{a(n)}, \widehat{\theta}_{a(n)})$*.*

(3) **Minimizing a norm over the reverse polar set.** Interestingly, deep cuts allow for another geometric interpretation that is related to the reverse polar set $\mathcal{R}^{i+1}_n(x^i_{a(n)}, \theta^i_n)$. It is based on the observation that $\mathcal{R}^{i+1}_n(x^i_{a(n)}, \theta^i_n)$ is directly linked to the normals of separating hyperplanes.

Lemma 3.19 (Lemma 2.9 in [10])**.** *Let* (π^i_n, π^i_{n0}) *be the coefficients of a* $\|\cdot\|$*-deep cut constructed at* $(x^i_{a(n)}, \theta^i_n) \notin \text{epi}(\overline{\text{co}}(\underline{Q}^{i+1}_n))$*. Then there exists some* $\alpha > 0$ *such that* $-\alpha(\pi^i_n, \pi^i_{n0})$ *minimizes* $\|\cdot\|$ *over the reverse polar set* $\mathcal{R}^{i+1}_n(x^i_{a(n)}, \theta^i_n)$*.*

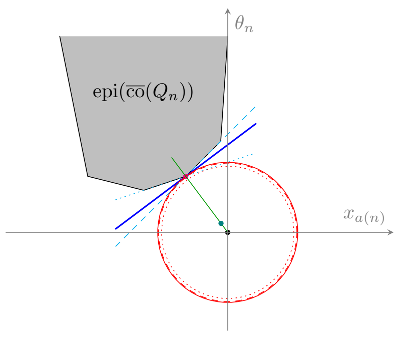
The main idea to prove this result is that due to positive homogeneity of norms, the norm $\|\cdot\|$ in the normalization constraint and the Lagrangian dual objective can be swapped.

To illustrate these three perspectives for different norms we provide an example, inspired by illustrations from the literature [9, 21].

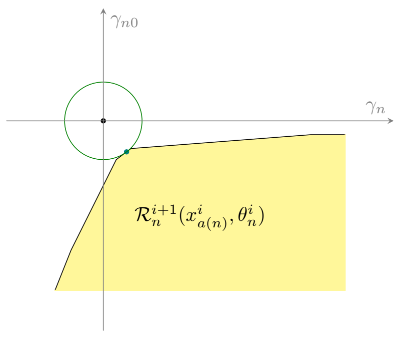
Example 3.20. *We consider a given epigraph* $\text{epi}(\overline{\text{co}}(Q_n))$*, an incumbent* $(x^i_{a(n)}, \theta^i_n)$*, the associated reverse polar set* $\mathcal{R}^{i+1}_n(x^i_{a(n)}, \theta^i_n)$ *and the obtained deep Lagrangian cuts for different norms (*ℓ^2*,* ℓ^1*,* ℓ^∞ *and a weighted* ℓ^1*-norm). The sets and cuts are illustrated in Fig. 2-5 for the different norms. In each case, the illustration consists of two parts (a) and (b). In part (a), the incumbent (*black dot*) and the epigraph are depicted. Moreover, several norm balls are shown for the respective dual norms (*red lines*). We can see that the obtained deep Lagrangian cuts (*blue lines*) maximize the distance between the incumbent and the hyperplane in the dual norm. This is illustrated by depicting different valid cuts (*dashed/dotted cyan lines*) with smaller distances. On the other hand, it is also shown that the deep cuts minimize the distances between the incumbent and the epigraph in the dual norm and support the epigraphs at the corresponding projection to the epigraph (*violet line or point*). In part (b), the reverse polar set is depicted. Moreover, the optimal solutions (*teal line or point*) for optimizing the given norm (illustrated by a norm ball,* green line*) over the reverse polar set are highlighted. These solutions (apart from sign changes) characterize the normal vectors of the obtained cuts (see Lemma 3.19), as is additionally illustrated in part (a). Note that for none of the considered cases, the deep cuts are tight for* $\text{epi}(\overline{\text{co}}(Q_n))$ *at* $x^i_{a(n)}$*, contrary to classical Lagrangian cuts.*

Example 3.20 also illustrates possible properties of deep cuts. Whereas deep cuts can be unique (see Fig. 2,3,4), also degenerate solutions with infinitely many different deep cuts are possible (see Fig. 5). This is the case if the optimization over $\mathcal{R}^{i+1}_n(x^i_{a(n)}, \theta^i_n)$ in Lemma 3.19 does not have a unique solution. Only for the ℓ^2-norm a unique solution is guaranteed. While non-unique solutions are not disadvantageous in general, degeneracy of the Lagrangian dual (14) may lead to selection of non-dominant cuts, compare Sect. 1.1.

Further, even if unique, deep cuts are not guaranteed to be facet-defining (see Fig. 2). In fact, while they cut deep into the suboptimal region, analyses in [10] (for the ℓ^2-norm) and in Hosseini and Turner [21] (for the ℓ^1-norm) show that they tend to be flat, at least in early stages of the algorithm when the optimality gap is still large. We can

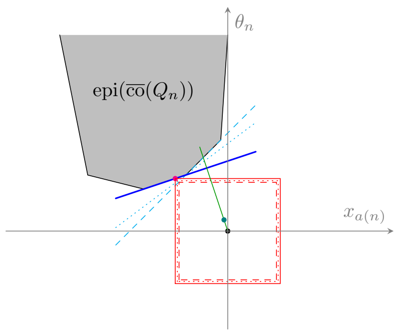


(a) Epigraph and deep cut.

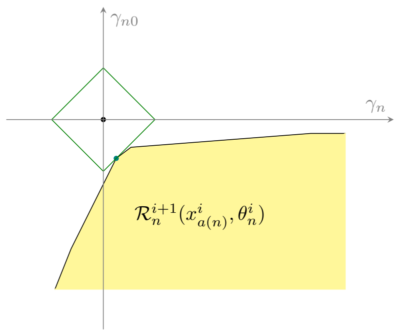


(b) Reverse polar set.

Figure 2: Illustration of deep cuts for the ℓ^2-norm.



(a) Epigraph and deep cut.



(b) Reverse polar set.

Figure 3: Illustration of deep cuts for the ℓ^1-norm.

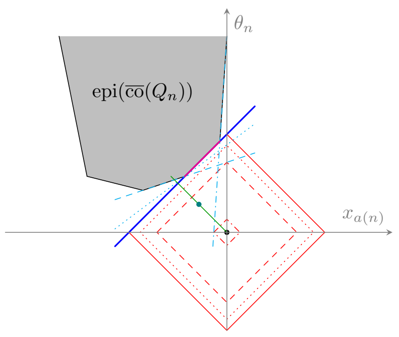
anticipate from Fig. 3 and Lemma 3.19, though, that deep cuts are facet-defining if $\|\cdot\|_*$ takes its minimum over $\mathcal{R}_n(x^i_{a(n)}, \theta^i_n)$ in a vertex.

Finally, also the projection problem (15) is not guaranteed to have a unique solution for all but the ℓ^2-norm (see Fig. 4). If the solution is non-unique, then the associated deep cut is unique and facet-defining.

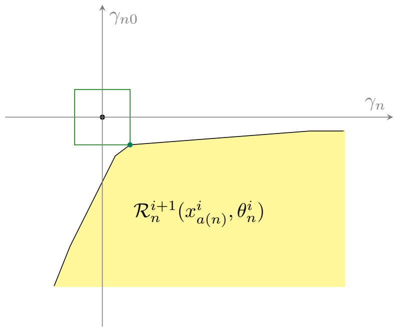
Remark 3.21. *Another intuitive way to bound the Lagrangian dual* (10) *and to select cut coefficients* (π_n, π_{n0}) *is to introduce simple box constraints for the multipliers. For symmetric boxes around the origin, which can be modeled by a weighted* ℓ^∞*-norm, we may interpret the obtained cuts as deep cuts. However, which specific cuts are selected highly depends on the chosen multiplier bounds. Too large bounds may favor degeneracy and very steep cuts, for too small bounds, only almost horizontal cuts can be selected.*

3.6 Linear Normalization

We consider the normalized Lagrangian dual (14) with a linear normalization function $g_n(\pi_n, \pi_{n0}) = u_n^\top \pi_n + u_{n0}\pi_{n0}$ defined by some coefficients $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \in \mathbb{R}$. Recall that the initial Lagrangian dual problem (10) is unbounded and that we introduce the normalization constraint in (14) in order to transform the problem to a bounded one to identify unbounded rays. In contrast to the norm-based normalization from Sect. 3.5,

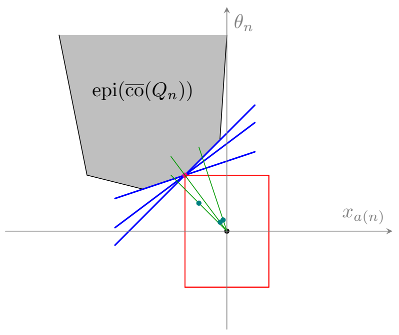


(a) Epigraph and deep cut.

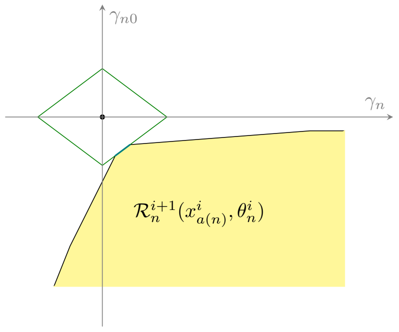


(b) Reverse polar set.

Figure 4: Illustration of deep cuts for the ℓ^∞-norm.



(a) Epigraph and deep cut.



(b) Reverse polar set.

Figure 5: Illustration of deep cuts for a weighted ℓ^1-norm.

a linear normalization does not guarantee boundedness, though. Hence, the choice of (u_n, u_{n0}) is crucial to ensure that an optimal solution exists. We further analyze this later in this section, but for now take the following assumption.

Assumption 3. *Given some* $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$*, the normalized Lagrangian dual* (14) *with* $g_n(\pi_n, \pi_{n0}) = u_n^\top \pi_n + u_{n0}\pi_{n0}$ *has a finite optimal value* $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) > 0$.

3.6.1 Linear Normalization Cuts

We can then define the associated type of Lagrangian cuts.

Definition 3.22. *Let* $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$ *and let the normalized Lagrangian dual* (14) *satisfy Assumption 3. Then, we refer to the Lagrangian cut* (13) *defined by its solution* (π_n, π_{n0}) *as a* linear normalization (LN) Lagrangian cut*.*

Again, we can take three different perspectives on LN cuts.

(1) **Pseudonorm perspective.** Hosseini and Turner [21] restrict to choices of (u_n, u_{n0}) such that $g_n(\pi_n, \pi_{n0}) = u_n^\top \pi_n + u_{n0}\pi_{n0} \geq 0$ for all $(\pi_n, \pi_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}_+$. In such a case, $g_n(\cdot)$ is a *linear pseudonorm* (in contrast to norms, it is not positive definite). This means that LN cuts can be interpreted as maximizing the distance between the associated hyperplane and $(\widehat{x}_{a(n)}^i, \widehat{\theta}_n^i)$ in a linear pseudonorm.

(2) **Projection on a line segment.** This perspective has been brought up several times in the literature in different variants, most recently by Seo et al. [26]. Even though the geometric idea is the same in the Lagrangian context, the formal description changes a bit.

First, we exploit that for linear $g_n(\cdot)$, the normalized Lagrangian dual (14) can be reformulated as an LP and then be dualized with strong duality.

Lemma 3.23. *Let $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$. The normalized Lagrangian dual* (14) *with $g_n(\pi_n, \pi_{n0}) = u_n^\top \pi_n + u_{n0}\pi_{n0}$ can be formulated as an LP, and its dual is*

$$\min_{\lambda_n, z_n, \eta_n} \Big\{ \eta_n \;:\; (\lambda_n, z_n) \in \operatorname{conv}(\mathcal{W}_n^{i+1}), \quad \eta_n \geq 0,\; u_{n0}\eta_n \geq c_n^\top \lambda_n - \theta_n^i, \\ u_n \eta_n = z_n - x_{a(n)}^i \Big\}. \tag{16}$$

We provide a proof in Appendix B.4. Problem (16) can be interpreted as finding the smallest scaling factor $\eta_n \geq 0$ such that starting from $(x_{a(n)}^i, \theta_n^i)$ along direction (u_n, u_{n0}) a point in $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ is reached. Again, for the optimal value it follows $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) = \eta_n^* > 0$ if and only if $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$. Given the optimal value, the projection of $(x_{a(n)}^i, \theta_n^i)$ onto $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ along direction (u_n, u_{n0}) can be determined as

$$(\widehat{x}_{a(n)}, \widehat{\theta}_{a(n)}) = (x_{a(n)}^i, \theta_n^i) + \eta_n^*(u_n, u_{n0}). \tag{17}$$

We then obtain the following result:

Corollary 3.24. *Let $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$ and let the normalized Lagrangian dual* (14) *satisfy Assumption 3. Furthermore, let $(\widehat{x}_{a(n)}, \widehat{\theta}_{a(n)}) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ satisfy* (17)*. Then, the associated LN cut supports $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ at $(\widehat{x}_{a(n)}, \widehat{\theta}_{a(n)})$.*

(3) **Maximizing a linear function over the reverse polar set.** Again, an interpretation with respect to $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ is possible based on its characterization of normal vectors of separating hyperplanes. More precisely, LN cuts can be obtained by maximizing the linear objective function $u_n^\top \gamma_n + u_{n0}\gamma_{n0}$ over $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$:

$$\max_{\gamma_n, \gamma_{n0}} \Big\{ u_n^\top \gamma_n + u_{n0}\gamma_{n0} \;:\; (\gamma_n, \gamma_{n0}) \in \mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i) \Big\}. \tag{18}$$

This perspective is discussed in detail in Brandenberg and Stursberg [9] for BD and in [13] for general convex sets (where solving problem (18) is shown to correspond to evaluating a function called *reverse gauge*). For the relation to the normalized Lagrangian dual (14), the following result holds. For a sketch of the proof, see Appendix B.5.

Theorem 3.25 (based on [9])**.** *Let $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$ and $(x_{a(n)}^i, \theta_n^i) \notin \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$.*

(i) If the normalized Lagrangian dual (14) *satisfies Assumption 3, then problem* (18) *has a finite optimal value, and vice versa. The optimal points are the same up to scaling with some negative scalar and the optimal values multiply to -1.*

(ii) The induced cuts for $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ are equivalent for both problems.

Geometrically, a favorable property of generating cuts based on $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ is that supporting cuts for $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ can be obtained in a straightforward way. Even more, if $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ is a full-dimensional polyhedron, then each vertex (γ_n, γ_{n0}) of $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ corresponds to the normal vector of a facet of $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$, and vice versa [9]. In order to identify such points, we can use the LP (18), given a choice of $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$ such that a finite optimum is attained.

In fact, perspectives (2) and (3) make a sufficient condition for such a finite optimum readily available, and by that also allow us to conclude when Assumption 3 is satisfied. This result is already proven in [13] and [9] using perspective (3), and then follows for the normalized Lagrangian dual (14) with Theorem 3.25. In Appendix B.6, we provide an alternative proof based on perspective (2).

Lemma 3.26. *Assumption 3 is satisfied if*

$$(u_n, u_{n0}) \in \operatorname{cone}\left(\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)\right) \setminus \{0\}, \tag{19}$$

where $\operatorname{cone}(S)$ *denotes the conical hull of a set* S.

Lemma 3.26 implies that also choosing (u_n, u_{n0}) from $\operatorname{epi}(\underline{Q}_n^{i+1}) - (x_{a(n)}^i, \theta_n^i)$ or even from $\operatorname{epi}(Q_n) - (x_{a(n)}^i, \theta_n^i)$ is sufficient. In other words, choosing reasonable coefficients $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$ boils down to finding a *core point* within one of these epigraphs. We discuss this in more detail in Sect. 3.6.2.

We can now address some beneficial properties of LN cuts with respect to the aforementioned cut quality criteria. According to Theorem 3.25 there exists a one-to-one relation between optimal solutions of the normalized Lagrangian dual (14) and problem (18). However, this does not extend to optimal vertices, leading to a slightly less strong result for facet-defining cuts with respect to problem (14) [9]. We obtain the following properties:

Theorem 3.27. *Let* $(u_n, u_{n0}) \in \mathbb{R}^{d_n} \times \mathbb{R}$. *Consider the normalized Lagrangian dual problem* (14) *with* $g_n(\pi_n, \pi_{n0}) = u_n^\top \pi_n + u_{n0}\pi_{n0}$. *Then,*

(i) for all $(u_n, u_{n0}) \in \operatorname{cone}\left(\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)\right) \setminus \{0\}$, *the optimal point* (π_n^*, π_{n0}^*) *defines a supporting cut for* $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$,

(ii) for all $(u_n, u_{n0}) \in \operatorname{cone}\left(\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)\right) \setminus \{0\}$, *there exists an optimal extreme point* (π_n^*, π_{n0}^*), *such that the obtained cut is facet-defining for* $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$,

(iii) for all $(u_n, u_{n0}) \in \operatorname{relint}\left(\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)\right)$, *any optimal point* (π_n^*, π_{n0}^*) *with* $\pi_{n0}^* > 0$ *defines a Pareto-optimal cut for* $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$ *on* $\operatorname{conv}(Z_{a(n)})$.

Part (i) directly follows from Lemma 3.26 and Corollary 3.24. Part (ii) follows from Theorem 3.3 in [9], and part (iii) follows from Theorem 3.43 in [28]. Note that the results from the literature require to choose (u_n, u_{n0}) from the relative interior of the epigraph restricted to $\operatorname{conv}(X_{a(n)})$. However, under Assumption 1, if we choose $Z_n = X_{a(n)}$ or $Z_n = \operatorname{conv}(X_{a(n)})$, in our case the considered epigraphs are always restricted to this set.

Considering part (ii), the only case in which no facet-defining cut is obtained occurs if the optimal solution to the normalized Lagrangian dual (14) is not unique. However, this is only the case for a small subset of choices (u_n, u_{n0}). Especially if the choice is adapted in each iteration, the occurrences of such cases should be negligible [9]. With respect to (iii), we should emphasize again that Pareto-optimality for $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ does not necessarily imply Pareto-optimality for $\text{epi}(\underline{Q}_n^{i+1})$.

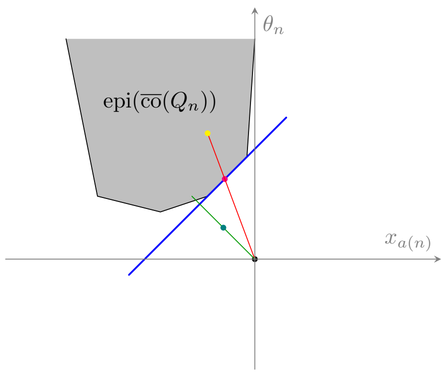
We finish our theoretical results in this subsection with two remarks.

Remark 3.28. *Based on the perspective taken, LN cuts are also called* pseudo-deep cuts *(perspective (1), Hosseini and Turner [21]) or* closest cuts *(perspective (2), Seo et al. [26]) in the literature. We could also refer to them as* core point cuts*, or based on perspective (3) as* reverse gauge cuts.

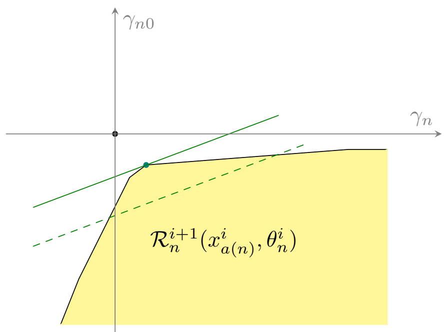
Remark 3.29. *Choosing $(u_n, u_{n0}) = (0, 1)$ in Definition 3.22 yields the classical Lagrangian cuts presented in Appendix A.*

We illustrate the different perspectives on LN cuts again using an example.

Example 3.30. *Consider epigraph $\text{epi}(\overline{\text{co}}(Q_n))$, incumbent $(x^i_{a(n)}, \theta^i_n)$ and reverse polar set $\mathcal{R}_n^{i+1}(x^i_{a(n)}, \theta^i_n)$ from Example 3.30. These objects and an exemplary LN cut are illustrated in Fig. 6. In part (a) the geometric idea of projection along a line segment is highlighted. The direction of this line segment is (u_n, u_{n0}) and obtained as the difference between a known core point (*yellow dot*) in $\text{epi}(\overline{\text{co}}(Q_n))$ and $(x^i_{a(n)}, \theta^i_n)$. In part (b) we can see that (apart from sign changes) the cut normal to the LN cut can be determined by maximizing the linear function $u_n^\top \gamma_n + u_{n0}\gamma_{n0}$ over $\mathcal{R}_n^{i+1}(x^i_{a(n)}, \theta^i_n)$. As the solution is an extreme point of $\mathcal{R}_n^{i+1}(x^i_{a(n)}, \theta^i_n)$ (*green dot*), the corresponding LN cut is facet-defining.*



(a) Epigraph and deep cut.



(b) Reverse polar set.

Figure 6: Illustration of LN cuts given some known core point.

3.6.2 Identifying Core Points

As described before, a key challenge of generating LN cuts with favorable properties is to choose (u_n, u_{n0}) appropriately, *i.e.*, according to Lemma 3.26 or Theorem 3.27. In the literature on decomposition methods, it is often proposed to evaluate feasible points in the objective function to obtain core points. Whereas this approach is straightforward for BD or the two-stage stochastic case [9, 28], in the multistage case, evaluating $Q_n(\cdot)$ exactly is computationally prohibitive in general. Furthermore, note that we also cannot evaluate $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$ efficiently, as this function is not known explicitly and evaluating it requires to solve a Lagrangian dual problem. Therefore, we propose heuristics to identify potential core points $(\widehat{x}^i_{a(n)}, \widehat{\theta}^i_n)$, and by that reasonable coefficients (u_n, u_{n0}), based on function $\underline{Q}^{i+1}_n(\cdot)$. This comes with some considerable challenges, as we shall see. Some of these heuristics are particularly suited to SDDiP where $X_{a(n)} = \{0,1\}^{d_{a(n)}}$.

We consider the following approaches:

- `Mid.` We set $\widehat{x}^i_{a(n)} = \text{mid}(\text{conv}(X_{a(n)}))$ where mid denotes the midpoint (we assume box constraints for $x_{a(n)}$). The idea is that incentivizing the LN cuts to support $\text{epi}(\overline{\text{co}}(\underline{Q}^{i+1}_n)$ in the interior of the state space may be useful to avoid the degeneracy issues for SDDiP discussed in Sect. 1.1. We then evaluate the approximate value function to obtain $\widehat{\theta}^i_n = \underline{Q}^{i+1}_n(\widehat{x}^i_{a(n)})$, even if this does not satisfy the relative interior requirement in Theorem 3.27.
- `In-Out.` We use the dynamic approach to compute core points proposed by Papadakos [23] heuristically, that is, we set $\widehat{x}^i_{a(n)} = \frac{1}{2}\widehat{x}^{i-1}_{a(n)} + \frac{1}{2}x^i_{a(n)}$ and $\widehat{\theta}^i_n = \underline{Q}^{i+1}_n(\widehat{x}^i_{a(n)})$ to obtain a candidate core point.
- `Eps.` For some $\varepsilon > 0$, we use an ε-perturbation of $x^i_{a(n)}$ into the interior of $\text{conv}(X_{a(n)})$, and set $\widehat{\theta}^i_n = \underline{Q}^{i+1}_n(\widehat{x}^i_{a(n)})$. This idea is inspired by the perturbation strategy described in Sherali and Lunday [27]. A similar approach is also used in Seo et al. [26]. It is particularly suited to SDDiP, as it avoids the possible degeneracy issues related to generating cuts at extreme points of the state space, see Sect. 1.1, and thus may contribute to identifying facet-defining cuts. For sufficiently small ε, the LN cut should still be supporting $\text{epi}(\overline{\text{co}}(\underline{Q}^{i+1}_n))$ at $(x^i_{a(n)}, \overline{\text{co}}(\underline{Q}^{i+1}_n)(x^i_{a(n)}))$.
- `Relint.` We solve an auxiliary feasibility problem with slack variables to find a potential core point in relint $\big(\text{epi}(\underline{Q}^{i+1}_n)\big)$. This feasibility problem is defined in a similar way to the one described in Sect. 5.1 of Conforti and Wolsey [12].

These heuristics are illustrated in Fig. 7 for a simple example, where all of them are sufficient to get reasonable directions (u_n, u_{n0}), but lead to very different cuts being selected.

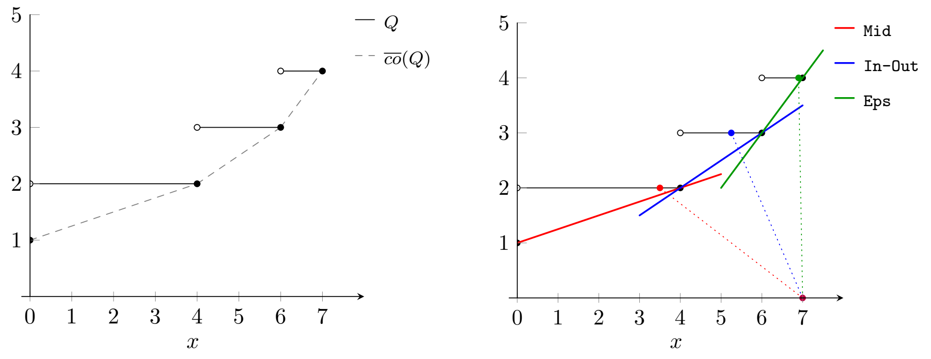


Figure 7: Illustration of core point identification heuristics.

Note. LEFT: Value function $Q(\cdot)$ and convex envelope $\overline{\text{co}}(Q)(\cdot)$ for an example with continuous state space $X = [0, 7]$. RIGHT: Identification of different core points and computation of the associated LN cuts for incumbent $(\bar{x}, \bar{\theta}) = (7, 0)$. For `In-Out` we assume $\widehat{x}^{i-1} = \frac{7}{2}$.

Despite their straightforwardness, the aforementioned approaches come with some notable challenges. Crucially, the first three approaches yield candidates satisfying $\widehat{x}^i_{a(n)} \in \operatorname{conv}(X_{a(n)})$. However, this choice is not necessarily feasible if integrality constraints are present, thus leading to $\underline{Q}^{i+1}_n(\widehat{x}^i_{a(n)}) = +\infty$ (especially if $Z_{a(n)} = X_{a(n)}$). We thus do not obtain a core point or a reasonable choice of (u_n, u_{n0}). This is illustrated in Fig. 8. Instead of evaluating $\underline{Q}^{i+1}_n(\cdot)$, in such a case we may revert to the value function $\underline{Q}^{LP,i+1}_n(\cdot)$ of the associated LP relaxation (`LP-value`). While this may yield a sufficient direction (u_n, u_{n0}), it may also yield one that does not point into $\overline{\operatorname{co}}(\underline{Q}^{i+1}_n)(\cdot)$, see Fig. 8. To mitigate this risk, we may alternatively utilize the current state θ^i_n (`epi-state`) or the primal objective value $\underline{Q}^{i+1}_n(x^i_{a(n)})$ at the incumbent (`primal-obj`) if they exceed $\underline{Q}^{LP,i+1}_n(\widehat{x}^i_{a(n)})$, see again Fig. 8. All these approaches have no guarantees to yield true core points, though.

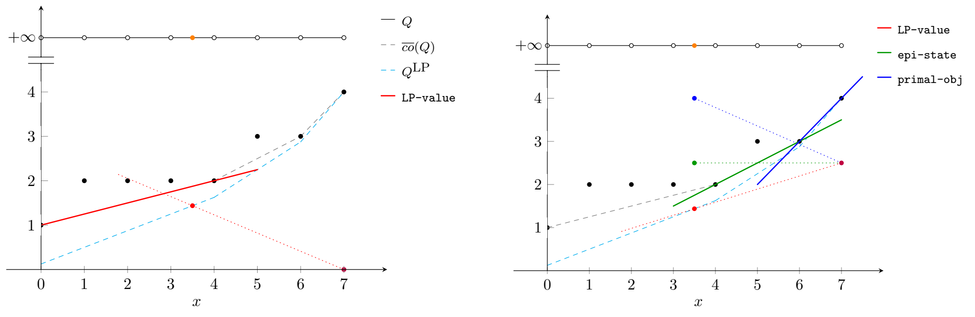


Figure 8: Core point identification challenges for integer state space.

Note. LEFT: Value function $Q(\cdot)$ and convex envelope $\overline{\operatorname{co}}(Q)(\cdot)$ for an example with integer state space $X = \{0, 1, 2, \ldots, 7\}$. Given incumbent $(\bar{x}, \bar{\theta}) = (7, 0)$, using the LP relaxation value leads to a sufficient core point candidate and generation of an LN cut. RIGHT: The approach `LP-value` is not sufficient given incumbent $(\bar{x}, \bar{\theta}) = (7, \frac{5}{2})$. A sufficient core point candidate for generation of an LN cut is obtained using `epi-state` or `primal-obj`, though.

Even with the proposed heuristics we may end up with a normalized Lagrangian dual problem (14) that is unbounded. This is unintended, because the decomposition method may terminate with an error. Therefore, we should check for a potential unboundedness in practice, and if it is detected, take some special counter-measures. For instance, we could try another heuristic, generate a different type of cut instead of an LN cut, *e.g.*, a strengthened Benders cuts, or artificially bound the normalized Lagrangian dual problem (14).

Problem (16) provides a natural way to check for unboundedness, or the validity of direction (u_n, u_{n0}), respectively. However, it cannot be solved immediately, as we do not know $\operatorname{conv}(\mathcal{W}^{i+1}_n)$ explicitly. We may instead solve the approximation

$$\min_{\lambda_n, z_n, \eta_n} \left\{ \eta_n \;:\; (\lambda_n, z_n) \in \mathcal{W}^{i+1}_n,\ \eta_n \geq 0,\ u_{n0}\eta_n \geq c_n^\top \lambda_n - \theta^i_n,\ u_n \eta_n = z_n - x^i_{a(n)} \right\} \tag{20}$$

using the known set $\mathcal{W}^{i+1}_n$ instead of $\operatorname{conv}(\mathcal{W}^{i+1}_n)$. If the normalized Lagrangian dual (14) is unbounded, then this problem is infeasible. Therefore, we can use infeasibility of (20) as a proxy for possible unboundedness. Unfortunately, in the presence of integrality constraints, infeasibility of (20) may occur very often, even given an appropriate direction (u_n, u_{n0}), thus leading to taking the previously discussed counter-measures more often than required. On the other hand, using the LP relaxation of (20) is not sufficient to rule out all cases of unboundedness. For an effective practical implementation of LN cuts this is a significant challenge.

Finally, let us present an alternative, fifth approach to come up with core points.

- `Conv.` We note that any convex combination of two (or more) feasible points $\big(x_1, \underline{Q}_n^{i+1}(x_1)\big)$ and $\big(x_2, \underline{Q}_n^{i+1}(x_2)\big)$ is always contained in $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})$. One such point is readily available with $\big(x^i_{a(n)}, \underline{Q}_n^{i+1}(x^i_{a(n)})\big)$. For the case $X_{a(n)} = \{0,1\}^{d_{a(n)}}$, an intuitive strategy to obtain a second one is to consider the diagonal counterpart of $x^i_{a(n)}$ (swapping 0 and 1 for all components) and its function value for $\underline{Q}_n^{i+1}(\cdot)$. If this counterpart is feasible, we obtain a whole family of core points. Setting the convex combination parameter appropriately, we may even obtain a core point with first component $\operatorname{mid}(\operatorname{conv}(X_{a(n)}))$, but without having to evaluate the approximate value function in a non-integer state. This approach is highlighted in Fig. 9.

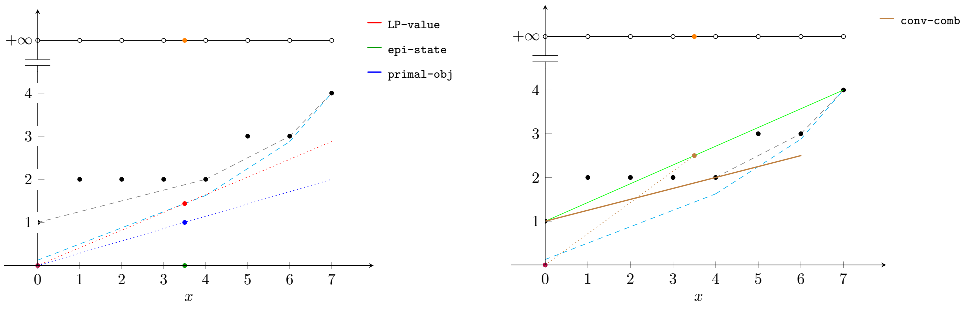


Figure 9: Illustration of core point identification heuristics.

Note. LEFT: Neither `LP-Value`, `epi-state` nor `primal-obj` are sufficient to identify an appropriate projection direction given incumbent $(\bar{x}, \bar{\theta}) = (0, 0)$. RIGHT: Choosing a core point as a convex combination of $(0, 1)$ and $(7, 4)$ (light green) with parameter $\frac{1}{2}$ and the resulting LN cut.

4 Convergence of NBD and SDDiP

For the case of binary state variables, *i.e.*, $x_n \in \{0,1\}^{d_n}$ for all $n \in \mathcal{N}$, classical Lagrangian cuts are sufficient to ensure (almost sure) finite convergence of decomposition methods such as NBD or SDDiP because these cuts are valid, tight and finite (see Appendix A). In this section, we address the convergence properties of these methods if the proposed new cut generation framework is incorporated. We restrict our discussion to NBD, that is exploring the whole scenario tree in each iteration. The considered NBD algorithm is displayed in Appendix C.

For SDDiP, only a sample of scenarios is considered in each iteration, so we have to replace line 6 of the NBD algorithm in Appendix C with a sampling step, but stagewise independence is assumed. Given the convergence of NBD, the almost sure finite convergence of SDDiP, follows with the same arguments as in the original SDDiP article [31, Theorem 2]. Moreover, instead of presenting a complete, self-contained convergence proof, we focus on the decisive properties of the Lagrangian cuts obtained in our new cut generation framework. The rest of the convergence proof remains unchanged compared to [31].

The validity of the new Lagrangian cuts is already proven in Lemma 3.7. Therefore, some results with respect to finiteness and tightness remain to be proven. An important property in that regard is the polyhedrality of the closed convex envelopes $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$.

Lemma 4.1. *For all $n \in \overline{\mathcal{N}}$ and all iterations i, $\overline{\text{co}}(Q_n)(\cdot)$ and $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$ are piecewise linear convex functions.*

Lemma 4.1 is proven in Appendix B.7. It implies that $\text{epi}(\overline{\text{co}}(Q_n))$ and $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ are polyhedra with finitely many facets. Additionally, we take a technical assumption:

Assumption 4. *In any node $n \in \overline{\mathcal{N}}$ and any iteration i of NBD, given the same trial point $(x_{a(n)}^i, \theta_n^i)$ and the same approximations Ψ_m^{i+1} for all $m \in \mathcal{C}(n)$, solving the normalized Lagrangian dual problem* (14) *yields the same cut.*

This assumption is required to avoid that infinitely many different cuts are generated given the same state and the same approximate value function. It should be satisfied by most deterministic MILP solvers.

4.1 Results for LN Cuts

For NBD using LN cuts, we can exploit that the cuts are guaranteed to be (almost always) facet-defining. If we restrict to facet-defining cuts, we obtain the following convergence result.

Theorem 4.2. *Let $X_n = \{0,1\}^{d_n}$ for all $n \in \overline{\mathcal{N}}$. Assume that for any $n \in \overline{\mathcal{N}}$ and any iteration i the normalized Lagrangian dual problem* (14) *and the generated cuts satisfy Theorem 3.27 (ii). Then, after a finite number of iterations i, for all $n \in \overline{\mathcal{N}}$ the approximations Ψ_n^{i+1} are exact for* $\text{epi}(Q_n(\cdot))$ *at all $x_n^i \in X_{a(n)}$ computed in the forward pass.*

We provide a proof in Appendix B.8. Note that the proof still works if the number of steps between the generation of facet-defining cuts is always finite.

4.2 Results for Deep Cuts

For deep Lagrangian cuts proving convergence is a bit more tedious, since the facet property is not as straightforward to assure. The main idea to prove finite convergence thus is the following: Even if deep cuts are not guaranteed to be facet-defining or tight at $x_{a(n)}^i$ in general, such tight cuts will eventually be generated after finitely many steps. This is illustrated in Fig. 10.

As a first step, we introduce an auxiliary result where we consider a sequence of trial points with fixed first component $\bar{x}_{a(n)}$. Here, we do not exploit the binary nature of the state variables yet, but only the polyhedrality of $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$. For notational convenience, we set $\bar{\theta}_n := \overline{\text{co}}(\underline{Q}_n^{i+1})(\bar{x}_{a(n)})$ for the remainder of this section. The proof is shown in Appendix B.9.

Lemma 4.3. *For any $n \in \overline{\mathcal{N}}$, consider a subsequence of trial points $(\bar{x}_{a(n)}, \theta_n^{i_\nu})_{\nu \in \mathbb{N}}$ with fixed first component $\bar{x}_{a(n)} \in X_{a(n)}$ for all $\nu \in \mathbb{N}$. Then, the point $(\bar{x}_{a(n)}, \bar{\theta}_n)$ and any sequence $(\widehat{x}_{a(n)}^{i_\nu}, \widehat{\theta}_n^{i_\nu})_{\nu \in \mathbb{N}}$ of solutions of the projection problems* (15) *satisfy $\lim_{\nu \to \infty}(\widehat{x}_{a(n)}^{i_\nu}, \widehat{\theta}_n^{i_\nu}) = (\bar{x}_{a(n)}, \bar{\theta}_n)$.*

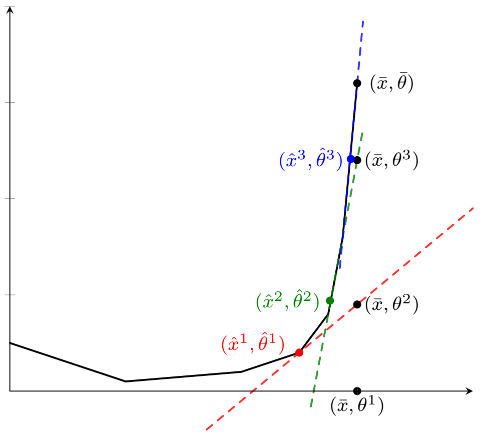

$(\bar{x}, \bar{\theta})$
$(\hat{x}^3, \hat{\theta}^3)$
$(\bar{x}, \theta^3)$
$(\hat{x}^2, \hat{\theta}^2)$
$(\bar{x}, \theta^2)$
$(\hat{x}^1, \hat{\theta}^1)$
$(\bar{x}, \theta^1)$

Figure 10: Illustration of deep cuts leading to tightness after finitely many steps for fixed $\bar{x}$. The blue cut is tight at $(\bar{x}, \bar{\theta})$, thus equal to a classical Lagrangian cut from Appendix A.

We now consider subproblems with fixed first component $\bar{x}_{a(n)}$ and with fixed approximations Ψ_m for all $m \in \mathcal{C}(n)$ and all $n \in \overline{\mathcal{N}}$ (for instance, the latter is satisfied for the leaf nodes). In this case, the set $\operatorname{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ remains constant over the iterations, so we drop the iteration index for simplicity.

Let K with $|K| \in \mathbb{N}$ denote the finite set of facets of $\operatorname{epi}(\overline{\text{co}}(\underline{Q}_n))$. We use the symbol F to refer to specific facets. Let $\overline{K} \subseteq K$ denote the subset of facets in which $(\bar{x}_{a(n)}, \bar{\theta}_n)$ is contained, *i.e.*, $(\bar{x}_{a(n)}, \bar{\theta}_n) \in F_k$ for all $k \in \overline{K}$. Analogously, let $\widehat{K}^\nu \subseteq K$ denote the subset of facets in which $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ is contained. Based on Lemma 4.3, we obtain the following result, which is proven in Appendix B.10.

Lemma 4.4. *There exists some $\widehat{\nu} \in \mathbb{N}$ such that for all $\nu \geq \widehat{\nu}$ we have $\widehat{K}^\nu \subseteq \overline{K}$.*

As $\widehat{K}^\nu \neq \emptyset$ by definition, this implies that for sufficiently large ν, the points $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ and $(\bar{x}_{a(n)}, \bar{\theta}_n)$ are located on a joint facet. Due to the convergence result in Lemma 4.3, this is intuitively clear. The case $\widehat{K}^\nu \not\subseteq \overline{K}$ is excluded by this convergence result. However, the case $\widehat{K}^\nu \subset \overline{K}$ is possible if $(\bar{x}_{a(n)}, \bar{\theta}_n)$ is located at the boundary of some facets.

We require some further auxiliary results.

Lemma 4.5. *There exists some $\bar{\nu} \in \mathbb{N}$ such that for all $\nu \geq \bar{\nu}$ the point $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ is equal to $(\bar{x}_{a(n)}, \bar{\theta}_n)$ or not a vertex of* $\operatorname{epi}(\overline{\text{co}}(\underline{Q}_n))$.

This result follows immediately from Lemma 4.3. For sufficiently large ν, $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ may only be a vertex if $(\bar{x}_{a(n)}, \bar{\theta}_n)$ is a vertex and if they are equal.

Lemma 4.6. *Consider a convex polyhedron S. Let $x^1, x^2 \in S$ be two points located on a joint face (not necessarily a facet) $\mathfrak{F}$ of S with $x^2 \in$* $\operatorname{relint}(\mathfrak{F})$. *Then, a cut supporting S at x^2 is also supporting S at x^1.*

We provide a proof in Appendix B.11. We are now able to prove our first main result, stating that for sufficiently large ν a deep Lagrangian cut supporting $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ will also support $(\bar{x}_{a(n)}, \bar{\theta}_n)$.

Lemma 4.7. *For any $\overline{\mathcal{N}}$, consider subproblem* (6) *with fixed approximations Ψ_m for all $m \in \mathcal{C}(n)$. Moreover, consider a subsequence of trial points $(\bar{x}_{a(n)}, \theta_n^{i_\nu})_{\nu \in \mathbb{N}}$ with fixed first component $\bar{x}_{a(n)} \in X_{a(n)}$ for all $\nu \in \mathbb{N}$. Then, there exists some $\check{\nu} \in \mathbb{N}$ such that for all $\nu \geq \check{\nu}$, a deep Lagrangian cut as defined in Definition 3.15 supports* $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n))$ *at $(\bar{x}_{a(n)}, \bar{\theta}_n)$.*

We provide a proof in Appendix B.12. We can now turn to properties of the cuts generated within NBD. Here, we exploit that $x_n \in \{0,1\}^{d_n}$ for all $n \in \mathcal{N}$.

Theorem 4.8. *Let $X_n = \{0,1\}^{d_n}$ for all $n \in \mathcal{N}$, and let Assumption 4 be satisfied. Then, after a finite number of iterations i, for all $n \in \overline{\mathcal{N}}$ the approximations Ψ_n^{i+1} are exact for* $\operatorname{epi}(Q_n(\cdot))$ *at all $x_n^i \in X_{a(n)}$ computed in the forward pass.*

A proof is presented in Appendix B.13.

4.3 Convergence Result

Based on the results in Theorem 4.2 or Theorem 4.8, respectively, finite convergence of NBD can be concluded. For more details, see [31].

Corollary 4.9. *Let $X_n = \{0,1\}^{d_n}$ for all $n \in \mathcal{N}$, and let Assumption 4 be satisfied. Assume that NBD is applied with generation of deep Lagrangian cuts or LN Lagrangian cuts satisfying Theorem 3.27 (ii). Then, NBD terminates with an optimal policy for (MS-MILP) after a finite number of iterations.*

We should emphasize again that this result only holds because of the finiteness and the binary character of X_n, so it is not necessarily true for general continuous state spaces. Furthermore, in the above proofs we use the idea that for any $x_{a(n)} \in X_{a(n)}$ after finitely many steps the deep or LN Lagrangian cuts will coincide with the original Lagrangian cuts, *i.e.* become *tight* in the sense of [31]. While this seems to imply more iterations than classical NBD (or SDDiP), the vision is that in practice convergence may actually be achieved faster.

5 Computational Experiments

We report results for a computational study of SDDiP using the proposed cut generation framework. For comparison, we also run tests using established cut generation techniques in SDDiP. More precisely, we consider the following approaches to generate cuts:

- `B`: Classical Benders cuts using either a single-cut or a multi-cut approach.
- `SB`: Strengthened Benders cuts [31] using either a single-cut or a multi-cut approach.
- `L`: Classical Lagrangian cuts from Appendix A using either a single-cut or a multi-cut approach.
- $\ell^1, \ell^\infty, \ell^{1\infty}$: Deep Lagrangian cuts from Definition 3.15 for the ℓ^1-norm, the ℓ^∞-norm and a linear combination of $0.5\|\cdot\|_1 + 0.5\|\cdot\|_\infty$.

- `LN`: LN Lagrangian cuts from Definition 3.22 using the `Mid`, `In-Out`, `Eps`, `Relint` and `Conv` heuristics from Sect. 3.6.2 to identify core points and to determine the normalization coefficients. For `Eps` we use a perturbation of the incumbent by 10^{-6}.

For now, we do not test the improvement techniques `epi-state` or `primal-obj` from Sect. 3.6.2. By construction, all deep and LN cuts require using a multi-cut approach.

We test the proposed methods on different instances of a capacitated lot-sizing problem (CLSP), which is described in the appendix of Trigeiro et al. [29] and has stagewise independent uncertain demand for each product. This problem is also considered in Ahmed et al. [1] and identified to be challenging for exact decomposition methods like SDDiP.

In our experiments, we consider instances with 3 or 10 state variables, 4, 6, 10 or 16 stages and 20 realizations of the uncertain demand at each stage. For the case of 3 state variables and 20 realizations per stage, we use the exact same scenarios as in Ahmed et al. [1], for larger instances we use the same methodology to generate scenarios.

For instances with 3 state variables we apply SDDiP with a binary approximation of the continuous state variables with discretization precision of 1.0 [see 31]. This means that the modified MS-MILP which is tackled by SDDiP contains 30 state variables. For instances with 10 state variables initially, using a binary approximation would produce state dimensions which are computationally intractable for SDDiP, as its complexity grows exponentially in the dimension of the state space [30]. Therefore, in these cases, we apply SDDiP without a binary approximation.

We describe our implementation of SDDiP and our parameter choices in more detail in Sect. 5.1. Then, we discuss results for experiments of CLSP with state binarization, where we compare the above cut generation techniques when they are applied individually (Sect. 5.2) as well as combined with `SB` cuts (Sect. 5.3). In Sect. 5.4 we study a restriction of the dual space. In Sect. 5.5 we deal with larger instances of CLSP where no binary approximation is applied.

5.1 Implementation Details

SDDiP and all cut generation approaches are implemented in Julia-1.5.3 [5] based on the existing packages `SDDP.jl` [15] and `JuMP.jl` [16]. The code is available on GitHub as part of a larger project called `DynamicSDDiP.jl` (see `https://github.com/ChrisFuelOR/DynamicSDDiP.jl`).

SDDiP is terminated after a predefined time limit or if the obtained lower bounds start to stall. In each forward pass, one scenario path is randomly sampled. After termination of SDDiP, an in-sample Monte Carlo simulation with 1000 replications is conducted on the finite scenario tree to compute a statistical upper bound for the current policy.

The Lagrangian dual problems in SDDiP are solved using a level bundle method with a maximum of 1000 iterations and an optimality tolerance of 10^{-4}. The multipliers π_n are initialized with a vector of zeros and π_{n0} is initialized with 1. Sometimes the level bundle method reports infeasibilities in the quadratic auxiliary problem. In that case, we proceed with a standard Kelley step instead. Moreover, in the case of different numerical issues in solving the Lagrangian dual, the solution process is stopped and a valid cut is constructed with the current values of the multipliers.

For the LN cuts, as pointed out before, the choice of normalization coefficients (u_n, u_{n0}) is crucial for the cut quality, but also to achieve a bounded subproblem. If the chosen heuristic yields *only* coefficients (close to or) equal to zero, in our implementation no cut is generated at all. Moreover, non-zero coefficients may not yield a bounded dual problem if they correspond to a direction $(u_n, u_{n0}) \notin \text{cone}\left(\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)\right) \setminus \{0\}$, see Sect. 3.6.2. We use infeasibility of problem (20) as a flag for possible unboundedness. In that case, we add some artificial bounds to the dual problem (14) to at least generate some valid cuts. Some pre-testing indicated that for `LN-Mid`, `LN-Relint` and `LN-Conv` using multipliers bounds $\pi_n \in [-10, 10]^{d_{a(n)}}, \pi_{n0} \in [0, 10]$ and for `LN-Eps` and `LN-In-Out` introducing an artificial objective bound of 1000 leads to reasonable results. Additionally, note that if problem (20) is feasible, we obtain an upper bound for the optimal value of the dual problem (14). We can use this bound to ensure boundedness for the approximating models in the level bundle method.

All occuring LP, MILP and QP subproblems are solved using Gurobi 9.0.3 with an optimality tolerance of 10^{-4} and a time limit of 300 seconds. All tests with binary approximation are run on a Windows machine with 64 GB RAM and an Intel Core i7-7700K processor (4.2 GHz). All tests without binary approximation are run on a Windows machine with 128 GB RAM and an Intel Xeon E5-1630v4 processor (3.7 GHz).

5.2 Comparison with Classical Lagrangian and Benders Cuts

For our first experiments we apply SDDiP (with binary approximation) to CLSP instances with 3 state variables. We only use one type of cut for the whole solution process. The results are illustrated in several figures throughout this section. The full results are provided in Appendix D.

First, we consider experiments with $T = 4$ or $T = 6$ stages and a maximum run time of 3 hours and 4 hours, respectively. The obtained lower bounds are depicted in Fig. 11. We observe that `B` and `SB` do not manage to close the optimality gap and that the obtained lower bounds stall very fast. `L`, whereas better in theory, leads to even worse lower bounds. One reason is that solving the dual problems is computationally costly, but additionally, compared to `SB` the tighter cuts seem to lead to worse incumbents on earlier stages or in following iterations. In fact, even in the first iteration, the lower bound obtained by `SB` is superior to that obtained by `L`. Many variants of deep and LN Lagrangian cuts outperform `SB` and `L` with respect to the lower bounds and gaps, even if the optimality gap is not completely closed in the predefined time horizon. While the quality of the lower bounds is better, the iteration times and the number of iterations in the bundle method are not necessarily reduced despite solving a bounded problem, especially not for `LN` cuts.

Among the new approaches, ℓ^∞-deep cuts perform rather bad. Our hypothesis is that for binary state variables this approach is very prone to degeneracy in the normalized Lagrangian dual problems, which then leads to cuts of bad quality, see Sect. 1. To test this hypothesis, we perform experiments using an additional optimization step that was proposed in an earlier version of `SDDP.jl` and resembles the two-step cut generation in [22]. In this step we minimize the norm among all optimal solutions of the Lagrangian dual. Therefore, we label this approach as `MNC` (minimal norm choice). With `MNC`, the bounds obtained by ℓ^∞-deep cuts improve considerably. For this reason, without further notice, we always use this approach in the following experiments. For all other cut generation approaches, we observe no significant improvement in lower bounds per time using an `MNC` step.

We now consider experiments with more stages, $T = 10$ and $T = 16$ to be precise,

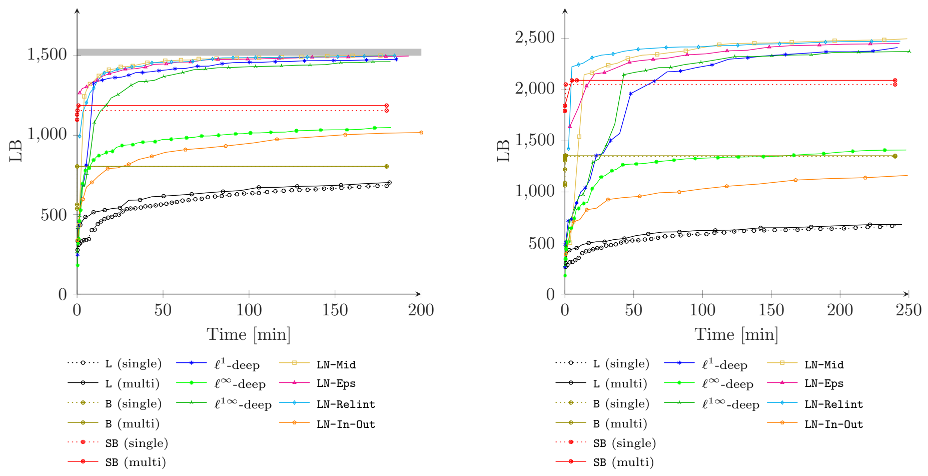


Figure 11: Lower bound development over time for experiments on CLSP with state binarization.

Note. LEFT: $T = 4$. RIGHT: $T = 6$. For $T = 4$, the shaded gray area is where the optimal value lies according to an approximate solution of the deterministic equivalent. For B and SB, SDDiP quickly terminates due to stalling lower bounds, so the last lower bound is interpolated over the whole time horizon. Marks at every second iteration, except for B and SB.

with run times of 5 and 8 hours, respectively. The obtained lower bounds are depicted in Fig. 12. We observe that deep cuts perform mediocre for 16 stages. LN cuts perform best with respect to the lower bounds, but for 16 stages hardly outperform SB. This is mainly due to long iteration times, even in comparison to deep cuts, see Fig. 13. Moreover, as Fig. 14 shows, an improvement in lower bounds does not necessarily translate to an improvement of the obtained policies. In fact, SB achieves the best simulated upper bounds. It seems that using SB it is possible to quickly identify good feasible solutions, but that the lower bounds are too loose to get a certificate for optimality, whereas for Lagrangian cuts it is the opposite.

5.3 Combination with SB Cuts

As shown in the previous section, using SDDiP with *only* Lagrangian cuts becomes extremely slow for large problems. Therefore, in practice, it is reasonable to combine different types of cuts. Already in the original SDDiP work [31] it is proposed to combine Lagrangian cuts, which can provide convergence guarantees, and strengthened Benders cuts, which can be computed efficiently.

To evaluate the performance of deep and LN cuts in this setting, we conduct experiments where we start with only SB for the first 20 iterations to get a quick bound improvement, and then generate SB cuts and Lagrangian cuts in each iteration. The lower bounds are depicted in Fig. 15, while the simulation results and optimality gaps are presented in Fig. 14 next to the ones of the previous case.

The lower bound results heavily improve for L, but are not affected too much for deep or LN cuts. They are still better for these approaches than the ones obtained using only SB, though. As the simulated upper bounds are better than in the previous setting for all types of cuts, we can conclude that a combination of SB cuts and Lagrangian

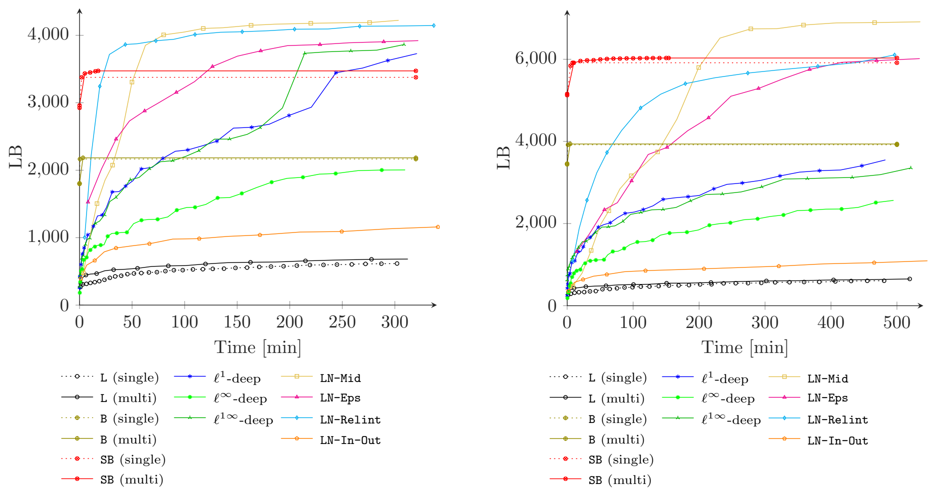


Figure 12: Lower bound development over time for experiments on CLSP with state binarization.

Note. LEFT: $T = 10$. RIGHT: $T = 16$.

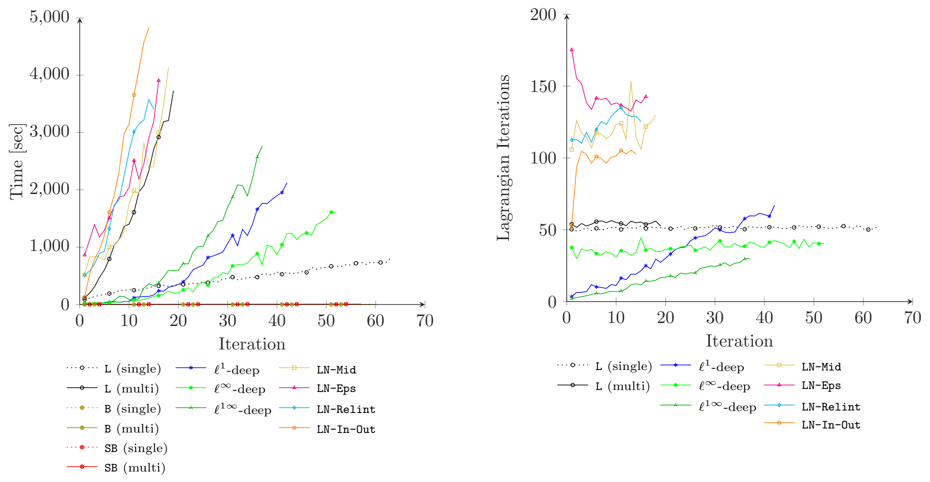


Figure 13: Different analyses for experiments on CLSP with state binarization and $T = 16$.

Note. LEFT: Time per iteration of SDDiP. RIGHT: Iterations required to solve Lagrangian dual over iterations of SDDiP.

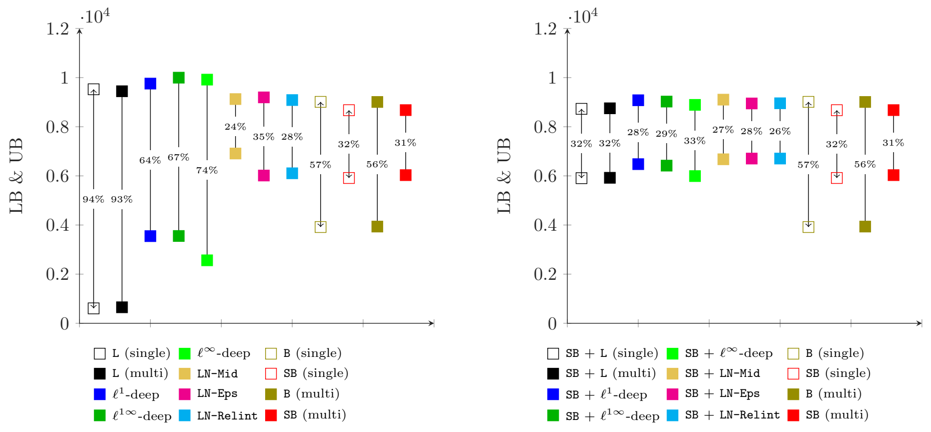


Figure 14: Optimality gaps for experiments on CLSP with state binarization with $T = 16$.

Note. LEFT: Run with only one type of cuts. RIGHT: Runs with `SB` plus additional cuts from iteration 21.

cuts combines the advantages of good lower bounds and reasonably good simulation results for the policies.

5.4 The Chen-Luedtke Approach: Restricting the Dual Space

Another approach suited to accelerate SDDiP was recently put forward by Chen and Luedtke [11] for the two-stage case. They propose to restrict the feasible set of the normalized Lagrangian dual problem (14) to a small subset of valid multipliers (π_n, π_{n0}). More precisely, the idea is to restrict the multipliers π_n to the span of a set of previously generated Benders cut coefficients $\widehat{\pi}_n^k, k = 1, \dots, K$, for some predefined parameter K. That is, we introduce the constraint

$$\pi_n = \sum_{k=1}^{K} \gamma_{nk} \widehat{\pi}_n^k,$$

which also means that we add variables $\gamma_{nk}, k = 1, \dots, K$, to the dual problem. While we lose tightness and convergence guarantees using this dual space restriction, the search space for the level Bundle method is significantly reduced, so that cuts can be generated faster. We refer to this as the `CL` approach.

In principle, the `CL` approach can be combined with any of the previously used normalization techniques. Additionally, it allows for an alternative normalization [11]. Instead of using $g(\pi_n, \pi_{n0}) = \|\pi_n, \pi_{n0}\|_1$, we may as well use some normalization function $g(\pi_n, \pi_{n0}, \gamma_n) = \|\gamma_n, \pi_{n0}\|_1$. This choice should lead to solutions with sparse γ. In the following, we denote this approach by `CL`-γ.

For our computational experiments, we choose $K = 20$. Apart from the dual space restriction, we use the same setting as for the previous runs (20 iterations of only `SB`, after that `SB` and Lagrangian cuts).

The results are depicted in Fig. 16. The number of Lagrangian iterations and the time per iteration are reduced significantly. Moreover, compared to only using La-

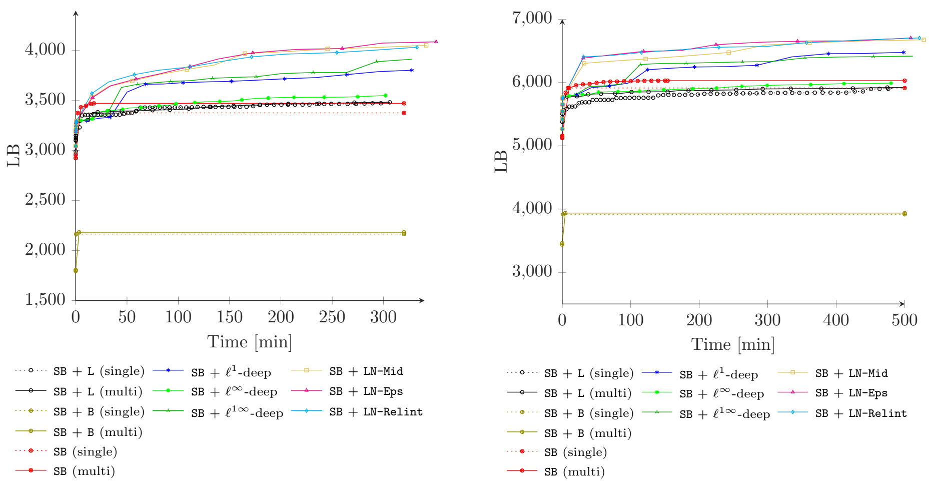


Figure 15: Lower bound development over time for experiments on CLSP with state binarization using a combination with SB cuts.

Note. LEFT: $T = 10$. RIGHT: $T = 16$. 20 iterations with SB and then SB and Lagrangian cuts in each iteration.

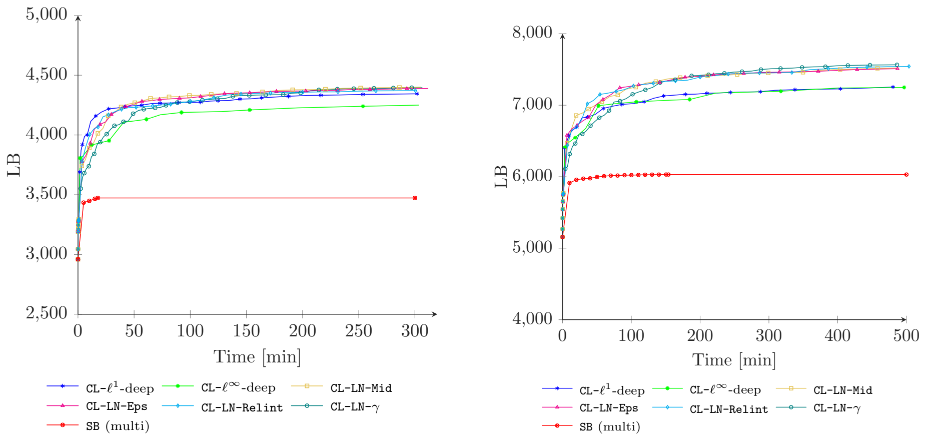


Figure 16: Lower bound development over time for experiments on CLSP with state binarization using the Chen-Luedtke approach.

Note. LEFT: $T = 10$. RIGHT: $T = 16$. 20 iterations with SB and then SB and Lagrangian cut in each iteration with dual space restriction for $K = 20$.

grangian cuts, much better lower bounds are obtained in the same time. This implies that the lower bounds for the combined approach are also much better than for only using `SB`. Interestingly, the lower bounds are even better *per iteration* than without dual space restriction, similar to what we observed for `SB` before. This illustrates that the quality of cuts does not only depend on tightness or depth, but also on the incumbents which they induce in the previous stages and following iterations. The chosen normalization approach seems not to be decisive in this setting.

5.5 Results for Larger Instances

As discussed before, for experiments with a larger state dimension, we do not apply a binary approximation. This means that we do not have convergence guarantees and cannot expect the optimality gap to be closed. However, in return iterations should take considerably less time. Note that this is often the go-to approach to apply SDDP-like methods to mixed-integer programs in practice.

In this case, we also consider `LN-Conv` with convex combination parameters of 0.5, 0.75, 0.9 and 0.99 (with higher values encoding a higher proximity to $\big(x^i_{a(n)}, \underline{Q}^{i+1}_n(x^i_{a(n)})\big)$).

The results show that deep and `LN` Lagrangian cuts manage to achieve better lower bounds and gaps than conventional cuts, see Fig. 17 and 18. Using a dual space restriction allows to further reduce the optimality gap to about 21% in 3 hours and to about 19% in 5 hours. This is a considerable improvement compared to Benders cuts, which are most frequently used in practice. This means that the proposed cut generation techniques may be helpful to improve the convergence behavior even if no binary approximation is applied. However, as for the previous experiments, we cannot conclude that the improvement in lower bounds necessarily leads to an improvement of the in-sample performance of the obtained policy, and thus to better optimality gaps.

We also observe that the average iteration time is reduced considerably compared to the previous test cases with binary state approximation (about 60-75% reduction). For this reason, and because the state space is also lower-dimensional, even without convergence guarantees, better optimality gaps are obtained than for the previous test cases in the same time.

5.6 Discussion and Potential Improvements

Overall, our results show significant improvements of the obtained lower bounds using the new cut generation framework in all cases: with state binarization, without state binarization, combined with strengthened Benders cuts or applying the cuts on their own. With binarization, especially LN cuts yield strong improvements, whereas without binarization also deep cuts perform reasonably well. We see that better lower bounds do not necessarily translate to better performances of the obtained policies, though. Additionally, we observe that even using the new framework, SDDiP suffers from well-known computational drawbacks such as high computational cost to solve Lagrangian dual problems (especially if a binary approximation of the state space is applied) and slow convergence of lower bounds due to premature stalling [2], so even after hours of run time the observed optimality gaps are still considerable.

In the future, the performance of SDDiP including our proposed cut generation framework could be improved in several ways. First, the solution of independent Lagrangian duals for nodes $m \in \mathcal{C}(n)$ could be parallelized. Second, potential warm starting or acceleration techniques for the Lagrangian dual (*e.g.* using sub-optimal so-

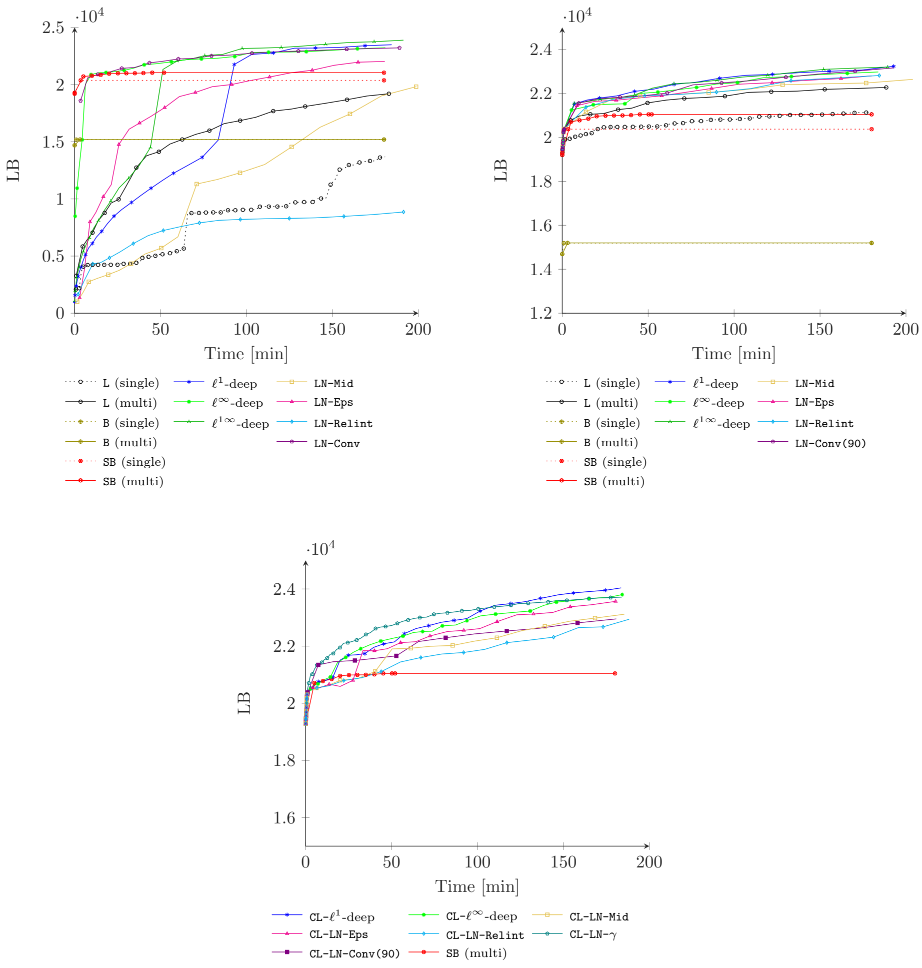


Figure 17: Lower bound development over time for experiments on CLSP with $T = 16$ and without state binarization.

Note. UP LEFT: Run with only one type of cuts. UP RIGHT: Runs with SB plus additional cuts from iteration 21. BELOW: Runs with Chen-Luedtke approach for $K = 20$.

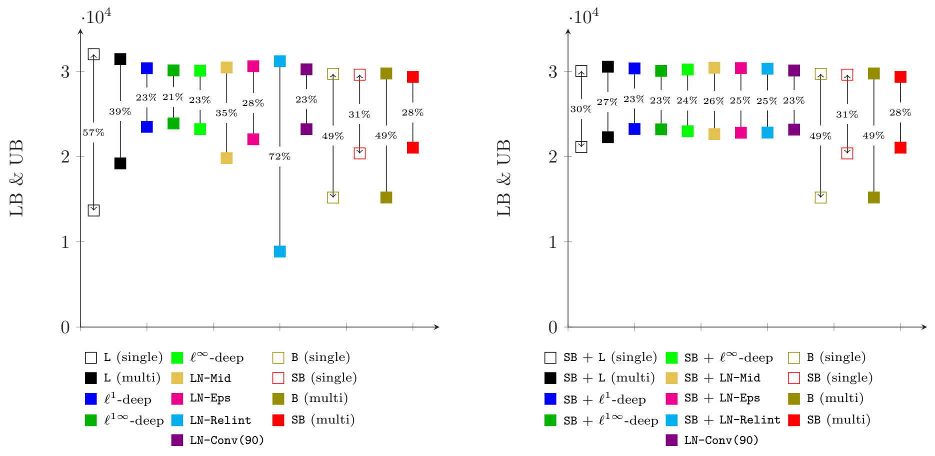


Figure 18: Optimality gaps for experiments on CLSP with $T = 16$ and without state binarization.

Note. LEFT: Run with only one type of cuts. RIGHT: Runs with `SB` plus additional cuts from iteration 21.

lutions) could be explored, even if challenging for multistage problems. Third, the dual space restriction suggested by Chen and Luedtke [11] looks promising to reduce the computational effort while not compromising cut quality by too much. We think that future research could focus more on priorly restricting the dual space to reduce the computational effort for solving Lagrangian dual problems.

Our computational experiments also reveal that, especially for LN cuts, SDDiP may occasionally suffer from numerical issues, and that the obtained results show a high sensitivity with respect to the chosen parameters. First of all, it is a common issue of cutting-plane methods that they may lead to ill-conditioned problems if the cut and problem coefficients are not properly scaled. In this context, an appropriate choice of normalization coefficients (u_n, u_{n0}) for LN cuts is crucial. In addition, core point identification in general remains a challenging task, especially when integer requirements are apparent. Addressing these challenges in detail merits further research.

Finally, we have only considered a very specific test problem so far. For a more profound and general performance assessment, therefore experiments for more and also larger test problems have to be carried out. In fact, we are planning to enhance our experiments with tests of a capacited facility location problem with pure binary state variables and local integer constraints. This will also allow us to further explore the challenges of core point identification in the context of integer requirements.

6 Conclusion

In this article, we propose a new framework to generate Lagrangian cuts for value functions occurring in MS-MILPs, which generalizes earlier proposals for 2-stage problems. We prove that using different normalizations of the Lagrangian dual problems, cuts with different favorable properties can be obtained, such as maximal depth, being facet-defining or Pareto-optimal. Our framework allows for a lot of flexibility in cut generation, and thus notably extends the toolbox of SDDiP. If all state variables are binary, finite almost sure convergence of SDDiP is assured, as for classical Lagrangian cuts.

We provide computational results for experiments on a capacitated lot-sizing problem. The results show that the lower bounds in SDDiP can be vastly improved by incorporating our proposed framework, although not eliminating other well-known computational drawbacks, such as excessive computational effort, slow convergence and inability to close the optimality gap. As described in the previous section, therefore more theoretical and computational research is required to efficiently apply our proposed framework, and SDDiP in general, on large-scale problems in practice.

Finally, our cut generation framework requires a multi-cut approach, whereas for multistage problems often a single-cut approach is favored computationally, as much less cuts have to be added per iteration. Trying to compute deep Lagrangian cuts or LN Lagrangian cuts in a single-cut framework in a computationally efficient way could therefore be an interesting research direction.

Acknowledgements

Andy Sun's research is partially funded by the National Science Foundation CAREER award 2316675. Christian Füllner's research is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)–445857709. This work was started during Christian Füllner's research visit at Georgia Institute of Technology, which was funded by the Karlsruhe House of Young Scientists (KHYS). The authors thank Filipe Cabral for providing data for the capacited lot-sizing problem considered in the computational experiments.

A Classical Lagrangian Cuts

For any $n \in \overline{\mathcal{N}}$, let $\mathfrak{Q}_n^{i+1}(\cdot)$ denote the current cut approximation for value function $Q_n(\cdot)$ in a decomposition method, such as SDDiP. Then, a Lagrangian cut can be generated by considering a special Lagrangian relaxation of the nodal subproblem (that is, subproblem (3) with $Q_m(\cdot), m \in \mathcal{C}(n)$, replaced by $\mathfrak{Q}_m^{i+1}(\cdot)$). More precisely, the copy constraints $z_n = x_{a(n)}^i$ are relaxed using a given vector of dual multipliers $\pi_n \in \mathbb{R}^{d_{a(n)}}$, which yields

$$\mathcal{L}_n^{i+1}(\pi_n) := \min_{x_n, y_n, z_n, (\theta_m)} \Big\{ f_n(x_n, y_n) + \sum_{m \in \mathcal{C}(n)} p_{nm}\theta_m - \pi_n^\top z_n \ : \ (z_n, x_n, y_n) \in \mathcal{F}_n, \\ z_n \in Z_{a(n)}, \theta_m \geq \mathfrak{Q}_m^{i+1}(\cdot)(x_n),\ m \in \mathcal{C}(n) \Big\}.$$

For varying π_n, this relaxation defines the *dual function* $\mathcal{L}_n^{i+1}(\cdot)$. The problem of optimizing the dual function over the dual multipliers π_n is the *Lagrangian dual problem*:

$$\max_{\pi_n} \Big\{ \mathcal{L}_n^{i+1}(\pi_n) + \pi_n^\top x_{a(n)}^i \Big\}. \tag{21}$$

By solving problem (21), a Lagrangian cut for $Q_n(\cdot)$ can be derived as

$$\theta_n \geq \mathcal{L}_n^{i+1}(\pi_n^i) + (\pi_n^i)^\top x_{a(n)}, \tag{22}$$

where π_n^i denotes feasible dual multipliers in (21) for node n [31]. If required, feasibility cuts can be derived in a similar fashion [see 11, 25].

The Lagrangian cuts (22) have useful properties [31]. Their right-hand sides are valid under-estimators of $Q_n(\cdot)$ and tight at $\overline{\text{co}}(\underline{Q}_n^{i+1})(x_{a(n)}^i)$ (given that optimal dual multipliers π_n^i are used). Moreover, only finitely many different Lagrangian cuts exist if only dual basic solutions are considered. Finally, if the state variables x_n are binary, the cuts are even tight at $\underline{Q}_n^{i+1}(x_{a(n)}^i)$. These properties ensure almost sure finite convergence of SDDiP.

B Proofs

In this section, we present the proofs that are not displayed in the main text.

B.1 Proof of Lemma 3.7

Proof. This is proven in [11] for $\text{epi}(Q_n)$, but we provide a customized proof here. Let $(x_{a(n)}, \theta_n) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$. Then,

$$\begin{aligned}(\pi_n^i)^\top x_{a(n)} + \pi_{n0}^i\theta_n &\geq \min_{x_{a(n)}, \theta_n} \Big\{ (\pi_n^i)^\top x_{a(n)} + \pi_{n0}^i\theta_n \ : \ (x_{a(n)}, \theta_n) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1})) \Big\} \\ &= \min_{\lambda_n, z_n} \Big\{ (\pi_n^i)^\top z_n + \pi_{n0}^i c_n^\top \lambda_n \ : \ (\lambda_n, z_n) \in \text{conv}(\mathcal{W}_n^{i+1}) \Big\} \\ &= \min_{\lambda_n, z_n} \Big\{ (\pi_n^i)^\top z_n + \pi_{n0}^i c_n^\top \lambda_n \ : \ (\lambda_n, z_n) \in \mathcal{W}_n^{i+1} \Big\} \\ &= \mathscr{L}_n^{i+1}(\pi_n^i, \pi_{n0}^i).\end{aligned}$$

The inequality follows by feasibility. The first equality uses the same relation that is also applied in (11). The second equality exploits that the objective function is linear, and the last one follows from the definition of $\mathscr{L}_n^{i+1}(\cdot)$ in (9). The second part of the assertion follows with $\operatorname{epi}(Q_n) \subseteq \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$. □

B.2 Proof of Lemma 3.10

Proof. According to Brandenberg and Stursberg [9], $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$ can be rewritten as

$$\begin{aligned}\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i) = \Big\{ &(\gamma_n, \gamma_{n0}) \in \mathbb{R}^{d_{a(n)}} \times \mathbb{R} \ : \\ &\gamma_n^\top x_{a(n)}^i + \gamma_{n0}\theta_n^i - \operatorname{supp}_{\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))}(\gamma_n, \gamma_{n0}) \geq 1 \Big\},\end{aligned} \tag{23}$$

where $\operatorname{supp}_{\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))}(\cdot)$ denotes the support function of $\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))$. This function can be expressed as follows:

$$\begin{aligned}&\operatorname{supp}_{\operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1}))}(\gamma_n, \gamma_{n0}) \\ &= \max_{x_{a(n)}, \theta_n} \left\{ \gamma_n^\top x_{a(n)} + \gamma_{n0}\theta_n \ : \ (x_{a(n)}, \theta_n) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{Q}_n^{i+1})) \right\} \\ &= \max_{x_{a(n)}, \theta_n, \lambda_n, z_n} \Big\{ \gamma_n^\top x_{a(n)} + \gamma_{n0}\theta_n \ : \ \tilde{A}_n\lambda_n + \tilde{B}_n z_n \geq \tilde{d}_n, z_n = x_{a(n)}, \theta_n - c_n^\top \lambda_n \geq 0 \Big\} \\ &= \min_{\mu_n, \pi_n, \pi_{n0}} \Big\{ \tilde{d}_n^\top \mu_n \ : \ \tilde{A}_n^\top \mu_n - c_n\pi_{n0} = 0, \tilde{B}_n^\top \mu_n - \pi_n = 0, \\ &\qquad\qquad\qquad\qquad \pi_n = \gamma_n, \pi_{n0} = \gamma_{n0}, \pi_{n0} \leq 0, \mu_n \leq 0 \Big\} \\ &= \min_{\mu_n} \Big\{ -\tilde{d}_n^\top \mu_n \ : \ -\tilde{A}_n^\top \mu_n - c_n\gamma_{n0} = 0, -\tilde{B}_n^\top \mu_n - \gamma_n = 0, \gamma_{n0} \leq 0, \mu_n \geq 0 \Big\}.\end{aligned} \tag{24}$$

The first equation applies the definition of support functions. The second one follows from Remark 3.3 and the third one exploits strong duality for LPs. We insert (24) into (23), and observe that the set remains unchanged if we replace the minimum operator using an existence quantor. □

B.3 Proof of Lemma 3.14

Proof. According to Theorem 3.5, we have $\widehat{v}_n^{D,i+1}(x_{a(n)}^i, \theta_n^i) = 0$ for the non-normalized Lagrangian dual (10). By definition of (10) and its normalization (14), we can thus conclude $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) \leq \widehat{v}_n^{D,i+1}(x_{a(n)}^i, \theta_n^i) = 0$.

Let $(\widehat{\pi}_n, \widehat{\pi}_{n0})$ be an optimal point for problem (10), *i.e.*, $\mathscr{L}_n^{i+1}(\widehat{\pi}_n, \widehat{\pi}_{n0}) - (\widehat{\pi}_n)^\top x_{a(n)}^i - \widehat{\pi}_{n0}\theta_n^i = 0$. If $\|\widehat{\pi}_n, \widehat{\pi}_{n0}\| \leq 1$, then it is also feasible for (14). As the objective of both problems is the same, $\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) = 0$.

Otherwise, there exists $\mu > 0$ such that $\frac{1}{\mu}(\widehat{\pi}_n, \widehat{\pi}_{n0})$ is feasible for (14). By feasibility, it follows

$$\begin{aligned}\widehat{v}_n^{ND,i+1}(x_{a(n)}^i, \theta_n^i) &\geq \mathscr{L}_n^{i+1}\left(\frac{1}{\mu}\widehat{\pi}_n, \frac{1}{\mu}\widehat{\pi}_{n0}\right) - \frac{1}{\mu}(\widehat{\pi}_n)^\top x_{a(n)}^i - \frac{1}{\mu}\widehat{\pi}_{n0}\theta_n^i \\ &= \frac{1}{\mu}\Big(\mathscr{L}_n^{i+1}(\widehat{\pi}_n, \widehat{\pi}_{n0}) - (\widehat{\pi}_n)^\top x_{a(n)}^i - \widehat{\pi}_{n0}\theta_n^i\Big) = 0,\end{aligned} \tag{25}$$

where we exploited that $\mathscr{L}_n^{i+1}(\cdot)$ is positive homogeneous. The reverse direction can be shown in a similar way. □

B.4 Proof of Lemma 3.23

Proof. By definition, the normalized Lagrangian dual problem is equivalent to

$$\begin{aligned}&\max_{(\pi_n,\pi_{n0})\in\Pi_n}\left\{\min_{(\lambda_n,z_n)\in\mathcal{W}_n^{i+1}}\left\{\pi_n^\top(z_n-x^i_{a(n)})+\pi_{n0}(c_n^\top\lambda_n-\theta_n^i)\right\}\right\}\\ =&\max_{(\pi_n,\pi_{n0})\in\Pi_n}\left\{\min_{(\lambda_n,z_n)\in\operatorname{conv}(\mathcal{W}_n^{i+1})}\left\{\pi_n^\top(z_n-x^i_{a(n)})+\pi_{n0}(c_n^\top\lambda_n-\theta_n^i)\right\}\right\}\end{aligned}$$

with $\Pi_n := \left\{(\pi_n,\pi_{n0}) \in \mathbb{R}^{d_{a(n)}}\times\mathbb{R} \;:\; \pi_{n0}\geq 0, u_n^\top\pi_n+u_{n0}\pi_{n0}\leq 1\right\}$. The equation follows from linearity.

Using Remark 3.3 and then LP duality for the inner minimization problem, we obtain the equivalent problem

$$\begin{aligned}\max_{\pi_n,\pi_{n0},\mu_n}\Big\{\tilde{d}_n^\top\mu_n-\pi_n^\top x^i_{a(n)}-\pi_{n0}\theta_n^i\;:\;&\mu_n\geq 0,\;\pi_{n0}\geq 0,\;u_n^\top\pi_n+u_{n0}\pi_{n0}\leq 1,\\ &\tilde{A}_n^\top\mu_n-\pi_{n0}c_n=0,\;\tilde{B}_n^\top\mu_n-\pi_n=0\Big\}.\end{aligned}$$

This is an LP. Using LP duality and Remark 3.3 again, the assertion follows. □

B.5 Proof of Theorem 3.25

To prove this, we first require the definition and some results for the alternative polyhedron.

Definition B.1 ([17])**.** *The* alternative polyhedron *of* (10) *is defined as*

$$\mathcal{A}_n(x^i_{a(n)},\theta^i_n) := \left\{(\mu_n,\pi_n,\pi_{n0})\in\mathbb{R}^k\times\mathbb{R}^{d_{a(n)}}\times\mathbb{R}\;:\;\begin{array}{l}\mu_n,\pi_{n0}\geq 0\\ \tilde{A}_n^\top\mu_n-\pi_{n0}c_n=0\\ \tilde{B}_n^\top\mu_n-\pi_n=0\\ \tilde{d}_n^\top\mu_n-\pi_n^\top x^i_{a(n)}-\pi_{n0}\theta^i_n=1\end{array}\right\}.$$

If we relax the last equality constraint to $\tilde{d}_n^\top\mu_n-\pi_n^\top x^i_{a(n)}-\pi_{n0}\theta^i_n\geq 1$*, we call the obtained set the* relaxed alternative polyhedron *and denote it by* $\widehat{\mathcal{A}}_n(x^i_{a(n)},\theta^i_n)$*.*

As shown in [9], the sets $\widehat{\mathcal{A}}_n(x^i_{a(n)},\theta^i_n)$ and $\mathcal{R}_n^{i+1}(x^i_{a(n)},\theta^i_n)$ are closely related by a linear transformation, which in our case involves a unit matrix $I_{d_{a(n)}}$ of dimension $d_{a(n)}$.

Lemma B.2 (Theorem 2.1 in [9])**.** *The reverse polar set* $\mathcal{R}_n^{i+1}(x^i_{a(n)},\theta^i_n)$ *and the relaxed alternative polyhedron* $\widehat{\mathcal{A}}_n(x^i_{a(n)},\theta^i_n)$ *satisfy the relation*

$$\begin{pmatrix}\begin{pmatrix}0 & -I_{d_{a(n)}}\end{pmatrix} & 0\\ 0 & -1\end{pmatrix}\widehat{\mathcal{A}}_n(x^i_{a(n)},\theta^i_n)=\mathcal{R}_n^{i+1}(x^i_{a(n)},\theta^i_n).$$

Therefore, for problem (18) we obtain the related problem

$$\max_{\mu_n,\pi_n,\pi_{n0}}\left\{-u_n^\top\pi_n-u_{n0}\pi_{n0}\;:\;(\mu_n,\pi_n,\pi_{n0})\in\widehat{\mathcal{A}}_n^{i+1}(x^i_{a(n)},\theta^i_n)\right\}. \qquad (26)$$

We can finally prove Theorem 3.25.

Proof. We first prove (i). It can be shown that the normalization constraint is binding at an optimal solution of the normalized Lagrangian dual problem (14), see the reasoning in Proposition 9 in [21]. Then, by using the reformulation ideas from Remark 3.3 and applying Theorem 3.20 from [28], this problem is equivalent to problem (26) in the following way:

Let $(\mu_n^*, \pi_n^*, \pi_{n0}^*)$ be optimal for (14) with optimal value $v_n^* > 0$. Then $(\bar{\mu}_n, \bar{\pi}_n, \bar{\pi}_{n0}) = \frac{1}{v_n}(\mu_n^*, \pi_n^*, \pi_{n0}^*)$ is optimal for (26) with optimal value $\bar{v}_n = -\frac{1}{v_n} < 0$. In reverse, let $(\bar{\mu}_n, \bar{\pi}_n, \bar{\pi}_{n0})$ be optimal for (26) with optimal value $\bar{v}_n < 0$. Then $(\mu_n^*, \pi_n^*, \pi_{n0}^*) = -\frac{1}{v_n}(\bar{\mu}_n, \bar{\pi}_n, \bar{\pi}_{n0})$ is optimal for (14) with optimal value $v_n^* = -\frac{1}{\bar{v}_n} > 0$. Note that the optimal values multiply to -1.

According to Corollary 2.2 in [9], which is based on the relation stated in Lemma B.2, $(\widehat{\gamma}_n, \widehat{\gamma}_{n0})$ is optimal for problem (18) if and only if there exists some $\widehat{\mu}_n$ such that $(\bar{\mu}_n, \bar{\pi}_n, \bar{\pi}_{n0}) = (\widehat{\mu}_n, -\widehat{\gamma}_n, -\widehat{\gamma}_{n0})$ is optimal for problem (26). Moreover, the optimal values $\widehat{v}_n$ and $\bar{v}_n$ are the same.

Hence, by the first step, we have a scaling relation between the optimal points of (14) and (26), and by the second step, we have a sign change relation between the optimal points of (26) and (18). The optimal values multiply to -1 after the first step and do not change in the second step. This proves the assertion.

We now prove (ii). Let $(\widehat{\gamma}_n, \widehat{\gamma}_{n0})$ be optimal for problem (18) with a valid certificate $\widehat{\mu}_n \geq 0$ in $\mathcal{R}_n^{i+1}(x_{a(n)}^i, \theta_n^i)$. Then, the valid cut

$$\tilde{d}_n^\top \widehat{\mu}_n + (\widehat{\gamma}_n)^\top x_{a(n)} + \widehat{\gamma}_{n0}\theta_n \leq 0$$

is induced, separating $(x_{a(n)}^i, \theta_n^i)$ from $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ [13]. By the optimality relations used to prove (i) and by multiplying with $-\frac{1}{\widehat{v}_n} = -\frac{1}{\bar{v}_n} > 0$, this is equivalent to

$$-\frac{1}{\bar{v}_n}\tilde{d}_n^\top \bar{\mu}_n + \frac{1}{\bar{v}_n}\bar{\pi}_n^\top x_{a(n)} + \frac{1}{\bar{v}_n}\bar{\pi}_{n0}\theta_n \leq 0.$$

However, this is equivalent to

$$\tilde{d}_n^\top \mu_n^* - (\pi_n^*)^\top x_{a(n)} - \pi_{n0}^*\theta_n \leq 0$$

and by definition also to

$$\mathscr{L}_n^{i+1}(\pi_n^*, \pi_{n0}^*) - (\pi_n^*)^\top x_{a(n)} - \pi_{n0}^*\theta_n \leq 0,$$

which exactly corresponds to the Lagrangian cut (13). The reverse direction follows in a similar way. □

B.6 Proof of Lemma 3.26

Proof. Suppose that condition (19) is satisfied. Then there exist some $(\tilde{x}_{a(n)}, \tilde{\theta}_n) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1})) - (x_{a(n)}^i, \theta_n^i)$ and $\mu > 0$ such that $(u_n, u_{n0}) = \mu(\tilde{x}_{a(n)}, \tilde{\theta}_n)$. This implies $(\tilde{x}_{a(n)} + x_{a(n)}^i, \tilde{\theta}_n + \theta_n^i) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$. Therefore, the system

$$\left\{(\lambda_n, z_n) \;:\; \tilde{\theta}_n + \theta_n^i \geq c_n^\top \lambda_n, (\lambda_n, z_n) \in \text{conv}(\mathcal{W}_n^{i+1}), z_n = \tilde{x}_{a(n)} + x_{a(n)}^i\right\}$$

is non-empty. However, this set is equivalent to

$$\Big\{(\lambda_n, z_n) \ : \ \frac{1}{\mu}u_{n0} \geq c_n^\top \lambda_n - \theta_n^i, (\lambda_n, z_n) \in \operatorname{conv}(\mathcal{W}_n^{i+1}), z_n - x_{a(n)}^i = \frac{1}{\mu}u_n\Big\}.$$

With choosing $\eta_n = \frac{1}{\mu} > 0$ it immediately follows that problem (16) is feasible. By $\eta_n \geq 0$, its optimal value is also bounded from below, hence it is finite. By LP duality this implies that Assumption 3 is satisfied. □

B.7 Proof of Lemma 4.1

Proof. Recall that $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$ can be expressed by problem (12). Moreover, according to Remark 3.3, under Assumption 1, the set $\operatorname{conv}(\mathcal{W}_n^{i+1})$ is a convex polyhedron. This implies that problem (12) can be reformulated as an LP where the state $x_{a(n)}$ appears in the RHS only. The assertion then follows from LP duality and the finite number of different dual extreme points. For $\overline{\text{co}}(Q_n)(\cdot)$ a very similar reasoning can be used. □

B.8 Proof of Theorem 4.2

Proof. We start with considering leaf nodes $n \in N \setminus \tilde{N}$, where $\Psi_m^{i+1} = \emptyset$ is fixed by definition. According to Lemma 4.1, $\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot)$ is a piecewise linear convex function, so $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$ is polyhedral and has finitely many facets. As Theorem 3.27 (ii) is satisfied, all generated cuts are facet-defining. This means that only finitely many different cuts can be generated, and thus only finitely many different realizations of Ψ_n^{i+1} exist. In particular, after finitely many steps, for any $(x_{a(n)}^i, \theta_n^i)$ computed in the forward pass of NBD, a cut will have been generated that supports $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1})(\cdot))$ at this point. Additionally, as no child nodes exist, we have $\underline{Q}_n^{i+1}(\cdot) \equiv Q_n(\cdot)$, and as $X_{a(n)} = \{0,1\}^{d_{a(n)}}$, we have $\overline{\text{co}}(Q_n)(x_{a(n)}) = Q_n(x_{a(n)})$ for all $x_{a(n)} \in X_{a(n)}$. This implies that the cut supports $\text{epi}(Q_n)$ at $(x_{a(n)}^i, \theta_n^i)$ (*tight* cut).

For each realization of Ψ_n^{i+1}, the same reasoning as above can now be applied to the ancestor nodes of the leaf nodes. The assertion then follows by induction over the whole scenario tree. This implies that NBD terminates with an optimal policy for (MS-MILP). □

B.9 Proof of Lemma 4.3

Proof. For any $\tilde{\nu} \in \mathbb{N}$, the trial point $(\bar{x}_{a(n)}, \theta_n^{i_{\tilde{\nu}}})$ either satisfies $(\bar{x}_{a(n)}, \theta_n^{i_{\tilde{\nu}}}) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i_{\tilde{\nu}}+1}))$, and by that $\theta_n^{i_{\tilde{\nu}}} = \bar{\theta}_n$, or it satisfies $\theta_n^{i_{\tilde{\nu}}} < \bar{\theta}_n$. In the first case, $(\bar{x}_{a(n)}, \theta_n^{i_{\tilde{\nu}}}) = (\bar{x}_{a(n)}, \bar{\theta}_n) = (\widehat{x}_{a(n)}^{i_{\tilde{\nu}}}, \widehat{\theta}_n^{i_{\tilde{\nu}}})$ and the assertion is trivially satisfied for all $\nu \geq \tilde{\nu}$.

In the second case, the trial point is separated from $\text{epi}(\overline{\text{co}}(\underline{Q}_n^{i_{\tilde{\nu}}+1}))$ by a newly constructed Lagrangian cut. Therefore, the sequence $(\theta_n^{i_\nu})_{\nu \in \mathbb{N}}$ is monotonically increasing and bounded, thus converging. More precisely, by the above separation argument it converges to $\bar{\theta}_n$. It immediately follows that the distance between the trial point and $(\bar{x}_{a(n)}, \bar{\theta}_n)$ converges to zero:

$$\lim_{\nu \to \infty} \|\bar{x}_{a(n)} - x_{a(n)}^{i_\nu}, \bar{\theta}_n - \theta_n^{i_\nu}\|_* = \|\bar{x}_{a(n)} - \bar{x}_{a(n)}, \bar{\theta}_n - \theta_n^{i_\nu}\|_* = 0. \tag{27}$$

Since $(\bar{x}_{a(n)}, \bar{\theta}_n)$ is always feasible for the projection problem (15), this also implies that any sequence $(\widehat{x}_{a(n)}^{i_\nu}, \widehat{\theta}_n^{i_\nu})_{\nu \in \mathbb{N}}$ of (possibly non-unique) solutions of problem (15) satisfies

$$\lim_{\nu \to \infty} \|\widehat{x}_{a(n)}^{i_\nu} - \bar{x}_{a(n)}, \widehat{\theta}_n^{i_\nu} - \theta_n^{i_\nu}\|_* = 0. \tag{28}$$

By using the triangle inequality together with (27) and (28) we obtain

$$\lim_{\nu \to \infty} \|\widehat{x}_{a(n)}^{i_\nu} - \bar{x}_{a(n)}, \widehat{\theta}_n^{i_\nu} - \bar{\theta}_n\|_* = 0.$$

From the definition of norms, the assertion follows. □

B.10 Proof of Lemma 4.4

Proof. We prove the result by contradiction. For that reason, we assume that there exists an infinite subsequence, indexed by $\ell \in \mathbb{N}$, such that for all ℓ there exists some $\widehat{k}^{\nu_\ell} \in \widehat{K}^{\nu_\ell}$ with $\widehat{k}^{\nu_\ell} \notin \overline{K}$. For simplicity, we assume that this facet index $\widehat{k}^{\nu_\ell}$ is the same for all ℓ, *i.e.*, $\widehat{k}^{\nu_\ell} \equiv: \tilde{k}$, even though this is not guaranteed. If it is not true, however, we can apply the same arguments after another restriction to subsequences for each possible facet, of which only finitely many exist.

Let the facet $F_{\tilde{k}}$ be described by the equation $q^\top x_{a(n)} + q_0 \theta_n = r$, with appropriate coefficients $q_0, r \in \mathbb{R}$ and $q \in \mathbb{R}^{d_{a(n)}}$. Because of $\tilde{k} \in \widehat{K}^{\nu_\ell}$ for all ℓ, it follows that

$$q^\top \widehat{x}_{a(n)}^{i_{\nu_\ell}} + q_0 \widehat{\theta}_n^{i_{\nu_\ell}} - r = 0$$

This immediately implies that

$$\lim_{\ell \to \infty} \left(q^\top \widehat{x}_{a(n)}^{i_{\nu_\ell}} + q_0 \widehat{\theta}_n^{i_{\nu_\ell}} - r \right) = 0. \tag{29}$$

On the other hand, we know from Lemma 4.3 that $(\widehat{x}_{a(n)}^{i_{\nu_\ell}}, \widehat{\theta}_n^{i_{\nu_\ell}})$ converges to $(\bar{x}_{a(n)}, \bar{\theta}_n)$, as for a convergent series every subsequence converges to the same limit. This implies

$$\begin{aligned}
&\lim_{\ell \to \infty} \left(q^\top \widehat{x}_{a(n)}^{i_{\nu_\ell}} + q_0 \widehat{\theta}_n^{i_{\nu_\ell}} - r \right) \\
&= q^\top \lim_{\ell \to \infty} \left(\widehat{x}_{a(n)}^{i_{\nu_\ell}} \right) + q_0 \lim_{\ell \to \infty} \left(\theta_n^{i_{\nu_\ell}} \right) - r \\
&= q^\top \bar{x}_{a(n)} + q_0 \bar{\theta}_n - r \\
&< 0.
\end{aligned} \tag{30}$$

The inequality follows from $\widehat{k}^{\nu_\ell} \notin \overline{K}$ for all $\ell \in \mathbb{N}$ and $(\bar{x}_{a(n)}, \bar{\theta}_n) \in \text{epi}(\overline{\text{co}}(\underline{Q}_n^{i+1}))$. Obviously, the results in (29) and (30) contradict each other, so our initial assumption must have been wrong. □

B.11 Proof of Lemma 4.6

Proof. Since at least x^2 satisfies $x^2 \in \text{relint}(\mathfrak{F})$, there exists some $\lambda > 1$ such that the point $x^3 := x^1 + \lambda(x^2 - x^1) = (1-\lambda)x^1 + \lambda x^2$ is contained in $\mathfrak{F}$ as well. We can now prove the assertion by contradiction. Let α and β denote the intercept and the slope of the cut supporting S at x^2. Then, we have $\alpha + \beta^\top x^2 = 0$. Now assume that the cut does not support x^1, which implies $\alpha + \beta^\top x^1 < 0$, as $x^1 \in S$. We obtain

$$\alpha + \beta^\top x^3 = \lambda(\alpha + \beta^\top x^2) + (1-\lambda)(\alpha + \beta^\top x^1) = (1-\lambda)(\alpha + \beta^\top x^1) > 0. \tag{31}$$

The first equation applies the definition of x^3, the second one follows from $\alpha+\beta^\top x^2 = 0$. The third one follows from $\alpha+\beta^\top x^1 < 0$ and $\lambda > 1$. The result in (31) implies that either $x^3 \notin S$ or that the cut is not valid for all $x \in S$. Both cases lead to a contradiction. □

B.12 Proof of Lemma 4.7

Proof. We assume that ν is sufficiently large, that is, $\nu \geq \check{\nu}$ for some $\check{\nu} \in \mathbb{N}$ satisfying $\check{\nu} \geq \tilde{\nu}$ from Lemma 4.3, $\check{\nu} \geq \widehat{\nu}$ from Lemma 4.4 and $\check{\nu} \geq \bar{\nu}$ from Lemma 4.5.

If for all $\nu \geq \check{\nu}$ we have $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n) = (\bar{x}_{a(n)}, \bar{\theta}_n)$, then by Corollary 3.18 the assertion follows immediately. Therefore, we assume that this is not true, and distinguish different cases.

`Case 1.` Let $|\overline{K}| = 1$, *i.e.*, $(\bar{x}_{a(n)}, \bar{\theta}_n) \in \mathrm{int}(F_{\bar{k}})$ with $\overline{K} = \{\bar{k}\}$. From Lemma 4.4 it follows $\widehat{K}^\nu = \{\bar{k}\}$ for all $\nu \geq \check{\nu}$, thus $(\widehat{x}^{i}_{a(n)}, \widehat{\theta}^{i_\nu}_n) \in \mathrm{int}(F_{\bar{k}})$ as well. According to Corollary 3.18, for each ν, the obtained deep Lagrangian cut supports $\mathrm{epi}(\overline{\mathrm{co}}(\underline{Q}_n))$ at $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$. The assertion then follows from Lemma 4.6.

`Case 2.` Let $|\overline{K}| > 1$, *i.e.*, $(\bar{x}_{a(n)}, \bar{\theta}_n) \in \mathrm{bd}(F_k)$ for all $k \in \overline{K}$. For any $\nu \geq \check{\nu}$ there are two possible sub-cases.

`Sub-case i).` The solution $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ to the projection problem (15) satisfies $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n) \in \mathrm{int}(F_k)$ for some $k \in \bar{K}$. Then, the assertion follows from Lemma 4.6.

`Sub-case ii).` The solution $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ to the projection problem (15) satisfies $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n) \in \mathrm{bd}(F_{k_\nu})$ for all $k^\nu \in \widehat{K}^\nu$, with $\widehat{K}^\nu \subseteq \overline{K}$. Then, $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ and $(\bar{x}_{a(n)}, \bar{\theta}_n)$ are still located on a joint sub-facet (face) $\mathfrak{F}$ of $\mathrm{epi}(\overline{\mathrm{co}}(\underline{Q}_n))$. Additionally, Lemma 4.5 implies that for all $\nu \geq \check{\nu}$, $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n)$ is not a vertex of $\mathrm{epi}(\overline{\mathrm{co}}(\underline{Q}_n))$. Therefore, we have $(\widehat{x}^{i_\nu}_{a(n)}, \widehat{\theta}^{i_\nu}_n) \in \mathrm{relint}(\mathfrak{F})$. Again, the assertion follows from Lemma 4.6. □

B.13 Proof of Theorem 4.8

Proof. We start with considering leaf nodes $n \in N \setminus \tilde{N}$, where $\Psi^{i+1}_m = \emptyset$ is fixed by definition. For any fixed $x_{a(n)} \in X_{a(n)}$, according to Lemma 4.7, after finitely many steps, a cut supporting $\mathrm{epi}(\overline{\mathrm{co}}(\underline{Q}^{i+1}_n)(\cdot))$ at $(x^i_{a(n)}, \theta^i_n)$ is obtained. Since $\underline{Q}^{i+1}_n(\cdot) \equiv Q_n(\cdot)$ for the leaf nodes and $\overline{\mathrm{co}}(Q_n)(x_{a(n)}) = Q_n(x_{a(n)})$ for all $x_{a(n)} \in X_{a(n)} = \{0,1\}^{d_{a(n)}}$, this implies that the cut supports $\mathrm{epi}(Q_n)$ at $(x^i_{a(n)}, \theta^i_n)$ (*tight* cut). As $X_{a(n)}$ is a finite set and as Assumption 4 is satisfied, it follows (i) that only finitely many different cuts can be generated, and thus that only finitely many different realizations of Ψ^{i+1}_n exist, (ii) that after finitely many steps these realizations become exact at all x^i_n computed in the forward pass. Note that not all of these realizations have to be generated in NBD, though, but that it is also possible that an optimal x^*_n has been reached before (all possible) cuts have been generated at $x_n \neq x^*_n$.

For each realization of Ψ^{i+1}_n, the same reasoning as above can now be applied to the ancestor nodes of the leaf nodes. The assertion then follows by induction over the whole scenario tree. □

C Nested Benders Decomposition

In Algorithm 1, we provide a description of the NBD algorithm.

Algorithm 1 Nested Benders Decomposition with the new cut generation framework

```
Require: Scenario tree $\mathcal{T} = (\mathcal{N}, \mathcal{E})$, tolerance $\varepsilon > 0$, normalization functions $g_n(\cdot)$ for all $n \in \mathcal{N}$.
 1: Initialization: bounds $\underline{v}^0 \leftarrow -\infty$, $\overline{v}^0 \leftarrow +\infty$, initial approximations $\Psi_n^1$ for all $n \in \mathcal{N}$ and
    iteration counter $i \leftarrow 0$.
 2: while $\overline{v}^0 - \underline{v}^0 > \varepsilon$ do
 3:     Set $i \leftarrow i + 1$.
 4:     Solve subproblem (6) for the root node $r$ to obtain a lower bound $\underline{v}^i$. Store the compo-
        nents $\big(x_r^i, y_r^i, (\theta_m^i)_{m \in \mathcal{C}(r)}\big)$ of the solution.
 5:     for stages $t = 1, \dots, T$ do
 6:         for nodes $n \in \mathcal{N}_t$ do
 7:             Solve subroblem (6) associated with function $\underline{Q}_n^i(x_{a(n)}^i)$ and store the components
                $\big(x_n^i, y_n^i, (\theta_m^i)_{m \in \mathcal{C}(n)}\big)$ of the solution.                ▷ Forward pass
 8:         end for
 9:     end for
10:     Obtain an upper bound as $\overline{v}^i = \sum_{n \in \mathcal{N}} f_n(x_n^i, y_n^i)$.
11:     for stages $t = T, \dots, 2$ do                                                  ▷ Backward pass
12:         for nodes $n \in \mathcal{N}_t$ do
13:             for children $m \in \mathcal{C}(n)$ do
14:                 Solve the normalized Lagrangian dual (14) for $(x_n^i, \theta_m^i)$ and $g_n(\cdot)$ to compute
                    a cut according to formula (13).
15:                 Update $\Psi_m^i$ to $\Psi_m^{i+1}$ in node $n$ using this cut.
16:             end for
17:         end for
18:     end for
19: end while
```

D Computational Results

D.1 CLSP with Binarization

The full computational results for our experiments of CLSP with state binarization are depicted in Tables 3-6. The table columns contain the number of stages, the used cut generation approach, the best lower bound obtained by SDDiP, a simulated statistical upper bound computed after termination of SDDiP (we report the upper limit of the computed confidence interval), the number of iterations, the time in seconds, the average time per iteration and the average number of iterations required in the level bundle method to solve the Lagrangian dual per iteration. We should note that in some cases, the simulation did not yield an upper bound estimate due to numerical issues.

In Table 3 we present the results for instances with $T = 4$, $T = 6$ and $T = 10$ stages and a time limit of 3 hours, 4 hours and 5 hours respectively. For $T = 4$, we also solve the deterministic equivalent with Gurobi for comparison.

In Table 4 we present the results for $T = 16$ stages and 8 hours of run time.

In Table 5 we present the results for runs that combine `SB` and different types of Lagrangian cuts.

In Table 6 we present the results for runs that use the Chen-Luedtke approach for a dual space restriction, again combining `SB` and different types of Lagrangian cuts.

D.2 CLSP without Binarization

For our tests of CLSP without state binarization, the full results are stated in Table 7.

Table 3: SDDiP results for CLSP with state binarization for $T = 4$, $T = 6$ and $T = 10$.

Method	Best LB	Stat. UB	Gap [%]	# Iter.	Time [s]	Time/It [s]	Lag-It/It
			$\boldsymbol{T = 4}$				
Det. Equiv.	1503.0	1542.3	3				
`B` (single)	803.2	1595.5	50	85	6	0	-
`B` (multi)	804.5	1602.6	50	59	5	0	-
`SB` (single)	1155.8	**1572.9**	27	32	16	1	-
`SB` (multi)	1187.1	1608.7	26	73	49	1	-
`L` (single)	681.8	1766.4	61	109	10800	99	49
`L` (multi)	702.6	1736.6	60	27	10908	404	51
ℓ^1-deep	1478.3	1582.5	**7**	39	11196	272	67
$\ell^{1\infty}$-deep	1462.9	1649.4	11	38	10944	288	63
ℓ^{∞}-deep	660.5	1854.1	64	43	11160	259	30
ℓ^{∞}-deep (MNC)	1049.0	1720.7	39	54	10944	203	77
`LN-Mid`	**1502.2**	-	-	38	11088	351	80
`LN-In-Out`	1017.0	1742.3	42	21	12024	573	97
`LN-Eps`	1499.0	-	-	28	11556	413	116
`LN-Relint`	1500.4	1621.2	8	33	11088	336	88
			$\boldsymbol{T = 6}$				
`B` (single)	1354.5	2854.2	53	479	69	0	-
`B` (multi)	1355.0	2889.4	53	165	37	0	-
`SB` (single)	2077.9	**2825.1**	26	46	41	1	-
`SB` (multi)	2093.4	2838.7	26	251	517	2	-
`L` (single)	669.7	3095.0	78	90	14580	162	50
`L` (multi)	682.5	3111.7	78	24	14688	612	130
ℓ^1-deep	2414.1	2975.4	19	34	14508	427	62
$\ell^{1\infty}$-deep	2375.8	2984.5	20	35	15012	429	59
ℓ^{∞}-deep (MNC)	1411.1	3003.5	53	48	14868	310	58
`LN-Mid`	**2500.3**	2892.9	**14**	24	14904	621	101
`LN-In-Out`	1162.2	3056.4	62	18	14940	830	97
`LN-Eps`	2454.2	2865.1	14	20	14580	729	135
`LN-Relint`	2477.0	2871.8	**14**	22	14616	664	108
			$\boldsymbol{T = 10}$				
`B` (single)	2165.3	5120.8	58	136	26	0	-
`B` (multi)	2183.6	5168.8	58	334	197	0	-
`SB` (single)	3377.1	4949.8	32	89	122	1	-
`SB` (multi)	3472.9	**4942.9**	30	206	1066	5	-
`L` (single)	616.4	5466.6	89	69	18144	263	51
`L` (multi)	682.4	5443.1	88	20	18720	936	54
ℓ^1-deep	3782.5	-	-	38	19260	507	47
$\ell^{1\infty}$-deep	3862.7	-	-	37	18504	500	47
ℓ^{∞}-deep (MNC)	2004.3	5256.8	62	44	18576	422	43
`LN-Mid`	**4222.2**	5168.9	**18**	18	18216	1012	110
`LN-Eps`	3642.8	5223.9	30	18	18360	1020	144
`LN-Relint`	4145.1	-	-	17	20232	1190	119

Table 4: SDDiP results for CLSP with state binarization for $T = 16$.

Method	Best LB	Stat. UB	Gap [%]	# Iter.	Time [s]	Time/Iter [s]	Lag-Iter/Iter
				$\boldsymbol{T = 16}$			
`B` (single)	3917.5	9012.6	57	224	74	0	-
`B` (multi)	3937.2	9008.2	56	254	274	1	-
`SB` (single)	5913.7	**8673.6**	32	194	493	3	-
`SB` (multi)	6030.2	8676.8	31	585	9252	16	-
`L` (single)	607.4	9524.5	94	63	28836	458	51
`L` (multi)	651.6	9444.7	93	19	31176	1641	55
ℓ^1-deep	3547.5	9756.3	64	42	28944	689	34
$\ell^{1\infty}$-deep	3353.0	9997.3	67	37	31248	845	16
ℓ^∞-deep (MNC)	2563.3	9917.7	74	52	29664	571	39
`LN-Mid`	**6910.7**	9125.9	**24**	18	32112	1784	119
`LN-Eps`	6014.5	9195.2	35	16	32040	2003	143
`LN-Relint`	6105.6	9085.9	33	15	29772	1985	123

Table 5: SDDiP results for CLSP using Lagrangian cuts combined with SB cuts.

Method	Best LB	Stat. UB	Gap [%]	# Iter.	Time [s]	Time/Iter [s]	Lag-Iter/Iter
				$\boldsymbol{T = 10}$			
`L` (single)	3475.5	**5003.9**	31	132	18072	136	50
`L` (multi)	3483.0	5014.4	31	51	18360	360	51
ℓ^1-deep	3804.1	5161.6	26	35	19656	562	88
$\ell^{1\infty}$-deep	3914.2	5233.7	25	36	19656	546	83
ℓ^∞-deep (MNC)	3551.5	5042.8	30	49	18108	340	180
`LN-Mid`	4052.9	5130.9	21	31	20484	661	115
`LN-Eps`	**4087.7**	5092.9	**20**	31	21096	681	141
`LN-Relint`	4034.3	5135.4	21	31	19944	643	119
				$\boldsymbol{T = 16}$			
`L` (single)	5907.1	**8728.4**	32	136	28944	213	47
`L` (multi)	5922.0	8750.8	32	52	30096	579	48
ℓ^1-deep	6477.5	9078.4	28	33	29916	907	90
$\ell^{1\infty}$-deep	6417.4	9024.6	29	34	30708	903	87
ℓ^∞-deep (MNC)	5991.3	8889.8	33	45	28800	640	191
`LN-Mid`	6675.7	9102.9	27	29	31680	1092	127
`LN-Eps`	6705.0	8947.8	**25**	29	30528	1053	146
`LN-Relint`	**6705.3**	8995.5	26	29	31284	1079	134

Table 6: SDDiP results for CLSP with $T = 10$ and $T = 16$ using the `CL` approach.

Method	Best LB	Stat. UB	Gap [%]	# Iter.	Time [s]	Time/Iter [s]	Lag-Iter/Iter
			$\boldsymbol{T = 10}$				
γ	4395.5	5224.4	**16**	74	18396	249	17
ℓ^1-deep	4344.5	-	-	53	18108	341	41
ℓ^∞-deep (MNC)	4249.9	5270.7	19	34	18216	536	383
`LN-Mid`	**4400.0**	**5217.6**	**16**	52	18288	352	34
`LN-Eps`	4389.3	5240.2	**16**	52	18684	359	38
`LN-Relint`	4371.4	5235.3	17	51	18036	354	34
			$\boldsymbol{T = 16}$				
γ	**7565.5**	9067.7	**17**	65	29160	449	18
ℓ^1-deep	7254.2	9095.4	20	49	29160	589	42
ℓ^∞-deep (MNC)	7248.9	9147.6	21	33	29808	903	432
`LN-Mid`	7521.0	9061.8	**17**	52	29340	611	38
`LN-Eps`	7510.8	9080.5	**17**	52	29232	622	40
`LN-Relint`	7543.3	**9051.4**	**17**	49	30204	617	38

Table 7: SDDiP results for CLSP with 10 state variables and no binary approximation.

Method	Best LB	Stat. UB	Gap [%]	# Iter.	Time [s]	Time/Iter [s]	Lag-Iter/Iter
			$T = 16$, One type of cut				
`B` (single)	15190	29701	49	193	57	0	-
`B` (multi)	15199	29747	49	207	189	1	-
`SB` (single)	20373	29601	31	111	211	2	-
`SB` (multi)	21045	**29343**	28	281	3123	11	-
`L` (single)	13676	32003	57	84	10832	129	42
`L` (multi)	19199	31437	39	27	10971	406	45
ℓ^1-deep	23486	30367	23	34	11061	325	30
$\ell^{1\infty}$-deep	**23887**	30111	**21**	32	11472	359	33
ℓ^{∞}-deep (MNC)	23224	30074	23	26	10839	417	48
`LN-Mid`	19820	30450	35	19	11922	628	88
`LN-Eps`	22027	30599	28	22	10820	492	63
`LN-Relint`	8857	31188	72	19	11484	604	96
`LN-Conv`(50)	22588	30438	26	16	11132	696	98
`LN-Conv`(75)	22931	30206	24	17	10829	637	88
`LN-Conv`(90)	23215	30235	23	19	11342	597	79
`LN-Conv`(99)	22639	30374	26	22	11025	501	62
			$T = 16$, Combination with `SB` cuts				
`L` (single)	21129	**30040**	30	92	10809	118	42
`L` (multi)	22271	30540	27	37	11312	306	44
ℓ^1-deep	**23235**	30342	**23**	35	11571	331	51
$\ell^{1\infty}$-deep	23197	30049	**23**	35	11364	325	49
ℓ^{∞}-deep (MNC)	22975	30210	24	36	11025	306	92
`LN-Mid`	22638	30413	26	30	12231	408	107
`LN-Eps`	22798	30382	25	32	10800	338	66
`LN-Relint`	22814	30303	25	29	11068	382	113
`LN-Conv`(50)	22641	30255	25	30	11941	398	110
`LN-Conv`(75)	22804	30120	24	31	12145	392	96
`LN-Conv`(90)	23162	30097	**23**	32	11590	362	79
`LN-Conv`(99)	23152	30118	**23**	33	11833	359	67
			$T = 16$, Combination with `SB` cuts, `CL` approach				
γ	23715	**29894**	**21**	72	11028	153	23
ℓ^1-deep	**24036**	30231	**21**	48	11006	229	51
ℓ^{∞}-deep (MNC)	23803	30253	**21**	49	11051	226	82
`LN-Mid`	23114	30184	23	36	11118	309	98
`LN-Eps`	23566	30252	22	41	10822	264	65
`LN-Relint`	22938	30096	24	36	11288	314	99
`LN-Conv`(50)	22817	30344	25	31	11870	383	91
`LN-Conv`(75)	22701	30287	25	31	10974	354	82
`LN-Conv`(90)	22951	30344	24	32	10838	339	71
`LN-Conv`(99)	22902	30085	24	33	11084	336	61

References

[1] S. Ahmed, F. G. Cabral, and B. Freitas Paulo da Costa. Stochastic Lipschitz dynamic programming. *Mathematical Programming*, 191:755–793, 2022.

[2] D. Ávila, A. Papavasiliou, and N. Löhndorf. Batch learning SDDP for long-term hydrothermal planning. *IEEE Transactions on Power Systems*, 39(1):614–627, 2024.

[3] E. Balas and P. L. Ivanescu. On the generalized transportation problem. *Management Science*, 11(1):188–202, 1964.

[4] D. P. Bertsekas. *Convex Optimization Theory.* Athena Scientific, 2009.

[5] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: a fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[6] J. R. Birge. Solution methods for stochastic dynamic linear programs. Technical report, Stanford University CA Systems Optimization Lab, 1980.

[7] J. R. Birge and F. Louveaux. *Introduction to stochastic programming.* Springer Series in Operations Research and Financial Engineering. Springer Science & Business Media, 2nd edition, 2011.

[8] E. A. Boyd. Fenchel cutting planes for integer programs. *Operations Research*, 42(1):53–64, 1994.

[9] R. Brandenberg and P. Stursberg. Refined cut selection for benders decomposition: applied to network capacity expansion problems. *Mathematical Methods of Operations Research*, 94:383–412, 2021.

[10] F. Cadoux. Computing deep facet-defining disjunctive cuts for mixed-integer programming. *Mathematical Programming, Ser. A*, 122:197–223, 2010.

[11] R. Chen and J. Luedtke. On generating Lagrangian cuts for two-stage stochastic integer programs. *INFORMS Journal on Computing*, 2022.

[12] M. Conforti and L. A. Wolsey. "Facet" separation with one linear program. *Mathematical Programming, Ser. A*, 178:361–380, 2019.

[13] G. Cornuéjols and C. Lemaréchal. A convex-analysis perspective on disjunctive cuts. *Mathematical Programming, Ser. A*, 106:567–586, 2006.

[14] S. S. Dey and M. Molinaro. Theoretical challenges towards cutting-plane selection. *Mathematical Programming*, 170:237–266, 2018.

[15] O. Dowson and L. Kapelevich. SDDP.jl: a Julia package for stochastic dual dynamic programming. *INFORMS Journal on Computing*, 33(1):27–33, 2020.

[16] I. Dunning, J. Huchette, and M. Lubin. JuMP: a modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017. doi: 10.1137/15m1020575.

[17] M. Fischetti, D. Salvagnin, and A. Zanette. A note on the selection of Benders' cuts. *Mathematical Programming, Ser. B*, 124:175–182, 2010.

[18] C. Füllner, X. A. Sun, and S. Rebennack. On Lipschitz regularization and Lagrangian cuts in multistage stochastic mixed-integer linear programming. Preprint, 2024.

[19] A. M. Geoffrion. Lagrangean relaxation for integer programming. In M.L. Balinski, editor, *Approaches to integer programming*, pages 82–114. Springer Berlin Heidelberg, Berlin, Heidelberg, 1974. ISBN 978-3-642-00740-8.

[20] M. Guignard. Lagrangean relaxation. *TOP*, 11(2):151–228, 2003.

[21] M. Hosseini and J. G. Turner. Deepest cuts for Benders decomposition. Preprint on arXiv, 2021.

[22] T.L. Magnanti and R.T. Wong. Accelerating benders decomposition: algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

[23] N. Papadakos. Practical enhancements to the Magnanti–Wong method. *Operations Research Letters*, 36:444–449, 2008.

[24] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52(1-3):359–375, 1991.

[25] R. Rahmaniani, S. Ahmed, T. G. Crainic, M. Gendreau, and W. Rei. The Benders dual decomposition method. *Operations Research*, 68(3):878–895, 2020.

[26] K. Seo, S. Joung, C. Lee, and S. Park. A closest Benders cut selection scheme for accelerating the Benders decomposition algorithm. *INFORMS Journal on Computing*, pages 1–24, 2022.

[27] H. D. Sherali and B. J. Lunday. On generating maximal nondominated Benders cuts. *Annals of Operations Research*, 210:57–72, 2013.

[28] P. M. Stursberg. *On the mathematics of energy system optimization.* PhD thesis, Technische Unviersität München, 2019.

[29] W. W. Trigeiro, L. J. Thomas, and J. O. McClain. Capacitated lot sizing with setup times. *Management Science*, 35(3):353–366, 1989.

[30] S. Zhang and X. A. Sun. Stochastic dual dynamic programming for multistage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 2022.

[31] J. Zou, S. Ahmed, and X. A. Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.

D.1 Supplementary material

We provide some supplementary material that was part of our research on Lagrangian cuts, but omitted in the submitted publication for reasons of space.

Supplementary Material "A new framework to generate Lagrangian cuts in multistage stochastic mixed-integer programming"

Christian Füllner, X. Andy Sun, Steffen Rebennack

1 Using a Single-cut Approach in the New Framework

The unified cut generation framework presented in Sect. 3 comes at the cost of having to use a multi-cut approach to approximate the epigraphs of the value functions separately. While this approach often yields better approximations than a single-cut approach, it also tends to be computationally detrimental because too many cuts have to be added to the subproblems. Therefore, in multistage stochastic programming usually a single-cut approach is preferred.

The main idea of a single-cut approach is to still generate cuts for each subproblem separately, thus exploiting decomposability of the multistage problem, but to then aggregate these cuts. There are a few reasons why such an aggregation is not possible in the unified framework. First, the cut coefficient π_{n0} in the Lagrangian cut formula (13) does not allow for a straightforward aggregation due to different scaling of cuts. At least for optimality cuts satisfying $\pi_{n0} > 0$, however, this issue may be avoided by first dividing by π_{n0} before aggregating, see Remark 3.8. Second, and more critical, setting up the Lagrangian dual problems (10) requires an incumbent $(x^i_{a(n)}, \theta^i_n)$. This implies that we need to compute a value θ^i_n for each node $n \in \overline{\mathcal{N}}$ in the forward pass, whereas a single-cut approach only provides an aggregated value.

It is still possible to combine a single-cut approach with the ideas of the proposed unified cut generation framework, as we show in the remainder of this section. However, for the above reasons, it is not possible while retaining complete decomposability of the subproblems at each stage t. Instead, we can only achieve partial decomposability.

For this section, we assume all assumptions made in Sect. 2 to hold as well. For notational simplicity, we mostly use the same symbols as for the multi-cut approach instead of introducing separate notation.

1.1 An Epigraph Perspective on Cut Generation

For all $n \in \overline{\mathcal{N}}$, we define by

$$\mathcal{Q}_{\mathcal{C}(n)}(x_{a(n)}) := \sum_{m \in \mathcal{C}(n)} p_{nm} Q_m(x_n)$$

the *expected value functions* starting from node n. Using the single-cut approach, our aim is to approximate $\text{epi}(\mathcal{Q}_{\mathcal{C}(n)})$ using polyhedral approximations $\Psi_{\mathcal{C}(n)}$ (recall the definition of epigraphs from (4). Note that under Assumption 1, if we choose the same set $Z_{a(n)}$ at all nodes $n \in \mathcal{N}(t)$, we have $\operatorname{dom}(\mathcal{Q}_{\mathcal{C}(n)}) = \operatorname{dom}(Q_m)$ for all $m \in \mathcal{C}(n)$.

In the same vein as in Sect. 3, we define the set

$$\widehat{\mathcal{W}}_n^{i+1} := \Big\{ \big(x_n, y_n, z_n, \theta_{\mathcal{C}(n)}\big) \ : \ (x_n, y_n, z_n) \in \mathcal{F}_n, \ z_n \in Z_{a(n)}, \ (x_n, \theta_{\mathcal{C}(n)}) \in \Psi_{\mathcal{C}(n)}^{i+1} \Big\},$$

$\lambda_n := \big(x_n, y_n, \theta_{\mathcal{C}(n)}\big)$ and $\widehat{c}_n^\top \lambda_n := f_n(x_n, y_n) + \theta_{\mathcal{C}(n)}$. Then, the approximate subproblems and approximate value functions can be expressed as

$$\underline{Q}_n^{i+1}(x_{a(n)}^i) = \min_{\lambda_n, z_n} \left\{ \widehat{c}_n^\top \lambda_n \ : \ (\lambda_n, z_n) \in \widehat{\mathcal{W}}_n^{i+1}, \ z_n = x_{a(n)}^i \right\}. \tag{32}$$

Analogously, we define the expected approximate value functions as

$$\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(x_{a(n)}) := \sum_{m \in \mathcal{C}(n)} p_{nm} \underline{Q}_m^{i+1}(x_n). \tag{33}$$

1.2 A Feasibility Problem for the Epigraph

Consider some node $n \in \widetilde{\mathcal{N}}$ and all successor nodes $\mathcal{C}(n)$. Given an incumbent $(x_n^i, \theta_{\mathcal{C}(n)}^i)$, we formulate a feasibility problem to verify if it satisfies $(x_n^i, \theta_{\mathcal{C}(n)}^i) \in \operatorname{epi}(\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1})$. In its most simple form this can be expressed as

$$\begin{aligned} v_{\mathcal{C}(n)}^{f,i+1}(x_n^i, \theta_{\mathcal{C}(n)}^i) := \min_{x_n, \theta_{\mathcal{C}(n)}} \Big\{ 0 \ &: x_n = x_n^i, \theta_{\mathcal{C}(n)} = \theta_{\mathcal{C}(n)}^i, \\ &(x_n^i, \theta_{\mathcal{C}(n)}^i) \in \operatorname{epi}(\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}) \Big\}. \end{aligned} \tag{34}$$

Using the definition of $\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1}(\cdot)$ from (33), this can be converted to

$$\begin{aligned} v_{\mathcal{C}(n)}^{f,i+1}(x_n^i, \theta_{\mathcal{C}(n)}^i) := \min_{(\lambda_m), (z_m)} \Big\{ 0 \ &: (\lambda_m, z_m) \in \widehat{\mathcal{W}}_m^{i+1}, z_m = x_m^i, m \in \mathcal{C}(n), \\ &\theta_{\mathcal{C}(m)}^i \geq \sum_{m \in \mathcal{C}(n)} p_{nm} \widehat{c}_m^\top \lambda_m, \Big\}, \end{aligned} \tag{35}$$

where $(\lambda_m) := (\lambda_m)_{m \in \mathcal{C}(n)}, (z_m) := (z_m)_{m \in \mathcal{C}(n)}$.

Importantly, and a crucial difference to the multi-cut approach from Sect. 3, here the feasibility problem contains variables and constraints that are related to several nodes $m \in \mathcal{C}(n)$, and thus epigraphs $\operatorname{epi}(\underline{Q}_m^{i+1})$, and not only to one single node and epigraph. In other words, since we use aggregated cuts to approximate $\operatorname{epi}(\mathcal{Q}_{\mathcal{C}(n)})$ by a single set of cuts, the obtained feasibility problem in the unified framework is no longer decomposed by subproblems. Instead, we obtain one single feasibility problem per stage.

Additionally, we notice that including constraints $(\lambda_m, z_m) \in \widehat{\mathcal{W}}_m^{i+1}$ for all $m \in \mathcal{C}(n)$ means that for each such m a separate set of cut approximations $\Psi_{\mathcal{C}(m)}^{i+1}$ are included (in the case of stagewise independent uncertainty as in SDDiP, the same cut constraints are included several times, but have to be satisfied by different variables). This contradicts the initial aim of the single-cut approach to reduce the number of cuts occurring in the subproblems. Fortunately, when we consider a Lagrangian relaxation of problem (35), we achieve a partial decomposability, so that each subproblem that is solved in the actual cut generation process only contains a single set of cuts. We shows this in the next subsection.

1.3 Lagrangian Cuts in the New Framework

As in the multi-cut approach, we apply a Lagrangian relaxation where we relax the copy and epigraph constraints. Importantly, here we have to relax $|\mathcal{C}(n)|$ copy constraints. This yields the relaxation

$$\begin{aligned}\mathfrak{L}_{\mathcal{C}(n)}^{i+1}\big((\pi_m), \gamma_{\mathcal{C}(n)}\big) := \min_{(\lambda_m),(z_m)} \Bigg\{ \sum_{m\in\mathcal{C}(n)} \pi_m^\top z_m + \gamma_{\mathcal{C}(n)} \Bigg(\sum_{m\in\mathcal{C}(n)} p_{nm} \widehat{c}_m^\top \lambda_m \Bigg) \ : \\ (\lambda_m, z_m) \in \widehat{\mathcal{W}}_m^{i+1}, m \in \mathcal{C}(n) \Bigg\},\end{aligned} \tag{36}$$

where we omitted the constant term in the objective.

The corresponding Lagrangian dual problem is

$$\begin{aligned}\widehat{v}_{\mathcal{C}(n)}^{D,i+1}(x_n^i, \theta_{\mathcal{C}(n)}^i) := \max_{(\pi_m),\gamma_{\mathcal{C}(n)}} \Big\{ \mathfrak{L}_{\mathcal{C}(n)}^{i+1}\big((\pi_m), \gamma_{\mathcal{C}(n)}\big) - \sum_{m\in\mathcal{C}(n)} \pi_m^\top x_n^i - \gamma_{\mathcal{C}(n)}\theta_{\mathcal{C}(n)} \ : \\ \gamma_{\mathcal{C}(n)} \geq 0 \Big\}.\end{aligned} \tag{37}$$

Compared to the multi-cut version, we only have to solve one Lagrangian dual problem (37) for all $m \in \mathcal{C}(n)$ together instead of one Lagrangian dual problem for each. Considering how computationally challenging it is to solve Lagrangian dual problems, this is favorable. On the other hand, the obtained Lagrangian dual problem is not decomposed by nodes, thus much larger, and possibly more time-consuming to solve.

Fortunately, at least some partial decomposability can be achieved. Let

$$\mathfrak{L}_m^{i+1}\big((\pi_m), \gamma_{\mathcal{C}(n)}\big) := \min_{\lambda_m, z_m} \Big\{ \pi_m^\top z_m + \gamma_{\mathcal{C}(n)} p_{nm} \widehat{c}_m^\top \lambda_m \ : \ (\lambda_m, z_m) \in \widehat{\mathcal{W}}_m^{i+1} \Big\}. \tag{38}$$

Then, the overall dual function defined in (36) can be equivalently written as

$$\mathfrak{L}_{\mathcal{C}(n)}^{i+1}\big((\pi_m), \gamma_{\mathcal{C}(n)}\big) := \sum_{m\in\mathcal{C}(n)} \mathfrak{L}_m^{i+1}\big((\pi_m), \pi_m\big), \tag{39}$$

which allows for a solution of the inner problem in the Lagrangian dual for each node $m \in \mathcal{C}(n)$ separately. However, we notice that all inner problems contain the same dual multiplier $\gamma_{\mathcal{C}(n)}$, so the outer problem (37) cannot be decoupled in a straightforward way.

The Lagrangian dual (37) has the same properties that we already discussed in Theorem 3.5 for the multi-cut case.

Theorem 1.1. *Under Assumptions 1 and 2, for the Lagrangian dual* (37) *it holds:*

(i) The dual function $\mathfrak{L}_{\mathcal{C}(n)}^{i+1}(\,\cdot\,)$ *is piecewise linear concave in* $\big((\pi_m), \pi_m\big)$.

(ii) Its optimal value $\widehat{v}_{\mathcal{C}(n)}^{D,i+1}(x_n^i, \theta_{\mathcal{C}(n)}^i)$ *satisfies*

$$\widehat{v}_{\mathcal{C}(n)}^{D,i+1}(x_n^i, \theta_{\mathcal{C}(n)}^i) = \begin{cases} 0, & \textit{if } (x_n^i, \theta_{\mathcal{C}(n)}^i) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{\mathcal{Q}}_{\mathcal{C}(n)}^{i+1})) \\ +\infty, & \textit{else.} \end{cases}$$

Based on these properties, analogously to Sect. 3, a valid Lagrangian cut can be defined.

Definition 1.2. *For all $n \in \overline{\mathcal{N}}$ and some multipliers $\big((\pi_m), \gamma_{\mathcal{C}(n)}\big)$, a Lagrangian cut is given by*

$$\begin{aligned} &\gamma^i_{\mathcal{C}(n)} \theta_{\mathcal{C}(n)} + \sum_{m \in \mathcal{C}(n)} (\pi^i_m)^\top x_n \geq \mathfrak{L}^{i+1}_{\mathcal{C}(n)}\big((\pi^i_m), \gamma^i_{\mathcal{C}(n)}\big) \\ \Leftrightarrow \quad &\gamma^i_{\mathcal{C}(n)} \theta_{\mathcal{C}(n)} + (\pi^i_{\mathcal{C}(n)})^\top x_n \geq \mathfrak{L}^{i+1}_{\mathcal{C}(n)}\big((\pi^i_m), \gamma^i_{\mathcal{C}(n)}\big) \end{aligned} \tag{40}$$

with $\pi^i_{\mathcal{C}(n)} := \sum_{m \in \mathcal{C}(n)} \pi^i_m$.

This type of cut is valid for any feasible $\big((\pi_m), \gamma_{\mathcal{C}(n)}\big)$ in (37).

Lemma 1.3. *Under Assumptions 1 and 2, for any $\big((\pi_m), \gamma_{\mathcal{C}(n)}\big) \in \mathbb{R}^{d_{a(n)}|\mathcal{C}(n)|} \times \mathbb{R}^+$ the Lagrangian cut* (40) *is satisfied by all $(x_n, \theta_{\mathcal{C}(n)}) \in \operatorname{epi}(\overline{\operatorname{co}}(\underline{\mathcal{Q}}^{i+1}_{\mathcal{C}(n)}))$, and thus, by all $(x_n, \theta_{\mathcal{C}(n)}) \in \operatorname{epi}(\overline{\operatorname{co}}(\mathcal{Q}_{\mathcal{C}(n)}))$.*

The presented approach at least allows to decompose the inner problem in solution methods for the Lagrangian dual, such as subgradient methods or cutting-plane methods. Whether it is sufficient to obtain performance gains compared to the multi-cut version is problem-dependent, since the dual problem grows in dimension. We did not perform any computational tests in this regard so far, so a computational evaluation is left for future research.

Paper E

Nonlinear cut-sharing in stochastic dual dynamic programming for log-linear autoregressive uncertainty in the right-hand side

Christian Füllner[a], Steffen Rebennack[a]

[a] *Karlsruhe Institute of Technology (KIT), Institute for Operations Research, Stochastic Optimization, Karlsruhe, Germany*

Submitted to *Operations Research*, suggested citation:
Füllner C., Rebennack S. (2024), Nonlinear cut-sharing in stochastic dual dynamic programming for log-linear autoregressive uncertainty in the right-hand side.

Nonlinear Cut-sharing in Stochastic Dual Dynamic Programming for Log-linear Autoregressive Uncertainty in the Right-hand Side

Christian Füllner[1*] and Steffen Rebennack[1]

[1]Institute for Operations Research (IOR), Stochastic Optimization (SOP), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[*]Address correspondence to: christian.fuellner@kit.edu

Abstract

We consider the generation of cuts in stochastic dual dynamic programming (SDDP) for multistage stochastic linear programming problems with stagewise dependent uncertainty in the right-hand side described by a nonlinear autoregressive process. We first derive cut formulas for the general nonlinear autoregressive process. For these general cuts, the computational tractability becomes a major challenge for SDDP-type algorithms. We are, however, able to develop closed-form cut formulas for the special case of log-linear autoregressive processes. The obtained cuts are in general non-convex in the history of the stochastic process, and thus can be used to tightly approximate the occurring non-convex value functions. As the cuts are still linear in all decision variables, they can be directly incorporated into the subproblems in SDDP without compromising their linearity. If solvers do not allow for this, our formulas can be used to adapt the intercept of a given cut to a scenario at hand in a computationally tractable way. In this sense, cuts generated for one specific scenario are *shared* with other scenarios. Our findings are supported by illustrative examples and by a computational study of a hydrothermal scheduling problem.

1 Introduction

1.1 Motivation

In many decision-making situations, multiple subsequent decisions have to be taken while at least some of the decision-relevant data are uncertain. Often, the premise is that at the first stage some decisions have to be taken before any uncertain data are revealed and to hedge against the existing uncertainty. At later stages, then corrective decisions can be made under the knowledge that some part of the uncertain data has been realized. Typically, the aim is to determine an optimal policy (sequence of decision rules) *in expectation*, and only linear objectives and constraints are considered. Formalizing this constitutes the class of *multistage stochastic linear programming problems* (MSLPs).

Stochastic dual dynamic programming (SDDP), introduced by Pereira and Pinto in 1991 [20], is one of the state-of-the-art solution methods for this class of problems. It is proven to converge to an optimal policy for a given MSLP in finitely many iterations *almost surely.* Since its invention, SDDP-type algorithms have been applied to numerous case-studies in practice, and extended to broader problem classes such as convex problems [12], mixed-integer linear problems [33], mixed-integer nonlinear non-convex problems [10, 31], risk-averse problems [24] or distributionally robust settings [9]. Moreover, several proposals have been made to accelerate the algorithm, *e.g.*, [1, 5, 14, 29]. For a comprehensive overview on SDDP we refer to the review [11].

A crucial requirement of standard SDDP is the stagewise independence of the uncertainty in the data. However, in many applications, this assumption is not warranted. Therefore, the last three decades have seen a lot of research on efficiently extending SDDP to problems exhibiting stagewise dependent uncertainty, comprising approximations of the uncertainty by some Markov chain [18], combining SDDP with an incorporated Markov chain [21], using conditional cuts [30], and using saddle cuts [6] or dual variants of SDDP [15] for stagewise dependent uncertainty in the objective function.

To model uncertainty in the right-hand side of the constraints, especially autoregressive (AR) processes have gained a lot of attention, as they are often used in practice, *e.g.*, to model inflows in hydrothermal scheduling problems [4, 17, 19]. Whereas significant progress has been made for the cases of linear and convex AR processes, results on nonlinear non-convex AR processes are limited. So far, only very simple nonlinear non-convex processes, satisfying additivity or a specific notion of linearity, have been successfully incorporated into SDDP [16]. More general nonlinear processes have not been analyzed yet. These types of processes are practically important, though, for instance to describe uncertain wind power [32] or uncertain non-negative inflows [27].

In this paper, we therefore deal with the question of extending SDDP to problems where the uncertain data are modeled by nonlinear, possibly non-convex AR processes. One of the main challenges is that in this case the expected value functions, which are iteratively approximated by cutting-planes (also called *cuts*) within SDDP, are non-convex. For this reason, linear cuts are not sufficient for valid and tight approximations, and thus cannot guarantee convergence of SDDP. As a remedy, we show that instead, cuts can be generated that are linear in the original state variables, but non-convex in the history of the considered AR process, therefore allowing for valid and tight approximations of the non-convex expected value functions. Our paper is the first work proposing *nonlinear* cuts in this context.

For these general cuts, the computational tractability becomes a major challenge for SDDP-type algorithms, as they are in general hard to convert into closed-form expressions and often computationally prohibitive to evaluate. We are, however, able to develop closed-form cut formulas for the special class of log-linear (periodic) AR processes. This class is prominently used to model uncertain non-negative inflows within hydrothermal systems in practice [27]. We show that for this class, tractable closed-form expressions for non-convex cuts can be derived, which then can be incorporated within SDDP to approximate the expected value functions. We perform computational tests for a long-term hydrothermal scheduling problem to assess the performance of this version of SDDP.

1.2 Contribution

The key contributions of this paper are summarized below.

(1) We show how to recursively compute generic nonlinear cuts for the non-convex expected value functions if the uncertainty in the right-hand side of a multistage

stochastic linear program follows a general nonlinear AR process (see Sect. 3.2). We identify the computational challenges that come with evaluating these recursive cuts or with translating them into closed-form cut formulas, rendering their application within SDDP computationally intractable in general.

(2) We then analyze log-linear (periodic) AR processes as a special and practically relevant class of non-convex AR processes that has been considered in the context of SDDP before (see Sect. 4). We show that for this class, valid and tight non-convex cuts for the expected value functions can be derived (see Sect. 4.3). These cuts can be computed and evaluated using tractable closed-form expressions.

(3) This new type of cuts is non-convex in the history of the AR process, but linear in all decision variables. Therefore, it can be directly incorporated into the subproblems in SDDP without compromising their linearity. If solvers do not allow for this, our formulas can be used to adapt the intercept of a given cut to a scenario at hand, thus to *share* the cut with that particular scenario. The incorporation into SDDP is discussed in Sect. 4.4.

(4) Even under special conditions where both the expected value functions and our proposed nonlinear cuts become *convex*, these cuts provide a more accurate approximation than previously proposed linear cuts, as we show for an illustrative example in Sect. 4.3.

(5) We conduct computational experiments on a long-term hydrothermal scheduling (LTHS) problem with stochastic inflows. Our experiments show that, although coming with a non-negligible computational overhead, our proposed version of SDDP with non-convex cuts yields favorable policies compared to running SDDP using linearized inflow models. More precisely, assuming that the log-linear AR process provides a more accurate representation of the inflows, which is supported by historical data, we achieve a 7-10% total cost reduction on average in an out-of-sample simulation.

1.3 Structure

This paper is structured as follows. In Sect. 2, we formally introduce the considered problems and the AR processes describing the uncertainty. In Sect. 3, we discuss extensions of SDDP to uncertainty modeled by general nonlinear AR processes. After that, in Sect. 4, we dedicate to the special class of log-linear AR processes, derive closed-form cut formulas and discuss their integration into SDDP. To support our theoretical findings, in Sect. 5, we present computational experiments for a hydrothermal scheduling problem. We finish with a conclusion in Sect. 6.

Appendix A contains some longer proofs that are outsourced from the main paper. In Appendix B, we summarize the notation used throughout this paper. Finally, we provide an electronic companion (EC) with supplementary material, especially on our computational experiments. For comparison, we also present SDDP in its standard form in the EC.

2 Problem Formulation

We consider a general multistage stochastic linear programming problem with a finite number $T \in \mathbb{N}$ of stages, henceforth referred to as (MSLP). At each stage $t \in [T] :=$

$\{1,\dots,T\}$, decisions $x_t \in \mathbb{R}^{n_t}, n_t \in \mathbb{N}$, have to be taken, which satisfy a set of $L_t \in \mathbb{N}$ linear constraints defined by matrices $T_{t-1} \in \mathbb{R}^{L_{t-1}\times n_t}, W_t \in \mathbb{R}^{L_t \times n_t}$ and vectors $h_t \in \mathbb{R}^{L_t}$, and parametrized by x_{t-1}, with $T_0 \equiv 0, x_0 \equiv 0$. (MSLP) can be represented in the following nested form (for equivalent non-nested representations we refer to [11, 25]):

$$\begin{aligned} v^* := \min_{\substack{x_1 \geq 0 \\ W_1 x_1 = h_1}} c_1^\top x_1 + \mathbb{E}_{\xi_2|\xi_1} \Bigg[\min_{\substack{x_2 \geq 0 \\ T_1 x_1 + W_2 x_2 = h_2}} c_2^\top x_2 + \mathbb{E}_{\xi_3|\xi_{[2]}} \Bigg[\cdots \\ + \mathbb{E}_{\xi_T|\xi_{[T-1]}} \Bigg[\min_{\substack{x_2 \geq 0 \\ T_{T-1}x_{T-1} + W_T x_T = h_T}} c_T^\top x_T \Bigg]\Bigg]\Bigg] \end{aligned} \tag{1}$$

We assume that some of the data in (MSLP) are subject to uncertainty, which is revealed over time. The premise is that decisions x_1 are taken hedging against the full uncertainty, whereas at the following stages $t = 2,\dots,T$, recourse decisions x_t are taken after the uncertain data on stage t has realized. The aim is to compute the optimal value and policy (sequence of decisions given a realization of the uncertainty) in expectation.

The uncertain data is modeled based on a filtered probability space $(\Omega, \mathscr{F}, \mathbb{P})$ with sample space Ω, σ-algebra $\mathscr{F}$ and probability measure $\mathbb{P}$. Further let $(\mathscr{F}_t)_{t\in[T]}$ with $\mathscr{F}_T := \mathscr{F}$ be a sequence of σ-algebras containing the events observable up to time t. Thus defining a filtration with $\mathscr{F}_1 \subseteq \mathscr{F}_2 \cdots \subseteq \mathscr{F}_T$, and let Ω_t be the sample space restricted to stage $t \in [T]$. We can then define a stochastic process $(\boldsymbol{\xi}_t)_{t\in[T]}$ with $\mathscr{F}_t$-measurable random vectors $\boldsymbol{\xi}_t : \Omega_t \to \mathbb{R}^{L_t}$ and corresponding realizations ξ_t. For the first stage, the data are assumed deterministic, *i.e.*, Ω_1 is a singleton.

Importantly, whereas in most of the literature on SDDP stagewise independent uncertainty is presumed, we allow the data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ to be stagewise dependent. This means that ξ_t may depend on lagged realizations of $(\boldsymbol{\xi}_t)_{t\in[T]}$, yielding an AR process. To formalize this, we make the following definitions.

- We denote by $\xi_{[t-1]} := (\xi_1,\dots,\xi_{t-1})^\top \in \mathbb{R}^{L_{[t-1]}}$, with $L_{[t-1]} := \sum_{k=1}^{t-1} L_k$, the history of the data process up to stage $t-1$.
- For all $t \in [T]$, let $b_t(\cdot,\cdot) : \mathbb{R}^{L_{[t-1]}} \times \mathbb{R} \to \mathbb{R}^{L_t}$ be a given vector-valued function, and let $\boldsymbol{\eta}_t$ be a stagewise independent random vector with realizations η_t. Then, we formalize the AR process by
$$\xi_t = b_t(\xi_{[t-1]}, \eta_t). \tag{2}$$
- For all $t \in [T]$, let $\widetilde{b}_t(\cdot,\cdot) : \mathbb{R}^{L_{[t-1]}} \times \mathbb{R} \to \mathbb{R}^{L_{[t]}}$ be a vector-valued function defined by
$$\xi_{[t]} = \big(\xi_{[t-1]}, b_t(\xi_{[t-1]}, \eta_t)\big)^\top =: \widetilde{b}_t(\xi_{[t-1]}, \eta_t). \tag{3}$$

Remark 2.1. *As defined, most generally $\xi_{[t-1]}$ is assumed to contain the whole history of the stochastic process over the horizon $[t-1]$. However, to reduce storage requirements we may also limit it to a minimal subset if the function $b_t(\cdot,\cdot)$ does not require all previous realizations. Furthermore, depending on the lag order of the data process, past realizations for stages $t < 1$ may be required in $b_t(\cdot,\cdot)$. In such a case, these realizations have to be stored in $\xi_{[t-1]}$ as well. For the ease of presentation, we abstain from this case unless stated otherwise. For a detailed discussion on this topic, see [13].*

Remark 2.2. *If function $b_t(\cdot,\cdot)$ is linear, it may be represented as a linear mapping using matrices, see for instance [16, 27]. In all other cases, it is most commonly defined componentwise using functions $b_{ti} : \mathbb{R}^{L_{[t-1]}} \times \mathbb{R} \to \mathbb{R}$ for all $i = 1, \dots, L_t$ [13].*

For simplicity, we assume that only the data vectors $h_t(\xi_t)$ occurring in the right-hand side (RHS) of the constraints are uncertain. However, all of the ideas presented in this paper do also hold if we allow for uncertainty in T_{t-1}, W_t and c_t, as long as this additional uncertainty is stagewise independent.

We further take the following assumptions with respect to the uncertainty in (MSLP).

Assumption 1. *The data process $(\boldsymbol{\xi}_t)_{t\in[T]}$ satisfies the following conditions:*

(a) For all $t \in [T]$, $h_t(\xi_t) = \xi_t$.

(b) For any $t = 2, \dots, T$, given any history $\xi_{[t-1]}$ and any realization η_t, the realization ξ_t is defined by relation (2) and the state $\xi_{[t]}$ is defined by relation (3).

(c) For all $t \in [T]$, the random vectors $\boldsymbol{\eta}_t$ follow a known, discrete and finite distribution with realizations $\eta_t^{(j)}$ and associated probabilities p_{tj} for $j = 1, \dots, q_t$ and $q_t \in \mathbb{N}$.

Part (a) simplifies the dependency of the RHS vector h_t on ξ_t and is standard for most work on SDDP. Part (b) formalizes our assumption of stagewise dependent uncertainty. We explicitly allow non-convex functions $b_t(\cdot,\cdot)$. Part (c) is a standard assumption to ensure computational tractability of (MSLP). Note that by relation (2) and assumption (c), the uncertainty can be illustrated by a finite scenario tree.

Notationwise, as done in (c), we set upper indices in brackets to distinguish them from exponents.

The nested formulation (1) of (MSLP) can be decomposed into a set of coupled subproblems by means of its *dynamic programming equations* (DPE) [2]. More precisely, for $t = 2, \dots, T$, any history $\xi_{[t-1]}$ and any $j = 1, \dots, q_t$, the DPE are given by

$$Q_t(x_{t-1}, \xi_t^{(j)}) := \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \mathcal{Q}_{t+1}(x_t, \xi_{[t]}) \\ \text{s.t.} & T_{t-1}x_{t-1} + W_t x_t = h_t(\xi_t^{(j)}) \\ & x_t \geq 0, \end{cases} \tag{4}$$

with $\xi_t^{(j)} = b_t(\xi_{[t-1]}, \eta_t^{(j)})$,

$$\mathcal{Q}_{t+1}(x_t, \xi_{[t]}) := \mathbb{E}_{\boldsymbol{\xi}_{t+1}|\xi_{[t]}} \left[Q_{t+1}(x_t, \boldsymbol{\xi}_{t+1}) \right] \tag{5}$$

and $\mathcal{Q}_{T+1}(x_T, \xi_{[T-1]}) \equiv 0$. $Q_t(\cdot,\cdot)$ is called *value function* and $\mathcal{Q}_t(\cdot,\cdot)$ is called *expected value function* or cost-to-go function. For the deterministic first stage, we obtain

$$v^* = \begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathcal{Q}_2(x_1, \xi_{[1]}) \\ \text{s.t.} & W_1 x_1 = h_1 \\ & x_1 \geq 0. \end{cases}$$

Importantly, by relation (2), the value functions $Q_t(\cdot,\cdot)$ do implicitly depend on the history $\xi_{[t-1]}$, and by using conditional expectations, also the expected value functions $\mathcal{Q}_t(\cdot,\cdot)$ depend on $\xi_{[t-1]}$.

To ensure feasibility, we make two more assumptions for the remainder of this paper:

Assumption 2.

(a) The feasible set of MSLP is non-empty and bounded, which implies that v^* is finite.

(b) We have relatively complete recourse*, i.e., for any x_{t-1} and $\xi_{[t-1]}$, the stage-t subproblem* (4) *is feasible for all $\xi_t^{(j)}, j = 1, \dots, q_t$.*

A challenge in exploiting the DPE to solve (MSLP) is that the expected value functions are not known in analytical form in advance. The key idea in SDDP is to iteratively approximate them from below using functions in all its arguments; see Sect. D in the EC for a detailed description. We refer to these functions as *cuts*. Combined, these cuts build outer approximations $\mathfrak{Q}_t(\cdot,\cdot)$ of $\mathcal{Q}_t(\cdot,\cdot)$ for all $t = 2, \dots, T$, which we refer to as *cut approximations*.

Replacing the true expected value functions with cut approximations in (4), we can define *approximate value functions*

$$\underline{Q}_t(x_{t-1}, \xi_t^{(j)}) := \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \mathfrak{Q}_{t+1}(x_t, \xi_{[t]}) \\ \text{s.t.} & T_{t-1}x_{t-1} + W_t x_t = h_t(\xi_t^{(j)}) \\ & x_t \geq 0. \end{cases} \tag{6}$$

For $\mathcal{Q}_{T+1}(\cdot,\cdot) \equiv 0$, we have $\mathfrak{Q}_{T+1}(\cdot,\cdot) \equiv 0$.

Cuts being also functions in $\xi_{[t-1]}$ is crucial because it allows cuts generated given one specific history of the data process to be *shared* with different nodes in the scenario tree. This is computationally important in SDDP, as otherwise cuts would have to be generated for each node in the scenario tree separately, with the number of nodes growing exponentially in T.

If $\mathcal{Q}_t(\cdot,\cdot)$ is convex, linear functions (also called *cutting-planes*, or *cuts* in a narrower sense) can be used for a tight and valid approximation from below. In the standard setting of SDDP with stagewise independent uncertainty ($\xi_t = b_t(\xi_{[t-1]}, \eta_t) \equiv \eta_t$, see Sect. D in the EC), this is satisfied. The expected value functions do not depend on $\xi_{[t-1]}$ in this special case, and can be shown to be piecewise linear and convex in their only argument x_{t-1}. In contrast, for general functions $b_t(\cdot,\cdot)$, things become more intricate, as $\mathcal{Q}_t(\cdot,\cdot)$ is in general not guaranteed to be convex in $\xi_{[t-1]}$. In this case, non-convex cuts may be required for tight and valid outer approximations. We analyze this in more detail in the following sections.

3 SDDP with General Nonlinear AR Processes

We now address SDDP for the case of general nonlinear stagewise-dependent uncertainty in the RHS. Again, we refer to Sect. D in the EC for a description of standard SDDP.

3.1 Existing Approaches for Stagewise Dependent Uncertainty

A natural approach to handle AR processes (2) in (MSLP) is to reformulate the problem in such a way that it exhibits stagewise independence as in standard SDDP. The key idea is to interpret $\xi_{[t-1]}$ as an additional state variable for the stage-t subproblem, thus considering the state $(x_{t-1}, \xi_{[t-1]})$. Then, the remaining uncertainty is only described by the stagewise independent random vector $\boldsymbol{\eta}_t$. This can be achieved by adding the process equation (3) as an explicit constraint to the subproblems and adding ξ_t as a decision variable, as for instance proposed by Shapiro et al. in [27].

$$\underline{\widetilde{Q}}_t\big((x_{t-1}, \xi_{[t-1]}), \eta_t^{(j)}\big) = \begin{cases} \min\limits_{x_t, \xi_t} & c_t^\top x_t + \mathfrak{Q}_{t+1}(x_t, \xi_{[t]}) \\ \text{s.t.} & T_{t-1}x_{t-1} + W_t x_t = h_t(\xi_t) \\ & \xi_{[t]} = \widetilde{b}_t(\xi_{[t-1]}, \eta_t^{(j)}) \\ & x_t \geq 0. \end{cases} \tag{7}$$

We use the symbol $\underline{\widetilde{Q}}_t(\cdot,\cdot)$ here to differentiate the value function from the original one where the dependence on $\xi_{[t-1]}$ is implicit, even though their values are the same for a given scenario.

While this reformulation is helpful in restoring stagewise independence, both the state space dimension and the decision space dimension are increased. Additionally, problem (7) only remains linear if $\widetilde{b}_t(\cdot,\cdot)$ is a linear function.

An alternative – that at least avoids the decision space expansion – is to replace each occurrence of $h_t(\xi_t^{(j)}) = \xi_t^{(j)}$ and of $\xi_{[t]}$ with equations (2) and (3), respectively [13].

$$\underline{\widetilde{Q}}_t\big((x_{t-1}, \xi_{[t-1]}), \eta_t^{(j)}\big) = \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \mathfrak{Q}_{t+1}\big(x_t, \widetilde{b}_t(\xi_{[t-1]}, \eta_t^{(j)})\big) \\ \text{s.t.} & T_{t-1}x_{t-1} + W_t x_t = b_t(\xi_{[t-1]}, \eta_t^{(j)}) \\ & x_t \geq 0. \end{cases} \tag{8}$$

The new state $\xi_{[t]}^{(j)}$ that enters stage $t+1$ as a parameter can then be computed using equation (3) outside of the subproblem.

The key idea of both approaches (often called *expanding the state space*) is that, by restoring stagewise independence and considering $\xi_{[t-1]}$ as an additional state variable, cuts for $\mathcal{Q}_t(\cdot,\cdot)$ can be naturally derived as functions in both x_{t-1} and $\xi_{[t-1]}$. Thus, they can be naturally shared between scenarios. However, as already mentioned in Sect. 2, *linear* cuts are only guaranteed to be valid under-approximators if $\mathcal{Q}_t(\cdot,\cdot)$ is convex in x_{t-1} *and* $\xi_{[t-1]}$. This, in turn, is only guaranteed if $b_t(\cdot,\cdot)$ only enters the RHS of problem (MSLP) and if $b_t(\cdot,\cdot)$ is linear. If we allow it to enter the RHS of $\geq$- constraints instead of equality constraints, convexity can also be established for convex functions $b_t(\cdot,\cdot)$ [13]. In contrast, if stagewise dependent uncertainty appears in W_t or c_t, or if $b_t(\cdot,\cdot)$ is a general non-convex function, then convexity of $\mathcal{Q}_t(\cdot,\cdot)$ is not ensured and linear cuts may lead to invalid approximations [23].

For this reason, extensions of SDDP to AR processes have been mostly limited to different types of linear processes so far (differing in lag order, periodicity, dependencies between different components of $\boldsymbol{\xi}_t$ etc.) [13, 16, 17, 22, 28]. Guigues [13] shows that by using subgradients, under certain assumptions (*i.e.*, ξ_t entering the RHS of $\geq$-constraints and $b_t(\cdot,\cdot)$ being monotonically increasing) also valid linear cuts can be derived for $\mathcal{Q}_t(\cdot,\cdot)$ if $b_t(\cdot,\cdot)$ is convex. Note that his results also allow for the RHS vector $h_t(\cdot)$ being a monotonically increasing, convex function of ξ_t instead of a constant (compare Assumption 1 (a)). Infanger and Morton [16] derive cut formulas for possibly non-convex functions $b_t(\cdot,\cdot)$, however relying on additivity in η_t and linearity in ξ_{t-1} in order to obtain computationally tractable formulas, meaning that the process may only be nonlinear in deterministic or stagewise independent problem components, such as $\eta_{t-1}, c_{t-1}, T_{t-2}$ or W_{t-1}.

In this paper, we extend these results to more general types of non-convex processes. In particular, we consider log-linear processes, where $b_t(\cdot,\cdot)$ satisfies additivity with respect to η_t, but is non-convex in $\xi_{[t-1]}$.

Finally, we should highlight that it is not necessarily required to *explicitly* include $\xi_{[t-1]}$ and η_t in the constraints, as it is done in (7)-(8). Instead, as Infanger and Morton [16] showed first (see also [13, 23]), it is also possible to compute $\xi_t = b_t(\xi_{[t-1]}, \eta_t)$ and $\xi_{[t]} = \widetilde{b}_t(\xi_{[t-1]}, \eta_t)$ outside of the stage-t subproblem and to only use these vectors in the SDDP subproblems then, similar to standard SDDP. In that case, the dependence on $\xi_{[t-1]}$ is not immediately visible in the subproblems, and only taken into account in the cut generation process. Again, cuts for $\mathcal{Q}_t(\cdot, \cdot)$ can be derived as functions in x_{t-1} and $\xi_{[t-1]}$. To include them in the stage-$(t-1)$ subproblem, ξ_{t-1} and $\xi_{[t-1]}$ are computed outside of the subproblem and then inserted into the RHS and the cut formulas. For stage t, this yields subproblems of the form

$$\underline{Q}_t(x_{t-1}, \xi_{[t]}^{(j)}) = \begin{cases} \min\limits_{x_t} & c_t^\top x_t + \mathfrak{Q}_{t+1}(x_t, \xi_{[t]}^{(j)}) \\ \text{s.t.} & T_{t-1}x_{t-1} + W_t x_t = h_t(\xi_t^{(j)}) \\ & x_t \geq 0. \end{cases} \tag{9}$$

Even more, it is also possible to adapt the cuts to the history $\xi_{[t]}$ outside of the subproblems, such that only ξ_t explicitly enters these problems. More precisely, the cut intercepts can be interpreted as being composed of a scenario-independent and a scenario-dependent term, with the latter being adapted to the scenario at hand before formulating the subproblem [16]:

$$\alpha_t(\xi_{[t-1]}) = \alpha_t^{\text{ind}} + \alpha_t^{\text{dep}}(\xi_{[t-1]}). \tag{10}$$

Explicit and recursive formulas for α_t^{ind} and $\alpha_t^{\text{dep}}(\xi_{[t-1]})$ are provided in [16] for different types of AR processes.

Importantly, even if cuts are computed, stored and evaluated in a different way with this *scenario-adaptable cut formula* approach, the obtained cuts are equivalent to those obtained by classical expanding the state space [22].

3.2 Cuts for General Nonlinear AR Processes

As pointed out before, for nonlinear AR processes $b_t(\cdot, \cdot)$, several challenges occur. First, if incorporated in the subproblems (7)-(8), the linearity of these problems is destroyed. Second, and more crucial, even if this is avoided (see (9)), the expected value functions $\mathcal{Q}_t(\cdot, \cdot)$ are in general non-convex in the history $\xi_{[t-1]}$. Therefore, they cannot be approximated by tight and valid *linear* cuts.

In this section, we show how *nonlinear* cuts for $\mathcal{Q}_t(\cdot, \cdot)$ can be derived instead, given general nonlinear processes $b_t(\cdot, \cdot)$. Afterwards, we highlight the computational challenges that come with these cuts and why incorporating them in SDDP is intractable in general. As a byproduct, we are able to identify why for the special case of log-linear processes tractable closed-form expressions can be derived for the cuts.

We take a general perspective on the cut generation process in SDDP given some nonlinear functions $b_t(\xi_{[t-1]}, \eta_t)$ and $\widetilde{b}_t(\xi_{[t-1]}, \eta_t)$ as defined in (2) and (3). For this case, we can derive a general cut formula for some arbitrary stage t.

Theorem 3.1. *For any $t = 2, \ldots, T$ and any state $(x_{t-1}, \xi_{[t-1]})$, a valid cut for the* *expected value function* $\mathcal{Q}_t(\cdot,\cdot)$ *is given by*

$$\mathcal{Q}_t(x_{t-1},\xi_{[t-1]}) \geq \underbrace{\sum_{j=1}^{q_t} p_{t-1,j}\beta_{tj}^\top}_{=:\beta_t^\top} x_{t-1} + \sum_{\tau=t}^{T} \underbrace{\sum_{j=1}^{q_t} p_{tj}\alpha_{tj}^{(\tau)}(\xi_{[t-1]})}_{=:\alpha_t^{(\tau)}(\xi_{[t-1]})} \tag{11}$$

with

$$\beta_{tj} := -(\pi_{tj}^*)^\top T_{t-1} \tag{12}$$

and

$$\begin{aligned} \alpha_{tj}^{(t)}(\xi_{[t-1]}) &:= (\pi_{tj}^*)^\top b_t(\xi_{[t-1]},\eta_t^{(j)}) \\ \alpha_{tj}^{(\tau)}(\xi_{[t-1]}) &:= \sum_{r\in R_{t+1}} \rho_{rtj}^*\alpha_{r,t+1}^{(\tau)}\big(\widetilde{b}_t(\xi_{[t-1]},\eta_t^{(j)})\big), \quad \tau = t+1,\dots,T. \end{aligned} \tag{13}$$

Here, π_{tj}^* *and* ρ_{rtj}^* *denote dual optimal solutions for the original constraints and existing cut constraints* $r \in R_{t+1}$.

This cut is tight for $\underline{\mathcal{Q}}_t(\cdot,\cdot) := \mathbb{E}_{\boldsymbol{\eta}_t}\big[\underline{\widetilde{Q}}_t(\cdot,\boldsymbol{\eta}_t)\big]$ *at the state* $(\bar{x}_{t-1},\bar{\xi}_{[t-1]})$ *of construction.*

We provide a proof in Appendix A.1.

As the intercept terms depend on $\widetilde{b}_t(\cdot,\cdot)$, the cut is nonlinear, and in general nonconvex. Cut formula (11) reveals some of the computational challenges of evaluating these cuts for a given scenario in SDDP. Assume that we cannot express the cuts (11) through a closed-form expression. Then, the cut intercept consists of $T-t+1$ summands and each of them has to be evaluated recursively. More precisely, each intercept term $\alpha_{tj}^{(\tau)}$ requires the intercepts for all cuts at stage $t+1$ evaluated for all possible realizations $\eta_{t+1}^{(i)}, i = 1,\dots,q_{t+1}$, each of which in turn require the intercepts for all cuts at stage $t+2$ evaluated for all possible realizations $\eta_{t+2}^{(k)}, k = 1,\dots,q_{t+2}$, and so on. This means that in order to evaluate cut formula (11), we have to traverse the whole subtree starting at the node associated with $\xi_{[t-1]}$. Even more, this has to be done every time a subproblem is solved for a specific scenario and for each cut in order to adapt it to that scenario. As the size of the scenario tree grows exponentially in T, this is computationally infeasible in general. Moreover, it counteracts the concept of SDDP to avoid traversing the whole tree in each iteration.

However, even if we are in principle able to derive closed-form expressions for the cut intercepts, generating and evaluating these nonlinear cuts remains computationally challenging. The reason is that for a general nonlinear function $b_t(\cdot,\cdot)$, the formulas become more and more complicated the more we step backwards through the stages. Let us make that more precise. The intercept $\alpha_T^{(T)}$ in formulas (11)-(13) contains function $b_T(\xi_{[T-1]},\eta_T)$. The intercept $\alpha_{T-1}^{(T)}$ in formula (A.1) essentially amounts to a function in the order of $b_T(\widetilde{b}_{T-1}(\xi_{[T-2]},\eta_{T-1}),\eta_T)$, and analogously the intercept $\alpha_t^{(T)}$ in (11) essentially amounts to a function in the order of

$$b_T\bigg(\widetilde{b}_{T-1}\Big(\cdots\widetilde{b}_{t+1}\big(\widetilde{b}_t(\xi_{[t-1]},\eta_t),\eta_{t+1}\big)\cdots,\eta_{T-1}\Big),\eta_T\bigg). \tag{14}$$

Most nonlinear functions $b_t(\cdot,\cdot)$ do not possess explicit general closed-form expressions for this kind of composition. In particular, the structure of the composite function may change for each stage. For instance, for polynomials the number of monomials and the degree of the polynomial may not be the same for all t. This may severely complicate deriving closed-form cut formulas in this case.

In contrast, for the linear case, for which efficient cut formulas have been derived in the literature (see Sect. 3.1), this complication does not exist. We explore this by first showing that formulas (11)-(13) generalize the linear case.

Corollary 3.2. *Consider a linear order-p PAR process of form*

$$\xi_t = b_t(\xi_{[t-1]}, \eta_t) = \sum_{k=1}^{p} \Phi_t^{(k)} \xi_{t-k} + \eta_t = \sum_{k=t-p}^{t-1} \Phi_t^{(t-k)} \xi_k + \eta_t \tag{15}$$

with parameter matrices $\Phi_t^{(k)}$, $k = 1, \dots, p$, $p \in \mathbb{N}$.

Then, the general cut formula (11) *is valid, but the cut intercept summands from* (13) *are described more specifically by the linear relations*

$$\alpha_t^{(\tau)}(\xi_{[t-1]}) = \sum_{k=t-p}^{t-1} C_t^{(\tau,t-k)} \xi_k + \omega_t^{(\tau)}, \quad \tau = t, \dots, T, \tag{16}$$

with matrices

$$\begin{aligned} C_{tj}^{(t,t-k)} &:= (\pi_{tj}^*)^\top \Phi_t^{(t-k)} \\ C_{tj}^{(\tau,t-k)} &:= \begin{cases} \sum_{r \in R_{t+1}} \rho_{rtj}^* \big(C_{r,t+1}^{(\tau,t+1-k)} + C_{r,t+1}^{(\tau,1)} \Phi_t^{(t-k)} \big), & \text{if } k \geq t-p+1 \\ \sum_{r \in R_{t+1}} \rho_{rtj}^* C_{r,t+1}^{(\tau,1)} \Phi_t^{(t-k)}, & \text{if } k = t-p \end{cases}, \end{aligned} \tag{17}$$

for $\tau = t+1, \dots, T$, *vectors*

$$\begin{aligned} \omega_{tj}^{(t)} &:= (\pi_{tj}^*)^\top \eta_t^{(j)} \\ \omega_{tj}^{(\tau)} &:= \sum_{r \in R_{t+1}} \rho_{rtj}^* \big(C_{r,t+1}^{(\tau,1)} \eta_t^{(j)} + \omega_{r,t+1}^{(\tau)} \big), \quad \tau = t+1, \dots, T, \end{aligned} \tag{18}$$

and

$$C_t^{(\tau,t-k)} := \sum_{j=1}^{q_t} p_{tj} C_{tj}^{(\tau,t-k)}, \quad \omega_t^{(\tau)} := \sum_{j=1}^{q_t} p_{tj} \omega_{tj}^{(\tau)}, \quad \tau = t, \dots, T. \tag{19}$$

We provide a proof for this result in Appendix A.2. Note that the formulas in Corollary 3.2 can be further aggregated and simplified. For instance, $C_t^{(\tau,t-k)}$ and $\omega_t^{(\tau)}$ can be aggregated over all $\tau = t, \dots, T$, so there is no need to store several of these terms, see also the cut formulas in the literature cited in Sect. 3.1. Here, we present the cuts in a form that directly relates to the formulas (11)-(13) for the general case. This allows us to identify the similarities and the key differences between the linear and the nonlinear case.

First, we obtain explicit closed-form expressions for $\alpha_t^{(\tau)}(\cdot)$, which only require the constant quantities $C_{r,t+1}^{(\tau,t+1-k)}$ and $\omega_{r,t+1}^{(\tau)}$ from the existing stage-$(t+1)$ cuts instead of a recursion including $\xi_{[t-1]}$. This means that we do not have to recursively evaluate the cut intercepts on the whole subtree starting at $\xi_{[t-1]}$, but only have to compute the cut coefficients once.

Second, for all t, $b_t(\cdot,\cdot)$ and $\widetilde{b}_t(\cdot,\cdot)$ are linear functions in $\xi_{[t-1]}$. As compositions of linear functions are linear functions again, recursively the closed-form cut intercept expressions for all stages can be shown to be linear, instead of changing their shape between stages and becoming overly complicated. It is also this linearity that allows for the previously mentioned additional aggregation.

For the general nonlinear case, one idea to deal with the challenges discussed above is to *approximate* the intercepts in a reasonable way. In order to retain validity of the cuts, the intercepts have to be underestimated. Under certain assumptions, this can be achieved by limiting the number of intercept terms that are taken into account.

Assume that for all $t = 2, \dots, T$, we consider inequality constraints

$$T_{t-1}x_{t-1} + W_t x_t \geq h_t(\xi_t)$$

in the subproblems, such that both types of dual multipliers π_t and $\rho_{rt}, r \in R_{t+1}$, are guaranteed non-negative. Additionally, assume that $b_t(\cdot,\cdot) \geq 0$ (and thus, also $\widetilde{b}(\cdot,\cdot) \geq 0$) for all t.

In that case, by backward recursion, we can conclude that in (13) we have $\alpha_{tj}^{(\tau)} \geq 0$ for all $t = 2, \dots, T$, $\tau = t, \dots, T$ and $j = 1, \dots, q_t$. This directly implies that we may underestimate the cut intercept by omitting some of these terms. In this regard, note that by definition in (13), $\alpha_t^{(t)}$ involves no recursion, $\alpha_t^{(t+1)}$ involves a one-step recursion, $\alpha_t^{(t+2)}$ involves a two-step recursion etc. Consequently, by truncating the cut intercept to terms $\alpha_t^{(\tau)}$ only for $\tau = t, \dots, \bar{\tau}$ for some $\bar{\tau} \in \mathbb{N}$, $\bar{\tau} \in [t+1, T]$, the required recursion can be significantly limited. Moreover, as this also reduces the complexity of the involved nonlinear function compositions, nonlinear closed-form expressions for the cut intercepts are less difficult to derive.

We formalize this result in the following corollary.

Corollary 3.3. *Let the coupling constraints in subproblems* (4) *be $\geq$-inequalities for all $t = 2, \dots, T$ and let $b_t(\cdot,\cdot) \geq 0$ for all $t = 1, \dots, T$ (and potentially required earlier lags). Further, let $\bar{\tau} \in \mathbb{N}$ and $\bar{\tau} \in [t+1, T]$.*

Then, for any $t = 2, \dots, T$ and any state $(x_{t-1}, \xi_{[t-1]})$, a valid cut for the expected value function $\mathcal{Q}_t(\cdot,\cdot)$ is given by

$$\mathcal{Q}_t(x_{t-1}, \xi_{[t-1]}) \geq \beta_t^\top x_{t-1} + \sum_{\tau=t}^{\bar{\tau}} \alpha_t^{(\tau)}(\xi_{[t-1]}). \tag{20}$$

The cuts given by (20) are not tight in general, so there exists a trade-off between computational tractability of the nonlinear cuts in formula (11) on the one side and approximation quality and convergence guarantees for SDDP on the other side. We leave exploring this in detail for future research.

In the remainder of this paper, we take a different approach by considering a special type of nonlinear process $b_t(\cdot,\cdot)$ that allows the derivation of tractable explicit closed-form expressions for nonlinear cuts. Importantly, similar to the linear case, the intercept coefficients can be computed *once* recursively and, despite being nonlinear, the cut formulas have the same structure for all stages $t \in [T]$. The main reason is that for this type of process, the compositions (14) are also structure-preserving. As a motivational example, let $b_t(\xi_{t-1}) = e^{a_t}\xi_{t-1}^{c_t}$ for all $t \in [T]$ with some constants $a_t, c_t \in \mathbb{R}$. Then, apart from a change in exponents,

$$b_t\big(b_{t-1}(\xi_{t-2})\big) = e^{a_t}\big(e^{a_{t-1}}\xi_{t-2}^{c_{t-1}}\big)^{c_t} = e^{a_t + a_{t-1}c_t}\xi_{t-2}^{c_{t-1}c_t} \tag{21}$$

has exactly the same structure as $b_t(\cdot,\cdot)$.

4 SDDP with Log-linear AR Processes

4.1 Log-linear AR Processes

In the remainder of this paper, we consider a special type of nonlinear AR processes to describe the uncertainty in the RHS of (MSLP), which we call *log-linear* AR processes, as they explain the autoregressive relation between ξ_t and $\xi_{[t-1]}$ in a logarithmized form. In [18], such processes are also called *geometric autoregressive*. In the literature on SDDP, this type of process is often considered in order to ensure that the realizations ξ_t are guaranteed non-negative, which may not be the case for classical linear AR processes. For instance, this feature is important for water inflows into hydro reservoirs [27].

We consider different types of log-linear AR processes, which differ in generality. Later on, we derive cut formulas for all of them.

The General Case. We define the log-linear AR process componentwise (see Remark 2.2), given some finite lag order p. More precisely, for any $t \in [T]$, for any realization ξ_t and for some order $p \in \mathbb{N}, p < T$, the components $\xi_{t\ell}, \ell = 1,\dots,L_t$, are defined by

$$\log(\xi_{t\ell}) = \gamma_{t\ell} + \sum_{k=1}^{p} \sum_{m=1}^{L_{t-k}} \phi_{t\ell m}^{(k)} \log(\xi_{t-k,m}) + \psi_{t\ell}\eta_{t\ell}. \tag{22}$$

Here, we use $\log(\cdot)$ to denote the natural logarithm. The quantities $\gamma_t \in \mathbb{R}^{L_t}$, $\psi_t \in \mathbb{R}^{L_t}$ and $\phi_{t\ell}^{(k)} \in \mathbb{R}^{L_{t-k}}$ for $\ell = 1,\dots,L_t$ and $k = 1,\dots,p$ are parameter vectors that are estimated from the data, whereas η_t is a realization of the stagewise independent vector $\boldsymbol{\eta}_t$ as defined in Sect. 2.

Remark 4.1. *We make several remarks with respect to the above definition.*

(1) By definition, process (22) *is only well-defined for $\xi_t > 0$ for $t \in [T]$.*

(2) In standard definitions of AR processes, the parameters in γ, ψ and $\phi^{(k)}$ are often assumed to be equivalent for all $t \in [T]$. In contrast, we allow them to change with t. For example, this is required if periodic autoregressive (PAR) processes are used where separate models are estimated for different months, years or the like, but $\phi_t = \phi_{t+\kappa}$, $\gamma_t = \gamma_{t+\kappa}$ and $\psi_t = \psi_{t+\kappa}$ for some period $\kappa \in \mathbb{N}$ [17].

(3) Importantly, process (22) *allows for dependencies between process components,* i.e.*, one component $\xi_{t\ell}$ is not only explained by an error term and by lagged values of the same component (*i.e.*, $\xi_{t-p,\ell},\dots,\xi_{t-1,\ell}$), but also by lagged values of other components of the process. As a practical example, the inflows of different hydro reservoirs may have some spatial dependency, which can be modeled by a spatial periodic autoregressive (SPAR) process [17].*

(4) In general (S)PAR processes, not only the coefficients, but also the lag orders $p_{t\ell m}$ are allowed to differ between stages t and components ℓ and m [13, 17]. However, we abstain from this, thus assuming a constant lag order $p \in \mathbb{N}, p < T$. As shown in Sect. 4.3, this proves crucial in the generation of neat cut formulas. If less lagged realizations are required for some index combination t, ℓ, m in practice, then we

may simply set the corresponding parameters $\phi_{t\ell m}$ to 0, so that the corresponding realization has no impact on the value of ξ_t.

(5) *In the literature, see for instance [17], (SP)AR processes are often introduced in a detrended form, which in our case translates to*

$$\frac{\log(\xi_{t\ell}) - \mu_{t\ell}}{\sigma_{t\ell}} = \sum_{k=1}^{p} \sum_{m=1}^{L_{t-k}} \tilde{\phi}_{t\ell m}^{(k)} \frac{\log(\xi_{t-k,m}) - \mu_{t-k,m}}{\sigma_{t-k,m}} + \eta_{t\ell}, \tag{23}$$

with $\mu_{t\ell}$ the expected value and $\sigma_{t\ell}$ the standard deviation of the process at stage t and for component ℓ. This does not diminish the generality of formulation (22), as (23) can always be converted to (22) by setting $\phi_{t\ell m}^{(k)} = \tilde{\phi}_{t\ell m}^{(k)} \frac{\sigma_{t\ell}}{\sigma_{t-k,m}}$, $\psi_{t\ell} = \sigma_{t\ell}$ and $\gamma_{t\ell} = \mu_{t\ell} - \sum_{k=1}^{p} \sum_{m=1}^{L_{t-k}} \phi_{t\ell m}^{(k)} \mu_{t-k,m}$.

For each $\ell = 1, \dots, L_t$ we can use exponentiation on both sides of (22) to obtain

$$\begin{aligned} \xi_{t\ell} &= e^{\gamma_{t\ell}} e^{\psi_{t\ell}\eta_{t\ell}} \prod_{k=1}^{p} \prod_{m=1}^{L_{t-k}} \xi_{t-k,m}^{\phi_{t\ell m}^{(k)}} \\ &= e^{\gamma_{t\ell}} e^{\psi_{t\ell}\eta_{t\ell}} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\ell m}^{(t-k)}} \\ &=: b_{t\ell}^{(p)}(\xi_{[t-p:t-1]}, \eta_t). \end{aligned} \tag{24}$$

Note that this expression resembles (21). Furthermore, this definition corresponds to the definition of function $b_t(\cdot,\cdot)$ in (2), with an additional index signifying the lag order p and with $\xi_{[t-p:t-1]} := (\xi_{t-p}, \dots, \xi_{t-1})$ summarizing only the required history of $\boldsymbol{\xi}_t$ for this lag order.

We consider two special cases of log-linear AR processes.

Special Case: Componentwise Independent Lag-p Process. In many applications, it may be sufficient to assume componentwise independent processes, that is, only lags of the same component are used to explain $\xi_{t\ell}$. In that case, it is reasonable to assume $L_t \equiv L$ for all $t \in [T]$. The process equation (22) then simplifies to

$$\log(\xi_{t\ell}) = \gamma_{t\ell} + \sum_{k=1}^{p} \phi_{t\ell}^{(k)} \log(\xi_{t-k,\ell}) + \psi_{t\ell}\eta_{t\ell}$$

for each $\ell = 1, \dots, L$, and (24) simplifies to

$$\xi_{t\ell} = e^{\gamma_{t\ell}} e^{\psi_{t\ell}\eta_{t\ell}} \prod_{k=t-p}^{t-1} \xi_{k\ell}^{\phi_{t\ell}^{(t-k)}} = b_{t\ell}^{(p)}(\xi_{[t-p:t-1]}, \eta_t).$$

Summarizing this for all components, we can use vector notation and express this more compactly as

$$\begin{aligned} \xi_t &= e^{\gamma_t} \odot e^{\psi_t \eta_t} \odot \xi_{t-1}^{\phi_t^{(1)}} \odot \cdots \odot \xi_{t-p}^{\phi_t^{(p)}} \\ &= e^{\gamma_t} \odot e^{\psi_t \eta_t} \odot \bigodot_{k=1}^{p} \xi_{t-k}^{\phi_t^{(k)}} \\ &= b_t^{(p)}(\xi_{t-p:t-1}, \eta_t). \end{aligned} \tag{25}$$

Here, all vector exponents are applied componentwise, either to the same value if the base is a scalar or to the corresponding component if the base is a vector. The operator $\odot$ denotes the Hadamard product (componentwise product) of vectors, and the operator $\bigodot$ denotes an indexed sequence of Hadamard products.

Special Case: Componentwise Independent Lag-1 Process. For $p = 1$, process (25) further simplifies to

$$\xi_t = e^{\gamma_t} \odot e^{\psi_t \eta_t} \odot \xi_{t-1}^{\phi_t} = b_t(\xi_{t-1}, \eta_t). \tag{26}$$

In this case, we may omit the upper index k for ϕ_t.

4.2 Existing Approaches

We now turn to how SDDP can be applied in the case of uncertainty in the RHS of (MSLP) that is modeled by processes (22) (or its special cases (25)-(26)). As discussed in Sect. 3.1, the functions $\mathcal{Q}_t(\cdot,\cdot)$ become non-convex, and nonlinear approximations are required to ensure convergence of SDDP.

To avoid this difficulty, in the literature on SDDP it is proposed to either draw on Markov chains instead of general AR processes to model nonlinear uncertainty [18] or to linearize the process (22) (or its special cases (25)-(26)) [27]. More precisely, Shapiro et al. [27] propose to approximate the function $b_t(\cdot,\cdot)$ in (25) by a linear function $\widehat{b}_t(\cdot)$ using a first-order Taylor approximation. This linear process has the special feature that the stagewise independent error terms η_t enter its formula in a multiplicative form. Still, this simplifies the model in such a way that only convex (but modified) expected value functions $\widehat{\mathcal{Q}}_t(\cdot,\cdot)$ are considered in SDDP. Therefore, expanding the state space and cut generation can be applied in the same way as for the standard linear case. As the random process is linearized, (MSLP) is only solved approximately, though.

In addition to this linearization approach, we should note that *under special conditions* the functions $b_t(\cdot,\cdot)$ defining the processes (22), (25) or (26) are *convex* in $\xi_{[t-p:t-1]}$. In such a case, despite the nonlinearity of the process, the expected value functions $\mathcal{Q}_t(\cdot,\cdot)$ remain convex in all states. This means that linear cuts can be constructed using the cut generation approach by Guigues [13] that we already mentioned in Sect. 3.2. As these cuts are not only valid, but also *tight* at trial points x_{t-1}^i, almost sure finite convergence of SDDP is ensured [13].

The crucial part is that some strict assumptions have to be satisfied for this case to apply. First, it is required that ξ_t only enters the RHS of $\geq$-constraints instead of equality constraints in (MSLP), which may not yield an equivalent problem. Second, convexity of $b_t(\cdot,\cdot)$ may impose strict restrictions on the possible choice of coefficients $\phi_{t\ell m}^k$. Third, the approach by Guigues [13] additionally requires that $b_t(\cdot,\cdot)$ is monotonically increasing, further restricting the possible values of $\phi_{t\ell m}^k$. These requirements may not be satisfied in all practical applications.

We propose a novel and differing SDDP approach, exploiting the special structure of processes (22). In particular, we do neither linearize the functions $b_t(\cdot,\cdot)$ nor require convexity, monotonicity or a specific type of constraints. This directly implies that we have to deal with non-convex value functions $\mathcal{Q}_t(\cdot,\cdot)$. In our SDDP approach, we therefore obtain cuts that are nonlinear (and in general non-convex) in $\xi_{[t-p:t-1]}$. In contrast to the general case presented in Sect. 3.2, however, we are able to derive tractable closed-form expressions for these cuts, for which the coefficients can be computed recursively.

Table 1: Comparison with approaches in the literature.

	Shapiro et al. [27]	Guigues [13]	Our work
Process $b_t(\cdot,\cdot)$			
- Type	linearized	convex*	non-convex
- Monotonicity	no	yes*	no
- Componentwise independent	no	yes	no (22) / yes (25),(26)
Constraint type	arbitrary	$\geq$	arbitrary
RHS $h_t(\cdot)$	constant	monotonic	constant
Expanding the state	explicitly	cut generation	cut generation
Value functions	convexified	convex	non-convex
Cuts			
- Type in x_{t-1}	linear	linear	linear
- Type in $\xi_{[t-1]}$	linear	linear	non-convex
- Tightness	yes	yes	yes
Tackled problem	approximation	original*	original

* restricts possible values of $\phi_{t\ell m}^{(k)}$

Finally, we postulate that even if we restrict to the convex setting discussed above, our proposed nonlinear (but then convex) cuts provide better approximations of the convex functions $\mathcal{Q}_t(\cdot,\cdot)$ than the linear ones proposed in [13], as they better capture the nonlinear nature of these functions.

We summarize and compare the properties of all three approaches in Table 1.

4.3 Nonlinear Cut Formulas

We derive and present closed-form expressions of nonlinear cuts for the expected value functions $\mathcal{Q}_t(\cdot,\cdot), t = 2,\dots,T$, if the log-linear autoregressive processes defined in the previous section are used to model the uncertainty in the RHS of (MSLP).

4.3.1 The General Case

We first consider the general case, that is, process (22) or (24), respectively.

For any $t = 2,\dots,T-1$, let the set R_{t+1} index cuts that have been derived for stage $t+1$ already, and let $\alpha_{r,t+1}$ denote the associated cut intercepts. Moreover, for any $t = 2,\dots,T$, some state $(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]})$ and some realization $\eta_t^{(j)}, j = 1,\dots,q_t$, let $(\pi_{tj}^*, \rho_{rtj}^*)$ denote the dual optimal solutions to the corresponding dual subproblems. We can then define cut coefficients

$$\begin{aligned}
\beta_{tj}^\top &:= -\big(\pi_{tj}^*\big)^\top T_{t-1}, \\
\alpha_{t\ell j}^{(t)} &:= \pi_{t\ell j}^* e^{\gamma_{t\ell}} e^{\psi_{t\ell}\eta_{t\ell}^{(j)}}, \\
\alpha_{t\ell j}^{(\tau)} &:= \Big(\sum_{r\in R_{t+1}} \rho_{rtj}^* \big(\alpha_{r,t+1,\ell}^{(\tau)}\big)\Big) \prod_{\nu=1}^{L_t} e^{\gamma_{t\nu}\Theta(t+1,\tau,\ell,\nu,t)} e^{\psi_{t\nu}\eta_{t\nu}^{(j)}\Theta(t+1,\tau,\ell,\nu,t)}, \\
&\quad \tau = t+1,\dots,T
\end{aligned} \tag{27}$$

for $t = 2, \dots, T, \ell = 1, \dots, L_\tau$ and $j = 1, \dots, q_t$, and

$$\begin{aligned} \beta_t &:= \sum_{j=1}^{q_t} p_{tj}\beta_{tj} \\ \alpha_{t\ell}^{(\tau)} &:= \sum_{j=1}^{q_t} p_{tj}\alpha_{t\ell j}^{(\tau)}, \quad \tau = t, \dots, T, \end{aligned} \tag{28}$$

together with recursively defined quantities

$$\begin{aligned} \Theta(t,t,\ell,m,k) &:= \phi_{t\ell m}^{(t-k)} \\ \Theta(t,\tau,\ell,m,k) &:= \begin{cases} \sum_{\nu=1}^{L_t} \big(\phi_{t\nu m}^{(t-k)}\Theta(t+1,\tau,\ell,\nu,t)\big) \\ \quad +\Theta(t+1,\tau,\ell,m,k), & \text{if } k \geq t-p+1 \\ \sum_{\nu=1}^{L_t} \big(\phi_{t\nu m}^{(p)}\Theta(t+1,\tau,\ell,\nu,t)\big), & \text{if } k = t-p \end{cases} \end{aligned} \tag{29}$$

for $t = 2, \dots, T$, $\tau = t+1, \dots, T$, $k = t-p, \dots, t-1$, $\ell = 1, \dots, L_\tau$ and $m = 1, \dots, L_k$.

For the expected value function at stage t, we obtain the following cut result.

Theorem 4.2. *Suppose that the RHS uncertainty in problem (MSLP) is defined by process* (24) *with $p \in \mathbb{N}, p < T$. For any $t = 2, \dots, T$ and any state $(x_{t-1}, \xi_{[t-p:t-1]})$, a nonlinear cut for $\mathcal{Q}_t(\cdot,\cdot)$ is given by*

$$\mathcal{Q}_t(x_{t-1}, \xi_{[t-p:t-1]}) \geq \beta_t^\top x_{t-1} + \sum_{\tau=t}^{T}\sum_{\ell=1}^{L_\tau}\left(\alpha_{t\ell}^{(\tau)} \prod_{k=t-p}^{t-1}\prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,\tau,\ell,m,k)}\right), \tag{30}$$

with the coefficients defined as in (27)-(29).

We provide a proof in Appendix A.3.

The cut coefficients in (27) are independent of $\xi_{[t-p:t-1]}$, so they just have to be computed once after solving the corresponding subproblems (6) in SDDP. We discuss this in more detail in Sect. 4.4.

Remark 4.3. *Recall that for linear AR processes, we can dissect the cut into a cut gradient for x_{t-1}, which is multiplied with x_{t-1}, a cut gradient for ξ_{t-1}, which is multiplied with ξ_{t-1} (this may also be interpreted as a scenario-dependent cut intercept, see* (10)*), and a scenario-independent cut intercept [16]. Here, the same additive dissection is not possible. We have a cut gradient β_t for x_{t-1}, but the cut intercept is the sum of scalar products of scenario-independent vectors $\alpha_t^{(\tau)}$ and powers of (components of) lagged variables in $\xi_{[t-p:t-1]}$.*

The cut formula (30) in Theorem 4.2 shows the validity of the nonlinear cuts with respect to underestimating $\mathcal{Q}_t(\cdot,\cdot)$. Analogously to linear cuts in standard SDDP, see Sect. D in the EC, these nonlinear cuts possess some more beneficial properties. The most important one is tightness.

Corollary 4.4. *The cuts defined in Theorem 4.2 are tight for $\underline{\mathcal{Q}}_t(\cdot,\cdot)$ at $(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]})$.*

Proof. This result follows from equation (40). Reformulating the right-hand side and taking expectations in the same way as in the proof of Theorem 4.2 yields the RHS of cut formula (30) evaluated at $(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]})$. Taking expectations over the left-hand side yields $\underline{\mathcal{Q}}_t(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]}))$. □

Additionally, as the dual feasible sets of the dual subproblems are independent of x_{t-1} and $\xi_{[t-p:t-1]}$ for all stages $t \in [T]$ (see proof of Theorem 4.2), by restricting to optimal dual basic solutions, only finitely many different cuts can be generated. This is crucial for finite convergence of SDDP.

4.3.2 Special Case: Componentwise Independent Lag-p Process.

We now address the special case of process (25). In this case, the cut coefficients simplify to the vectorial form

$$\begin{aligned} \beta_{tj}^\top &:= -\big(\pi_{tj}^*\big)^\top T_{t-1}, \\ \alpha_{tj}^{(t)} &:= \pi_{tj}^* \odot e^{\gamma_t} \odot e^{\psi_t \eta_t^{(j)}}, \\ \alpha_{tj}^{(\tau)} &:= \Big(\sum_{r \in R_{t+1}} \rho_{rtj}^* \big(\alpha_{r,t+1}^{(\tau)}\big)\Big) \odot e^{\gamma_t \Theta(t+1,\tau,t)} \odot e^{\psi_t \eta_t^{(j)} \Theta(t+1,\tau,t)}, \\ &\quad \tau = t+1, \dots, T \end{aligned} \tag{31}$$

and

$$\begin{aligned} \beta_t &:= \sum_{j=1}^{q_t} p_{tj} \beta_{tj} \\ \alpha_t^{(\tau)} &:= \sum_{j=1}^{q_t} p_{tj} \alpha_{tj}^{(\tau)}, \quad \tau = t, \dots, T, \end{aligned} \tag{32}$$

with recursively defined quantities

$$\begin{aligned} \Theta(t,t,k) &:= \phi_t^{(t-k)} \\ \Theta(t,\tau,k) &:= \begin{cases} \phi_t^{(t-k)} \Theta(t+1,\tau,t) + \Theta(t+1,\tau,k), & \text{if } k \geq t-p+1 \\ \phi_t^{(p)} \Theta(t+1,\tau,t), & \text{if } k = t-p \end{cases} \end{aligned} \tag{33}$$

for $t = 2, \dots, T$, $\tau = t+1, \dots, T$ and $k = t-p, \dots, t-1$.

For the cuts, we obtain the following result.

Corollary 4.5. *Suppose that the RHS uncertainty in problem (MSLP) is defined by process* (25) *with* $p \in \mathbb{N}, p < T$. *For any* $t = 2, \dots, T$ *and any state* $(x_{t-1}, \xi_{[t-p:t-1]})$, *a valid nonlinear cut for* $\mathcal{Q}_t(\cdot,\cdot)$ *is given by*

$$\mathcal{Q}_t(x_{t-1}, \xi_{[t-p:t-1]}) \geq \beta_t^\top x_{t-1} + \sum_{\tau=t}^{T} \big(\alpha_t^{(\tau)}\big)^\top \bigodot_{k=t-p}^{t-1} \xi_k^{\Theta(t,\tau,k)}, \tag{34}$$

with the coefficients defined as in (31)-(33). *This cut is tight at* $\underline{\mathcal{Q}}_t(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]}))$.

4.3.3 Special Case: Componentwise Independent Lag-1 Process.

With a further simplification to $p = 1$, *i.e.*, for process (26), we obtain cut coefficients

$$\begin{aligned}
\beta_{tj}^\top &:= -\big(\pi_{tj}^*\big)^\top T_{t-1}, \\
\alpha_{tj}^{(t)} &:= \pi_{tj}^* \odot e^{\gamma_t} \odot e^{\psi_t \eta_t^{(j)}}, \\
\alpha_{tj}^{(\tau)} &:= \Big(\sum_{r \in R_{t+1}} \rho_{rtj}^* \big(\alpha_{r,t+1}^{(\tau)}\big)\Big) \odot e^{\gamma_t \prod_{k=t+1}^{\tau} \phi_k} \odot e^{\psi_t \eta_t^{(j)} \prod_{k=t+1}^{\tau} \phi_k}, \\
&\quad \tau = t+1, \dots, T
\end{aligned} \tag{35}$$

and

$$\begin{aligned}
\beta_t &:= \sum_{j=1}^{q_t} p_{tj} \beta_{tj} \\
\alpha_t^{(\tau)} &:= \sum_{j=1}^{q_t} p_{tj} \alpha_{tj}^{(\tau)}, \quad \tau = t, \dots, T.
\end{aligned} \tag{36}$$

For the cuts, we obtain the following result

Corollary 4.6. *Suppose that the RHS uncertainty in problem (MSLP) is defined by process* (26)*. For any $t = 2, \dots, T$ and any state (x_{t-1}, ξ_{t-1}), a valid nonlinear cut for $\mathcal{Q}_t(\cdot,\cdot)$ is given by*

$$\mathcal{Q}_t(x_{t-1}, \xi_{t-1}) \geq \beta_t^\top x_{t-1} + \big(\alpha_t^{(t)}\big)^\top \xi_{t-1}^{\phi_t} + \sum_{\tau=t+1}^{T} \big(\alpha_t^{(\tau)}\big)^\top \xi_{t-1}^{\prod_{k=t}^{\tau} \phi_k}, \tag{37}$$

with the coefficients defined as in (35)*-*(36)*. This cut is tight at $\underline{\mathcal{Q}}_t(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]}))$.*

4.3.4 Illustrative Examples

Example 4.7 (Non-convex expected value function)**.** *We consider an illustrative 3-stage example for SDDP with componentwise independent lag-*1 *processes* (26)*.*

The subproblems in the first iteration of SDDP are

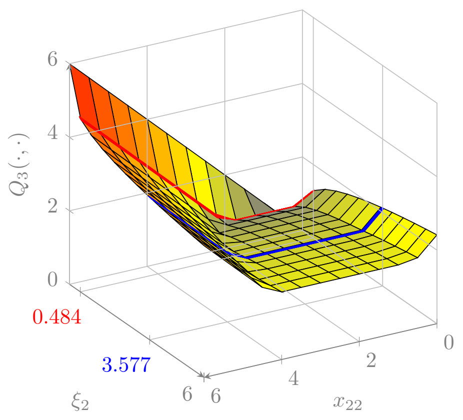
$$\begin{aligned}
v^* &= \min \Big\{ |x_1 - 4| + \theta_1 \ : \ x_1 \in [0,6], \theta_1 \geq 0 \Big\}, \\
Q_2(x_1, \xi_2) &= \min \Big\{ 2y_2 + x_2 + \theta_2 \ : \ y_2 - x_2 = \xi_2 - x_1, x_2, y_2 \in [0,6], \theta_2 \geq 0 \Big\}, \\
Q_3(x_2, \xi_3) &= \min \Big\{ x_{31} + x_{32} \ : \ x_{31} - x_{32} = \xi_3 - x_2, x_{31}, x_{32} \in [0,6] \Big\}
\end{aligned}$$

with the RHS uncertainty described by

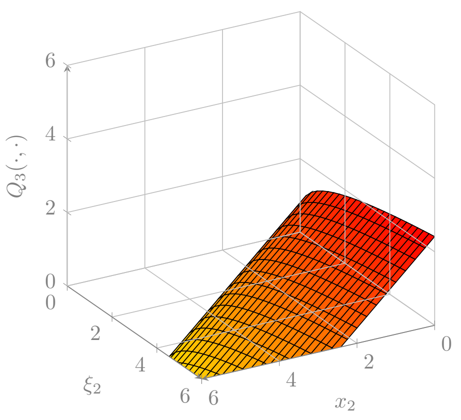
$$\xi_1 = 3, \quad \xi_t = e^{\eta_t} \xi_{t-1}^{\frac{1}{4}}, \quad \eta_t \in \{-1, 1\}, \quad \text{for } t = 2, 3. \tag{38}$$

*The stage-*1 *objective is nonlinear, but the absolute value can be reformulated using linear constraints.*

We focus our analysis of the non-convex cuts on $\mathcal{Q}_3(\cdot,\cdot)$, for which values of $\xi_2 \approx 0.484$ and $\xi_2 \approx 3.577$ may occur in the subproblems. This function is depicted in Figure 1a. It is convex in x_{22}, but not in ξ_2, thus non-convex.



(a) Expected value function $\mathcal{Q}_3(\cdot,\cdot)$.



(b) Non-convex cut for $\mathcal{Q}_3(\cdot,\cdot)$.

Figure 1: Expected value function $\mathcal{Q}_3(\cdot,\cdot)$ and non-convex cut for Example 4.7.

In iteration $i = 1$ of SDDP, we sample $(\eta_2^1, \eta_3^1) = (1, 1)$, and thus obtain $\xi_2^1 \approx 3.577$ and $\xi_3^1 \approx 3.738$. The forward pass yields the trial points $x_1^1 = 4$ and $x_2^1 = 0.4225$ at which cuts are constructed in the backward pass.

For stage 3, we obtain the non-convex cut

$$\theta_2 \geq -x_2 + \frac{1}{2}(e^{-1} + e^1)\xi_2^{\frac{1}{4}},$$

which is illustrated in Figures 1b and 2.

For stage 2, we may derive the non-convex cut

$$\theta_1 \geq \frac{1}{2}x_1 - \frac{1}{2}e^{-1}\xi_1^{\frac{1}{4}} + \frac{1}{4}(e^{-\frac{3}{4}} + e^{\frac{5}{4}})\xi_1^{\frac{1}{16}},$$

even though its dependence on ξ_1 is irrelevant due to ξ_1 being fixed to 3.

Example 4.8 (Convex expected value functions)**.** *We consider the illustrative problem from Example 4.7, but now with $\geq$-inequalities in all former equality constraints. Moreover, while ξ_1, ξ_2 and η_2 are defined as in* (38)*, the RHS uncertainty at stage 3 is now described by*

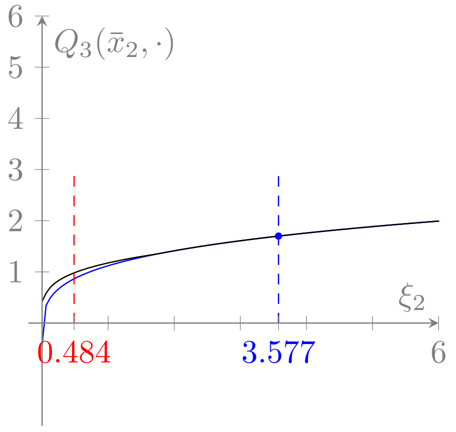
$$\xi_3 = e^{\eta_3}\xi_2^{\frac{3}{2}}, \quad \eta_3 \in \{-2, -1\}.$$

In this setting, $\mathcal{Q}_3(\cdot,\cdot)$ is convex in x_2 and ξ_2, as shown in Figure 3a.

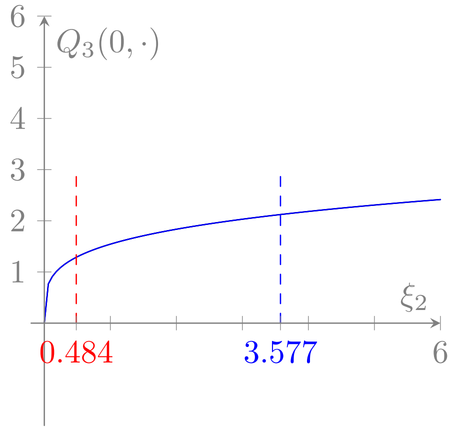
In iteration $i = 1$, at stage $t = 3$ and trial point $(x_2^1, \xi_2) \approx$ (0.4225, 3.577)*, we generate the cut*

$$\theta_2 \geq -x_2 + \frac{1}{2}(e^{-1} + e^{-2})\xi_2^{\frac{3}{2}},$$

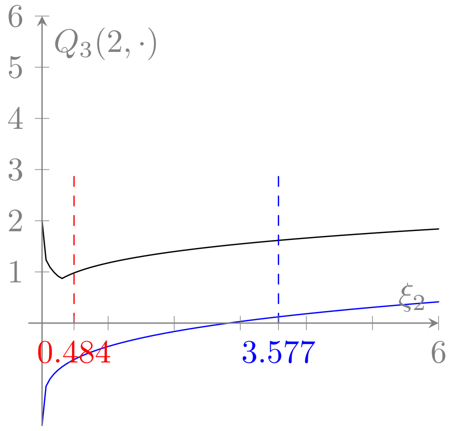
which is a nonlinear, but convex function. Additionally, we use the approach from [13] to generate a linear cut at the same state. Both cuts are illustrated in Figure 3b. We can see that both cuts are tight at the point of construction, but that the nonlinear cut provides a better approximation at different values of ξ_2.



(a) $\mathcal{Q}_3(\cdot,\cdot)$ (black) and non-convex cut (blue) projected on trial point $x_2^1 = 0.4225$.

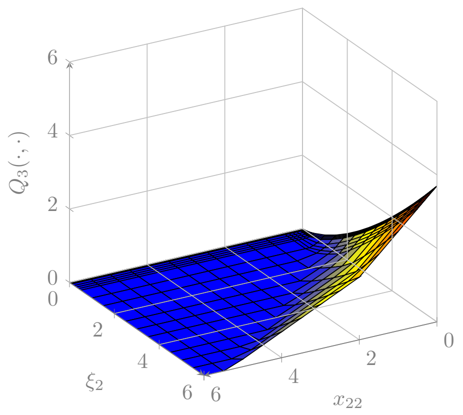


(b) $\mathcal{Q}_3(\cdot,\cdot)$ (black) and non-convex cut (blue) projected on $x_2 = 0$. Both projections coincide.

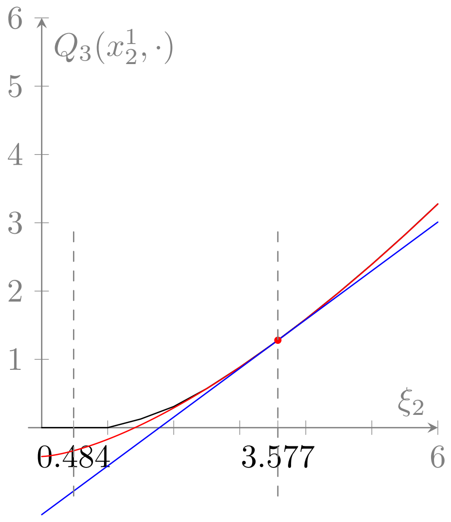


(c) $\mathcal{Q}_3(\cdot,\cdot)$ and non-convex cut projected on $x_2 = 2$.

Figure 2: Non-convex cut obtained for $\mathcal{Q}_3(\cdot,\cdot)$ in Example 4.7.



(a) Convex expected value function $\mathcal{Q}_3(\cdot,\cdot)$.



(b) Convex expected value function $\mathcal{Q}_3(\cdot,\cdot)$ (black), linear cut (blue) and nonlinear cut (red) projected on $x_2^1 = 0.4225$.

Figure 3: Expected value function and cuts for Example 4.8.

4.4 Integration into SDDP

The nonlinear (and in general non-convex) cuts derived in Sect. 4.3 can be incorporated into SDDP (see Alg. 1 in Sect. D of the EC) without substantial changes to the algorithm. Importantly, due to the validity, tightness and finiteness of the cuts, almost sure finite convergence is assured as for standard SDDP.

Approach I: Including Nonlinear Cuts in Subproblems. In the vein of formulation (8), for all stages $t = 1, \dots, T-1$, we can incorporate the cuts for $\mathcal{Q}_{t+1}(\cdot,\cdot)$ in the stage-t subproblem (6) in their nonlinear form (30). In both the RHS of the original constraints and in cut formula (30), we then have to replace each occurrence of ξ_t with function $b_t(\xi_{[t-p:t-1]}, \eta_t)$ in order to express it using the stage-t state variables. As functions in $\xi_{[t-p:t-1]}$, the cuts can then be evaluated for the scenario at hand.

Whereas this approach introduces nonlinear terms into the stage-t subproblem, these terms are only nonlinear in parameters η_t and states $\xi_{[t-p:t-1]}$ that are fixed for the subproblem, but remain linear in all decision variables, as long as we do not follow formulation approach (7). Therefore, the subproblems remain linear and we can use SDDP as usual. Apart from expanding the state space, the only difference in SDDP is that when cuts are generated (lines 15-16 of Alg. 1 in Sect. D of the EC), the formulas for linear cuts have to be replaced with the respective formulas from Sect. 4.3.

We acknowledge that many solvers and modeling frameworks (*e.g.*, the `SDDP.jl` and `JuMP.jl` packages in Julia that our experiments in Sect. 5 are based on) do not allow for nonlinear terms in linear programs, no matter if they contain parameters or decision variables. In such a case, a different approach can be applied to use the nonlinear cuts from Sect. 4.3.

Approach II: Adapting the Cut Intercepts. In the vein of formulation (9) and the cut-adaptation formulas by Infanger and Morton [16], alternatively we can

Table 2: Computation complexity and storage requirements of cut formulas.

Process	(24)	(25)	(26)	
Lag Order	$p > 1$	$p > 1$	$p = 1$	
Formulas	(27)-(30)	(31)-(34)	(35)-(37)	
Operation		Complexity		Frequency
COMPUTE(Θ)	$\mathcal{O}(T^2L^3p)$	$\mathcal{O}(T^2Lp)$	$\mathcal{O}(T^2L)$	once
COMPUTE(β)	$\mathcal{O}(Lnq)$	$\mathcal{O}(Lnq)$	$\mathcal{O}(Lnq)$	per cut
COMPUTE(α)	$\mathcal{O}(RTL^2q)$	$\mathcal{O}(RTLq)$	$\mathcal{O}(RTLq)$	per cut
EVAL(stage-t cut)	$\mathcal{O}(TL^2pn)$	$\mathcal{O}(TLpn)$	$\mathcal{O}(TLn)$	per cut & iteration
EVAL(stage-t cuts)	$\mathcal{O}(RTL^2pn)$	$\mathcal{O}(RTLpn)$	$\mathcal{O}(RTLn)$	per stage & iteration
EVAL(all cuts)	$\mathcal{O}(RT^2L^2pn)$	$\mathcal{O}(RT^2Lpn)$	$\mathcal{O}(RT^2Ln)$	per iteration
STORE(γ)	$\mathcal{O}(TL)$	$\mathcal{O}(TL)$	$\mathcal{O}(TL)$	
STORE(ψ)	$\mathcal{O}(TL)$	$\mathcal{O}(TL)$	$\mathcal{O}(TL)$	
STORE(ϕ)	$\mathcal{O}(TL^2p)$	$\mathcal{O}(TLp)$	$\mathcal{O}(TL)$	
STORE(Θ)	$\mathcal{O}(T^2L^2p)$	$\mathcal{O}(T^2Lp)$	$\mathcal{O}(T^2L)$	
STORE(one stage-t cut)	$\mathcal{O}(TL)$	$\mathcal{O}(TL)$	$\mathcal{O}(TL)$	
STORE(all stage-t cuts)	$\mathcal{O}(RTL)$	$\mathcal{O}(RTL)$	$\mathcal{O}(RTL)$	
STORE(all cuts)	$\mathcal{O}(RT^2L)$	$\mathcal{O}(RT^2L)$	$\mathcal{O}(RT^2L)$	

T : number of stages, L : dimension of ξ, n : dimension of x (state space dimension)
R : number of existing cuts, p : lag order, q : number of realizations of η

first compute ξ_t outside of the stage-t subproblem. For all cuts existing in the stage-t subproblem, we can then use the known values of $\xi_{[t+1-p:t]}$ and $\eta_t^{(j)}$ to evaluate the intercepts in cut formula (30) and to adapt them to the given scenario within the stage-t subproblem. In this case, in SDDP still linear cut formulas are used, but the intercepts are scenario-dependent. This means that each time, a subproblem is solved in SDDP (lines 6, 9, 13, 18 of Alg. 1 in Sect. D of the EC), first the intercepts of all existing cuts have to be adapted to the current scenario.

Computation and Storage Requirements. We now analyze the computational and storage requirements if our nonlinear cuts are used within SDDP. All the coefficients, exponents and parameters that have to be computed, stored or evaluated depend on several index sets. In order to provide a clear picture of the requirements, we summarize them in Table 2. We distinguish between the three cases of log-linear processes for which we derived cut formulas in the previous subsections. The storage requirements are based on the indices of the considered cuts and cut coefficients (note that some of them are hidden in vector notation in the cut formulas). The computational complexity is based on the formulas itself.

First, clearly the parameters ϕ and γ defining the data process (22) have to be known and stored (see STORE(γ) and STORE(ϕ) in Table 2). Additionally, the cut formulas require quantities $\Theta(\cdot)$, as defined in (29). Noteworthy, whereas depending on the stage t, the intercept factor index τ, the lag k and the component ℓ, these quantities do neither depend on the current iteration nor on the considered scenario. Therefore, they can be computed recursively starting at stage T *once* before running SDDP (see COMPUTE(Θ) in Table 2) and then be stored centrally (see STORE(Θ) in Table 2).

When a nonlinear cut (30) is computed, the *scenario-independent* coefficients β_t and

$\alpha_t^{(t)}, \dots, \alpha_t^{(\tau)}$ have to be computed using formulas (27)-(28) or the simpler alternative formulas (see COMPUTE(β) and COMPUTE(α) in Table 2). Once computed, these coefficients have to be stored for the given cut (see STORE(one stage-t cut) in Table 2). They can then be used in order to add the cut formula to the subproblem of stage-($t-1$) explicitly (Approach I), to evaluate and adapt the cut for a given scenario outside of the subproblem (Approach II), and also to compute the coefficients at the previous stage; recall that each computation of some $\alpha_t^{(\tau)}$ requires the intercept factors $\alpha_{t+1}^{(\tau)}$ from the following stage.

If Approach II is used, additionally all cut intercepts have to be adapted manually to the current scenario before a particular subproblem can be solved (see EVAL in Table 2). As all required coefficients like $\alpha_t^{(\tau)}$ and $\Theta(\cdot)$ have already been computed and stored before, only the actual cut intercept formula has to be evaluated by taking the required values and inserting them into formula (30). Importantly, the product $\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,\tau,\ell,m,k)}$ in the intercept formula only has to be computed once per subproblem, as it is cut-independent.

Given that Approach II is used and that one cut is generated per stage, we can estimate the (worst-case) computational complexity per iteration by

$$\text{COMPUTE}(\beta) \cdot T + \text{COMPUTE}(\alpha) \cdot T + \text{EVAL(all cuts)} = \mathcal{O}(RT^2L^2pnq).$$

We conclude that the computation/evaluation and storage effort can be quite significant, especially for large values of T, L and R. In particular, it grows with the number of previously generated cuts $R = TI$ with I the number of finished iterations. However, in contrast to the general nonlinear case discussed in Sect. 3.2, the complexity does not grow exponentially in the number of stages, as it does not depend on the size of the scenario tree. It is polynomial.

As SDDP is known to have a worst-case iteration complexity (number of iterations until convergence) that is exponential in T, exponential in n (as well as p and L if expanding the state space) and polynomial in q_t, we suspect that the polynomial cut generation cost should be computationally bearable. Additionally, compared to using linear cuts, our cuts ensure almost sure convergence for SDDP applied to the true (MSLP) (see Table 1). Finally, when the value functions of (MSLP) are convex, our cuts lead to higher cost per iteration than linear cuts, but may also reduce the number of required iterations to achieve convergence.

5 Computational Tests

5.1 Case Study Description and Inflow Modeling

We test our version of SDDP on a long-term hydrothermal scheduling (LTHS) problem using a simplified version of the Brazilian power system with 4 reservoirs, 95 generators and a planning horizon of 60 months. The reservoirs are modeled as energy-equivalent reservoirs (EER) representing the regions Southeast (SE), South (S), Northeast (NE) and North(N) of the Brazilian power system, and thus aggregating information of several existing hydro reservoirs. In addition to the four EERs, the power system contains a fifth transshipment node (T) used to model transmission constraints.

This problem has been widely studied in the literature, see for instance [18, 26, 27]. The objective of LTHS is to compute an optimal operation policy for the power system, meaning that the load in each month and node is covered by the power supply from thermal generators and hydro power plants or power flow from another node at a cost minimum, while satisfying generator limits and transmission constraints. As we assume the hydro inflows to be stochastic, LTHS can be modeled as a multistage stochastic problem, where we aim for a policy with optimal *expected* cost. For the full problem formulation, we refer to Sect. D in the EC.

To model the stochastic inflows for the four reservoirs, we fit a loglinear PAR model of type (25) based on historic monthly inflow data from 1931 to 2009. We consider two different model variants.

- `LOG-BIC`. A loglinear PAR model where the lag orders for each component are selected using the Bayesian Information Criterion (BIC). This choice yields $p = 10$.
- `LOG-1`. A loglinear PAR model where the lag order is fixed to $p = 1$.

A comprehensive description of the model fitting and validation steps, as well as the obtained model specification, is provided in Sect. E of the EC.

For comparison, we also consider variants of a linearized AR model, based on the Taylor approximation approach described in Sect. 4.2.

- `LIN-FIT`. A linear AR model of order 1 fitted using the same procedure as the log-linear models.
- `LIN-SHA`. A linear AR model of order 1 with parameters as reported in [26].

Note that our fitted model `LIN-FIT` uses a univariate distribution for each component to model the stagewise independent random variable $\boldsymbol{\eta}_t$, whereas `LIN-SHA` uses a multivariate distribution [26].

Remarkably, the average inflow level obtained using `LOG-BIC` and `LOG-1` is consistently lower than for the linearized models. For instance, for the largest system SE it is 3-4% lower on average. A simulation analysis indicates that sample paths obtained from the log-linear models better match the statistical properties of the historical monthly inflow data, so they provide a more accurate representation of the inflows. Note that similar observations are made in [18]. For details, again we refer to Sect. E of the EC.

As we shall see, this difference in inflow levels also results in a clear difference of total costs, thus highlighting the importance of directly incorporating log-linear inflows models into SDDP instead of linearizing them. On the other hand, different inflow levels make it hard to get a clear comparison of both considered versions of SDDP in itself.

5.2 Implementation

Our version of SDDP, incorporating the nonlinear cuts from Sect. 4, is implemented in Julia-1.9.2 [3] based on the existing packages `SDDP.jl` [7] and `JuMP.jl` [8]. The code and our test data are available on GitHub, see `https://github.com/ChrisFuelOR/LogLinearSDDP.jl`.

All LP subproblems are solved using Gurobi 9.0.3 with an optimality tolerance of 10^{-4}. As Gurobi does not allow nonlinear functions in the constraints, even if they are linear in all decision variables, we apply "Approach II: Adapting the Cut Intercepts" from Sect. 4.4 in our implementation.

LTHS is solved for a planning horizon of 120 stages (60 original stages + 60 stages to remove the end-of-horizon effect) and for 100 different realizations of the stagewise independent random variable $\boldsymbol{\eta}_t$ per stage. SDDP is terminated after 1000 iterations. In each forward pass, one scenario path is randomly sampled. We execute five independent runs of SDDP for each model variant, differing in the seed chosen for the forward pass sampling, and present average results.

After termination of SDDP, an in-sample Monte Carlo simulation is conducted to compute a statistical upper bound for the current policy. Additionally, we perform several out-of-sample Monte Carlo simulations. In particular, we cross-validate each policy (obtained for one specific inflow model) using out-of-sample inflow data from each of the other inflow models. All simulations use 2000 replications.

For the linearized inflow models, the same configuration as above is used for running SDDP and the following simulations, however, the standard SDDP implementation in `SDDP.jl` is applied.

All experiments are executed on a Windows machine with 128 GB RAM and an AMD Ryzen 9 7900X 12-Core processor (4.7 GHz).

5.3 Computational Results

We now present the results of our experiments, starting with the output of SDDP.

5.3.1 SDDP Results

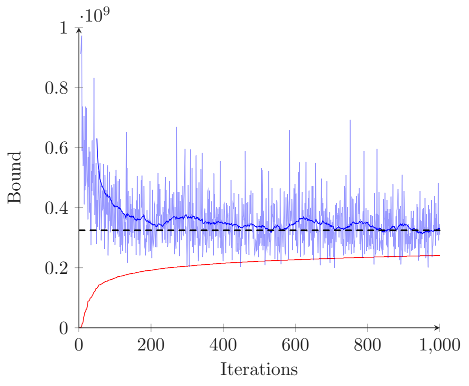
We first analyze the quality and development of the bounds obtained in SDDP. The lower bounds (LB) and the simulated objective values per iteration are depicted in Fig. 4 for all four model variants. More precisely, what is shown are the averages over the 5 independent runs of SDDP. For the fluctuating simulated objective values we additionally plot a moving-average with a window length of 50. We observe a similar convergence behavior of SDDP in all four cases. The bounds obtained for `LOG-BIC` and `LOG-1` are similar, with slightly higher values for `LOG-BIC`.

Both, the LBs and the simulated objective values, are significantly lower for `LIN-FIT` and `LIN-SHA`. As mentioned before, the main reason for this difference is that the inflows obtained using the linearized inflow models tend to exceed those obtained using the log-linear models by about 3-4%, which leads to about 30% lower total costs.

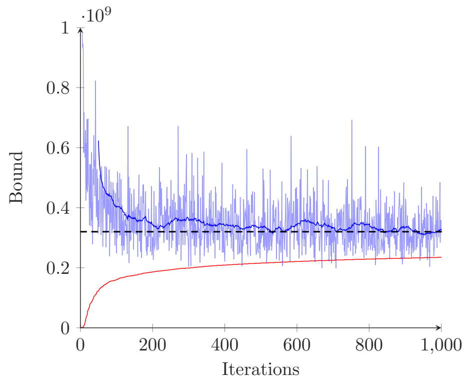
Despite different levels of inflows, we may still compare both versions of SDDP by consulting the statistical UBs computed in the in-sample simulation. These bounds are also depicted in Fig. 4 in dashed black lines. On average, the gap between the UBs and LBs is smaller for the log-linear models (27-28% compared to 29-32%).

On the other hand, comparing the computational cost, we observe that our version of SDDP exhibits some significant overhead. As Fig. 5 shows, the iteration time is considerably higher than for standard SDDP using a linearized AR model. Moreover, it also shows a linear increase over the iterations, as the number of cuts increases. This is in line with our theoretical analysis in Sect. 4.4. As expected, the time per iteration is higher for `LOG-BIC` compared to `LOG-1`. However, the difference is not too substantial given that p is 10 times larger for `LOG-BIC`.

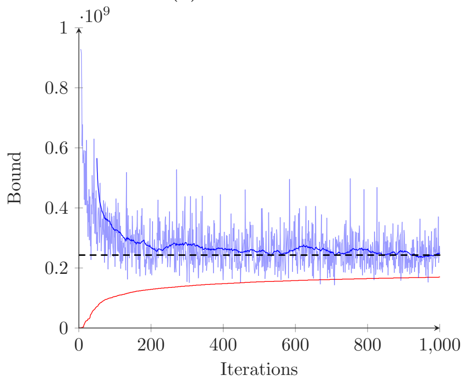
To analyze which algorithmic steps are most time consuming in practice, we acquire data on the time requirements for individual steps. The results are presented in Sect. F of the EC. It is evident that two steps are critical. First, the computation of factors $\Big(\sum_{r \in R_{t+1}} \rho^*_{rtj}\big(\alpha^{(\tau)}_{r,t+1,\ell}\big)\Big)$. Second, the re-evaluation of the cut intercepts for each cut and each given scenario. Both steps have in common that they require to iterate over all existing cuts, *i.e.*, $r \in R_{t+1}$.



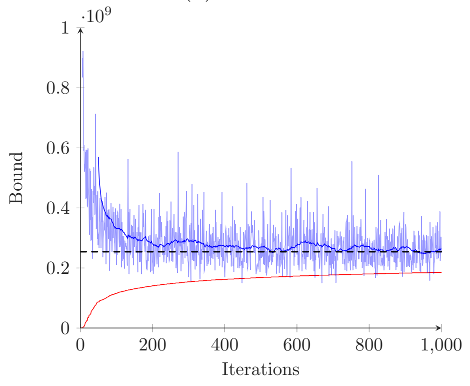
(a) `LOG-BIC`.



(b) `LOG-1`.



(c) `LIN-FIT`.



(d) `LIN-SHA`.

Figure 4: Development of lower bounds (LB) and simulated objective values over iterations of SDDP. Bounds are averaged over 5 independent runs. For the simulated values, also a moving-average with windows length 50 is plotted (see thick blue lines). The black dashed lines highlight the statistical UBs obtained in the in-sample simulation after SDDP has terminated.

5.3.2 Out-of-sample Simulation Performance

As mentioned before, for each policy obtained by running SDDP with a specific inflow model, we perform simulations using out-of-sample data for each of the other inflow models. Again, we present results for averages over 5 independent runs, *i.e.*, 5 different policies. In each run, for each policy, the same out-of-sample data is used. Fig. 6 illustrates the obtained average total costs (statistical UBs) and confidence intervals. We can see that the log-linear models consistently outperform the linearized ones, and yield better performing policies.

Assuming that the true inflows follow a log-linear PAR model, running our tailor-made version of SDDP instead of standard SDDP with a linearized inflow model leads to a 7-10% reduction of the average total cost in the out-of-sample simulations. These improvements are not limited to average costs: The cost distributions over all sample paths lean towards lower costs in general, as shown in Sect. F.1 of the EC.

An explanation for this observation is that the policies based on linear inflow models

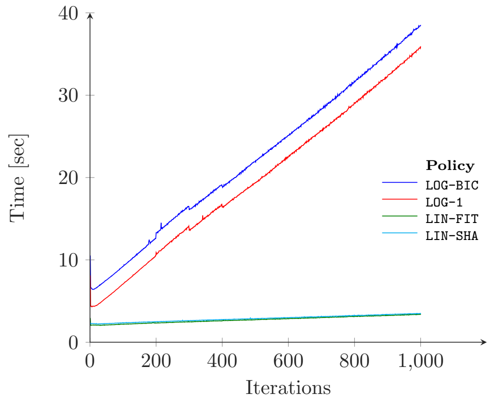


Figure 5: Time (in seconds) per iteration of SDDP.

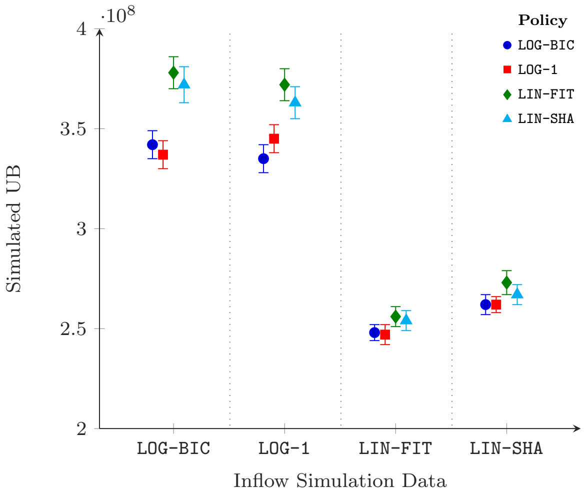


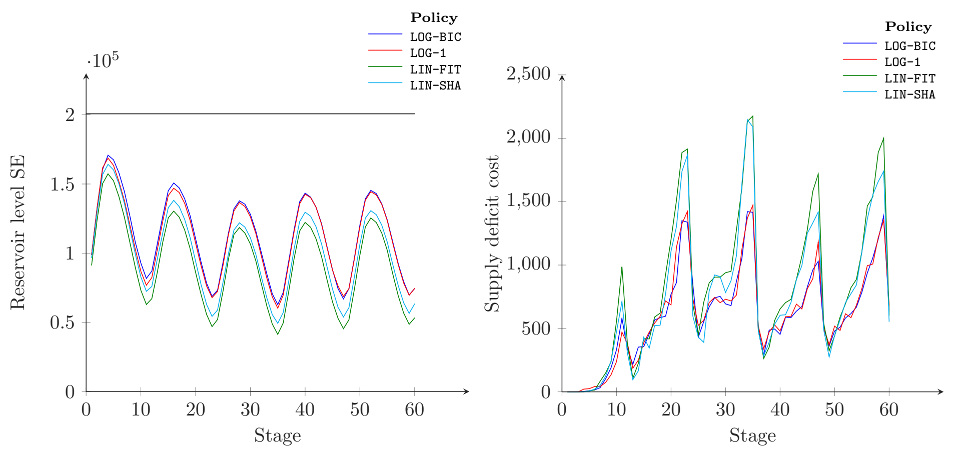
Figure 6: Average statistical upper bounds (and confidence intervals) in out-of-sample simulations over 5 independent runs.

tend to use more hydro power in water-rich periods, leading to lower reservoir levels and the occurrence of higher supply deficits in arid periods, as noticeable in Fig. 7. This, in turn, may be attributed to the policies being "trained" on a higher inflow level on average. However, interestingly, the policies based on log-linear inflow models even yield slightly lower costs on average when simulated with data from the linearized inflow models, *i.e.*, data exhibiting higher inflows on average. One possible explanation is that for both types of models, after 1000 iterations of SDDP, the obtained policies are still far from optimal, but with lower in-sample gaps for the log-linear models, see above.

5.4 Discussion

Our computational experiments show that our proposed version of SDDP works as intended, so they can be regarded as a proof of concept.

We observe that the performance of SDDP using log-linear and linearized inflow



(a) Reservoir volumes for system SE. (b) Supply deficit costs.

Figure 7: Average of reservoir volumes and supply deficit costs over the first 60 stages for 2000 out-of-sample simulations on `LOG-1` inflow data.

models is hard to compare, as the inflow levels differ significantly. Especially the obtained bounds cannot be directly compared. A comparison of the out-of-sample performance of the obtained policies is possible, though. Here, the policies obtained from the log-linear models yield superior results. In particular, assuming that the log-linear inflow model is correct, using standard SDDP with a linearization leads to a 7-10% cost increase in out-of-sample simulations compared to our proposed version of SDDP. This is important, as our model validation results indeed show that the log-linear models provide a more accurate description of the inflows based on their historical data.

In general, for both types of models, we observe a remarkable difference in total cost and obtained policies. This sensitivity underlines the importance of running SDDP with an appropriate inflow model. Our proposed approach allows for more flexibility, and thus potentially more accuracy in that regard.

Crucially, we experience a considerable computational overhead using our non-convex cuts in SDDP. This is mostly caused by the requirement to iterate over all existing cuts when computing cut intercept factors or adapting the cut intercepts to a given scenario. There exist several mechanisms to improve the performance, though. First, we conjecture that the computation of the cut intercept factors may be implemented in a more efficient way than we did. Second, the overhead from adapting the cut intercepts can be reduced by directly incorporating the non-convex cuts into the subproblems giving that solvers support this. Third, we show in Sect. G of the EC that our approach naturally allows for a hybrid SDDP where a log-linear AR process and non-convex cuts are used in early stages, while a coarser linearized process and linear cuts are used in later stages.

Finally, we notice that our experiments do not allow for an unobscured performance comparison between non-convex and linear cuts, and the associated versions of SDDP. The reason is that the considered inflow models also lead to differences in the inflow levels, and thus the considered models itself. Some tests using scaled inflows, see Sect. F.2 of the EC, indicate that for similar inflow levels also the performance of the policies obtained from both versions of SDDP is similar. However, exploring this in more detail is left for future research. To get a completely fair comparison, a case study with a convex log-linear AR process could be analyzed. In that case, the exact same inflows can be used in SDDP with either non-convex cuts or linear cuts.

6 Conclusion

In this paper, we explore in detail the extension of SDDP to stagewise dependent uncertainty in the RHS described by non-convex AR processes. This version of SDDP makes use of non-convex cuts to approximate the expected value functions. First, we show the computational challenges that come with these cuts for general processes. Then, we show how for log-linear AR processes closed-form cut formulas can be derived and incorporated into SDDP.

We perform computational tests on a LTHS problem, which show that our proposed version of SDDP works as intended, allows to incorporate more accurate inflow models into SDDP, and thus yields better performing policies than previously proposed approaches to linearize the log-linear processes.

In general, our contribution to the toolbox of SDDP allows for more flexibility, and therefore potentially more accuracy in modeling uncertainty in the RHS when dealing with multistage stochastic linear programs.

For future research, three extensions of our work seem natural. First, the efficiency of our implementation could be improved to reduce the computational overhead of using our proposed non-convex cuts. Second, experiments could be conducted for a case study with a convex log-linear AR process to get a clearer comparison of the quality of non-convex and linear cuts. Third, our proposed version of SDDP could be compared to the Markov chain SDDP approach from [18] that also allows for non-convex processes.

Acknowledgements

The authors thank Nils Löhndorf for providing the problem and inflow data for the case study analyzed in the computational tests.

A Proofs

A.1 Proof of Theorem 3.1

Proof. Consider stage T, some state $(\bar{x}_{T-1}, \bar{\xi}_{[T-1]})$, some $j \in \{1, \ldots, q_T\}$ and a corresponding optimal dual solution π^*_{Tj}. Using the same line of arguments as for standard SDDP (see Sect. D of the EC), *i.e.*, π^*_{Tj} being dual feasible, but not necessarily optimal for any $(x_{T-1}, \xi_{[T-1]})$, we obtain

$$\widetilde{Q}_T((x_{T-1}, \xi_{[T-1]}), \eta_T^{(j)}) \geq \underbrace{-(\pi^*_{Tj})^\top T_{T-1}}_{\overset{(12)}{=} \beta_{Tj}^\top} x_{T-1} + \underbrace{(\pi^*_{Tj})^\top b_T(\xi_{[T-1]}, \eta_T^{(j)})}_{\overset{(13)}{=} \alpha_{Tj}^{(T)}(\xi_{[T-1]})}.$$

Taking expectations, this yields the cut

$$\mathcal{Q}_T(x_{T-1}, \xi_{[T-1]}) \geq \underbrace{\sum_{j=1}^{q_T} p_{Tj} \beta_{Tj}}_{\overset{(11)}{=} \beta_T^\top} x_{T-1} + \underbrace{\sum_{j=1}^{q_T} p_{Tj} \alpha_{Tj}^{(T)}(\xi_{[T-1]})}_{\overset{(11)}{=} \alpha_T^{(T)}(\xi_{[T-1]})}$$

with scenario-dependent cut intercept $\alpha_T^{(T)}(\xi_{[T-1]})$. The use of the upper index (T) becomes evident in the following.

Going backwards, let R_T denote the set of stage-T cuts that have already been generated and added to stage $T-1$. Then, for some state $(\bar{x}_{T-2}, \bar{\xi}_{[T-2]})$ and some $j \in \{1, \ldots, q_{T-1}\}$, the dual objective function is

$$-\pi_{T-1}^\top T_{T-2} x_{T-2} + \pi_{T-1}^\top b_{T-1}(\xi_{[T-2]}, \eta_{T-1}^{(j)}) + \sum_{r \in R_T} \rho_{r,T-1} \alpha_{rT}^{(T)} \big(\widetilde{b}_{T-1}(\xi_{[T-2]}, \eta_{T-1}^{(j)})\big).$$

Using optimal dual solutions $\pi^*_{T-1,j}$ and $\rho^*_{r,T-1,j}$ for all $r \in R_T$, we obtain

$$\begin{aligned}\underline{\widetilde{Q}}_{T-1}\big((\bar{x}_{T-2}, \bar{\xi}_{[T-2]}), \eta_{T-1}^{(j)}\big) = {} & -(\pi^*_{T-1,j})^\top T_{T-2} \bar{x}_{T-2} + (\pi^*_{T-1,j})^\top b_{T-1}(\bar{\xi}_{[T-2]}, \eta_{T-1}^{(j)}) \\ & + \sum_{r \in R_T} \rho^*_{r,T-1,j} \alpha_{rT}^{(T)} \big(\widetilde{b}_{T-1}(\bar{\xi}_{[T-2]}, \eta_{T-1}^{(j)})\big).\end{aligned}$$

With the same reasoning as for stage T, it follows

$$\begin{aligned}\underline{\widetilde{Q}}_{T-1}\big((x_{T-2}, \xi_{[T-2]}), \eta_{T-1}^{(j)}\big) \geq {} & \underbrace{-(\pi^*_{T-1,j})^\top T_{T-2}}_{\overset{(12)}{=} \beta_{T-1,j}^\top} x_{T-2} + \underbrace{(\pi^*_{T-1,j})^\top b_{T-1}(\xi_{[T-2]}, \eta_{T-1}^{(j)})}_{\overset{(13)}{=} \alpha_{T-1,j}^{(T-1)}(\xi_{[T-2]})} \\ & + \underbrace{\sum_{r \in R_T} \rho^*_{r,T-1,j} \alpha_{rT}^{(T)} \big(\widetilde{b}_{T-1}(\xi_{[T-2]}, \eta_{T-1}^{(j)})\big)}_{\overset{(13)}{=} \alpha_{T-1,j}^{(T)}(\xi_{[T-2]})}.\end{aligned}$$

Here, $\alpha_{rT}^{(T)}(\cdot)$ denotes the scenario-dependent cut intercept related to cut r.

Since $\underline{\widetilde{Q}}_{T-1}(\cdot,\cdot) \leq \widetilde{Q}_{T-1}(\cdot,\cdot)$ and by taking expectations, this yields the cut

$$\mathcal{Q}_{T-1}(x_{T-2},\xi_{[T-2]}) \geq \underbrace{\sum_{j=1}^{q_{T-1}} p_{T-1,j}\beta_{T-1,j}^\top}_{\overset{(11)}{=}\beta_{T-1}^\top} x_{T-2} + \sum_{\tau=T-1}^{T} \underbrace{\sum_{j=1}^{q_{T-1}} p_{T-1,j}\alpha_{T-1,j}^{(\tau)}(\xi_{[T-2]})}_{\overset{(11)}{=}\alpha_{T-1}^{(\tau)}(\xi_{[T-2]})}.$$

Proceeding this way recursively, we obtain a general cut formula for some arbitrary stage t.

The tightness result follows with the same reasoning as for standard SDDP, see Sect. D in the EC, which is based on strong duality of linear programs and using optimal dual solutions to the subproblems. □

A.2 Proof of Corollary 3.2

Proof. We prove the result using backward induction.

Base case. For the base case, we consider the stage-T subproblem (8) for some fixed state $(\bar{x}_{T-1},\bar{\xi}_{[T-1]})$ and some arbitrary realization $\eta_T^{(j)}$, with $j\in\{1,\dots,q_T\}$. Using the dual objective to this subproblem and some dual optimal solution π_{Tj}^*, we obtain

$$\widetilde{Q}_T\big((\bar{x}_{T-1},\bar{\xi}_{[T-1]}),\eta_T^{(j)}\big) = (\pi_{Tj}^*)^\top\Big(b_T(\bar{\xi}_{[T-1]},\eta_T^{(j)}) - T_{T-1}\bar{x}_{T-1}\Big).$$

As the uncertainty only appears in the RHS of the subproblem, which translates to the objective of its dual problem, the dual feasible region is scenario-independent. Moreover, the dual feasible region is also independent of x_{T-1}. Therefore, π_{Tj}^* is feasible, but not necessarily optimal, for any state (x_{T-1},ξ_{T-1}) differing from $(\bar{x}_{T-1},\bar{\xi}_{[T-1]})$. We obtain

$$\begin{aligned}
\widetilde{Q}_T\big((x_{T-1},\xi_{[T-1]}),\eta_T^{(j)}\big) &\geq (\pi_{Tj}^*)^\top\Big(b_T(\bar{\xi}_{[T-1]},\eta_T^{(j)}) - T_{T-1}x_{T-1}\Big)\\
&= (\pi_{Tj}^*)^\top\left(\sum_{k=T-p}^{T-1}\Phi_T^{(T-k)}\xi_k + \eta_T - T_{T-1}x_{T-1}\right)\\
&= -(\pi_{Tj}^*)^\top T_{T-1}x_{T-1} + (\pi_{Tj}^*)^\top\left(\sum_{k=T-p}^{T-1}\Phi_T^{(T-k)}\xi_k\right) + (\pi_{Tj}^*)^\top\eta_T,
\end{aligned}$$

where we applied the process definition (15) in the pre-last step.

By applying the definitions in (17), (18) and (12), the last line can be reformulated, which yields

$$\widetilde{Q}_T\big((x_{T-1},\xi_{[T-1]}),\eta_T^{(j)}\big) \geq \beta_{Tj}^\top x_{T-1} + \sum_{k=T-p}^{T-1} C_{Tj}^{(T,T-k)} + \omega_{Tj}^{(T)}$$

According to (19), by taking expectations with respect to $\boldsymbol{\eta}_T$, it follows

$$\begin{aligned}
\mathcal{Q}_T(x_{T-1},\xi_{[T-1]}) &\geq \beta_T^\top x_{T-1} + \sum_{k=T-p}^{T-1} C_T^{(T,T-k)} + \omega_T^{(T)}\\
&\overset{(16)}{=} \beta_T^\top x_{T-1} + \alpha_T^{(T)}(\xi_{[T-1]}).
\end{aligned}$$

This cut has the same form of formula (11), where all but one summand related to τ vanish for $t = T$.

Induction step. We consider some arbitrary stage $t \in \{2, \dots, T-1\}$, and assume that cut formula (11) holds for stage $t+1$. More precisely, we assume that R_{t+1} cuts of form (11), (16)-(19) have been generated for stage $t+1$ so far. We prove that the assertion then also holds for stage t.

As a preparation step, we use formula (16) to express $\alpha^{(\tau)}_{r,t+1}(\xi_{[t]})$ through $\xi_{[t-1]}$ for all $r \in R_{t+1}$ and $\tau = t+1, \dots, T$. We obtain

$$\begin{aligned}
&\alpha^{(\tau)}_{r,t+1}(\xi_{[t]}) \\
&= \sum_{k=t+1-p}^{t} C^{(\tau,t+1-k)}_{r,t+1}\xi_k + \omega^{(\tau)}_{r,t+1} \\
&= \sum_{k=t+1-p}^{t-1} C^{(\tau,t+1-k)}_{r,t+1}\xi_k + \omega^{(\tau)}_{r,t+1} + C^{(\tau,1)}_{r,t+1}\xi_t \\
&\overset{(15)}{=} \sum_{k=t+1-p}^{t-1} C^{(\tau,t+1-k)}_{r,t+1}\xi_k + \omega^{(\tau)}_{r,t+1} + C^{(\tau,1)}_{r,t+1}\left(\sum_{k=t-p}^{t-1} \Phi^{(t-k)}_t \xi_k + \eta_t\right) \\
&= \sum_{k=t+1-p}^{t-1} \big(C^{(\tau,t+1-k)}_{r,t+1} + C^{(\tau,1)}_{r,t+1}\Phi^{(t-k)}_t\big)\xi_k + C^{(\tau,1)}_{r,t+1}\Phi^{(p)}_t \xi_{t-p} + \omega^{(\tau)}_{r,t+1} + C^{(\tau,1)}_{r,t+1}\eta_t.
\end{aligned} \tag{39}$$

The dual objective function of subproblem (8) at stage t is

$$-\pi_t^\top T_{t-1}x_{t-1} + \pi_t^\top \xi_t + \sum_{r \in R_{t+1}} \rho_{rt}\left(\sum_{\tau=t+1}^{T} \alpha^{(\tau)}_{r,t+1}(\xi_{[t]})\right).$$

Exploiting (39) and the process definition (15), this can be rewritten as

$$\begin{aligned}
&-\pi_t^\top T_{t-1}x_{t-1} + \pi_t^\top\left(\sum_{k=t-p}^{t-1} \Phi^{(t-k)}_t \xi_k + \eta_t\right) + \sum_{r \in R_{t+1}} \rho_{rt}\Bigg(\sum_{\tau=t+1}^{T}\Bigg(\\
&\quad \sum_{k=t+1-p}^{t-1} \big(C^{(\tau,t+1-k)}_{r,t+1} + C^{(\tau,1)}_{r,t+1}\Phi^{(t-k)}_t\big)\xi_k + C^{(\tau,1)}_{r,t+1}\Phi^{(p)}_t\xi_{t-p} + \omega^{(\tau)}_{r,t+1} + C^{(\tau,1)}_{r,t+1}\eta_t\Bigg)\Bigg).
\end{aligned}$$

For the next step, consider some fixed state $(\bar{x}_{t-1}, \bar{\xi}_{[t-1]})$ and some arbitrary realization $\eta^{(j)}_t, j \in \{1, \dots, q_t\}$, and let π^*_{tj} and $\rho^*_{rtj}, r \in R_{t+1}$, denote the corresponding optimal dual solutions. Then, using the same reasoning as for the base case, we obtain

$$
\begin{aligned}
&\widetilde{Q}_t\big((x_{t-1}, \xi_{[t-1]}), \eta_t^{(j)}\big) \\
&\geq \underbrace{-(\pi_{tj}^*)^\top T_{t-1}}_{\overset{(12)}{=}\beta_{tj}^\top} x_{t-1} + (\pi_{tj}^*)^\top \left(\sum_{k=t-p}^{t-1} \Phi_t^{(t-k)} \xi_k + \eta_t^{(j)} \right) + \sum_{r \in R_{t+1}} \rho_{rtj}^* \Bigg(\sum_{\tau=t+1}^{T} \Bigg(\\
&\qquad \sum_{k=t+1-p}^{t-1} \big(C_{r,t+1}^{(\tau,t+1-k)} + C_{r,t+1}^{(\tau,1)} \Phi_t^{(t-k)} \big) \xi_k + C_{r,t+1}^{(\tau,1)} \Phi_t^{(p)} \xi_{t-p} + \omega_{r,t+1}^{(\tau)} + C_{r,t+1}^{(\tau,1)} \eta_t^{(j)} \Bigg) \Bigg) \\
&= \beta_{tj}^\top x_{t-1} + \sum_{k=t-p}^{t-1} \underbrace{(\pi_{tj}^*)^\top \Phi_t^{(t-k)}}_{\overset{(17)}{=} C_{tj}^{(t,t-k)}} \xi_k + \underbrace{(\pi_{tj}^*)^\top \eta_t^{(j)}}_{\overset{(18)}{=} \omega_{tj}^{(t)}} + \sum_{r \in R_{t+1}} \rho_{rtj}^* \left(\sum_{\tau=t+1}^{T} \omega_{r,t+1}^{(\tau)} + C_{r,t+1}^{(\tau,1)} \eta_t^{(j)} \right) \\
&\qquad + \sum_{r \in R_{t+1}} \rho_{rtj}^* \left(\sum_{\tau=t+1}^{T} \left(\sum_{k=t+1-p}^{t-1} \big(C_{r,t+1}^{(\tau,t+1-k)} + C_{r,t+1}^{(\tau,1)} \Phi_t^{(t-k)} \big) \xi_k + C_{r,t+1}^{(\tau,1)} \Phi_t^{(p)} \xi_{t-p} \right) \right).
\end{aligned}
$$

By distributivity and exchanging sums, we can reformulate the last two lines and obtain

$$
\begin{aligned}
&\widetilde{Q}_t\big((x_{t-1}, \xi_{[t-1]}), \eta_t^{(j)}\big) \\
&\geq \beta_{tj}^\top x_{t-1} + \sum_{k=t-p}^{t-1} C_{tj}^{(t,t-k)} \xi_k + \omega_{tj}^{(t)} + \sum_{\tau=t+1}^{T} \underbrace{\sum_{r \in R_{t+1}} \rho_{rtj}^* \big(\omega_{r,t+1}^{(\tau)} + C_{r,t+1}^{(\tau,1)} \eta_t^{(j)} \big)}_{\overset{(18)}{=} \omega_{tj}^{(\tau)}} \\
&\qquad + \sum_{\tau=t+1}^{T} \sum_{k=t+1-p}^{t-1} \Bigg(\underbrace{\sum_{r \in R_{t+1}} \rho_{rtj}^* \Big(C_{r,t+1}^{(\tau,t+1-k)} + C_{r,t+1}^{(\tau,1)} \Phi_t^{(t-k)} \Big)}_{\overset{(17)}{=} C_{tj}^{(\tau,t-k)}} \Bigg) \xi_k \\
&\qquad + \sum_{\tau=t+1}^{T} \underbrace{\sum_{r \in R_{t+1}} \rho_{rtj}^* C_{r,t+1}^{(\tau,1)} \Phi_t^{(p)}}_{\overset{(17)}{=} C_{tj}^{(\tau,p)}} \xi_{t-p} \\
&= \beta_{tj}^\top x_{t-1} + \sum_{k=t-p}^{t-1} C_{tj}^{(t,t-k)} \xi_k + \omega_{tj}^{(t)} + \sum_{\tau=t+1}^{T} \left(\omega_{tj}^{(\tau)} + \sum_{k=t+1-p}^{t-1} C_{tj}^{(\tau,t-k)} \xi_k + C_{tj}^{(\tau,p)} \xi_{t-p} \right) \\
&= \beta_{tj}^\top x_{t-1} + \sum_{\tau=t}^{T} \omega_{tj}^{(\tau)} + \sum_{k=t-p}^{t-1} C_{tj}^{(t,t-k)} \xi_k + \sum_{\tau=t+1}^{T} \sum_{k=t-p}^{t-1} C_{tj}^{(\tau,t-k)} \xi_k \\
&= \beta_{tj}^\top x_{t-1} + \sum_{\tau=t}^{T} \left(\omega_{tj}^{(\tau)} + \sum_{k=t-p}^{t-1} C_{tj}^{(\tau,t-k)} \xi_k \right)
\end{aligned}
$$

According to (19), taking expectations with respect to $\boldsymbol{\eta}_t$ on both sides yields

$$\mathcal{Q}_t(x_{t-1}, \xi_{[t-1]}) \geq \beta_t^\top x_{t-1} + \sum_{\tau=t}^{T} \underbrace{\left(\omega_t^{(\tau)} + \sum_{k=t-p}^{t-1} C_t^{(\tau,t-k)} \xi_k \right)}_{\overset{(16)}{=} \alpha_t^{(\tau)}(\xi_{[t-1]})}$$

Again, this corresponds to cut formula (11). □

A.3 Proof of Theorem 4.2

Proof. We prove the result using backward induction.

Base case. For the base case, we consider the stage-T subproblem for some fixed state $(\bar{x}_{T-1}, \bar{\xi}_{[T-p:T-1]})$ and some arbitrary realization $\eta_T^{(j)}$, with $j \in \{1, \ldots, q_T\}$. Using the dual objective to this subproblem and some dual optimal solution π_{Tj}^*, we obtain

$$\widetilde{Q}_T\big((\bar{x}_{T-1}, \bar{\xi}_{[T-p:T-1]}), \eta_T^{(j)}\big) = (\pi_{Tj}^*)^\top \Big(b_T^{(p)}(\bar{\xi}_{[T-p:T-1]}, \eta_T^{(j)}) - T_{T-1}\bar{x}_{T-1} \Big).$$

As the uncertainty only appears in the RHS of the subproblem, which translates to the objective of its dual problem, the dual feasible region is scenario-independent. Moreover, the dual feasible region is also independent of x_{T-1}. Therefore, π_{Tj}^* is feasible, but not necessarily optimal, for any state $(x_{T-1}, \xi_{[T-p:T-1]})$ differing from $(\bar{x}_{T-1}, \bar{\xi}_{[T-p:T-1]})$. We obtain

$$\widetilde{Q}_T\big((x_{T-1}, \xi_{[T-p:T-1]}), \eta_T^{(j)}\big) \geq (\pi_{Tj}^*)^\top \Big(b_T^{(p)}(\bar{\xi}_{[T-p:T-1]}, \eta_T^{(j)}) - T_{T-1}x_{T-1} \Big).$$

By applying the definitions in (24), (27) and (29), it follows

$$\begin{aligned}
&\widetilde{Q}_T\big((x_{T-1}, \xi_{[T-p:T-1]}), \eta_T^{(j)}\big) \\
&\overset{(24)}{\geq} -(\pi_{Tj}^*)^\top T_{T-1}x_{T-1} + \sum_{\ell=1}^{L_T} \pi_{T\ell j}^* e^{\gamma_{T\ell}} e^{\psi_{T\ell}\eta_{T\ell}^{(j)}} \prod_{k=1}^{p} \prod_{m=1}^{L_{T-k}} \xi_{T-k,m}^{\phi_{T\ell m}^{(k)}} \\
&\overset{(27)}{=} \beta_{Tj}^\top x_{T-1} + \sum_{\ell=1}^{L_T} \alpha_{T\ell j}^{(T)} \prod_{k=1}^{p} \prod_{m=1}^{L_{T-k}} \xi_{T-k,m}^{\phi_{T\ell m}^{(k)}} \\
&= \beta_{Tj}^\top x_{T-1} + \sum_{\ell=1}^{L_T} \alpha_{T\ell j}^{(T)} \prod_{k=T-p}^{T-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{T\ell m}^{(T-k)}} \\
&\overset{(29)}{=} \beta_{Tj}^\top x_{T-1} + \sum_{\ell=1}^{L_T} \alpha_{T\ell j}^{(T)} \prod_{k=T-p}^{T-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(T,T,\ell,m,k)}
\end{aligned}$$

where we rearranged some indices in the prelast step.

Using expectations over all $\eta_T^{(j)}, j = 1, \ldots, q_T$, this cut has the same form as stated in (30), where all but one summand related to τ vanish for $t = T$.

Induction step. We consider some arbitrary stage $t \in \{2, \ldots, T-1\}$, and assume that cut formula (30) holds for stage $t+1$. More precisely, we assume that R_{t+1} cuts of form (30) have been generated for stage $t+1$ so far. We prove that the assertion then also holds for stage t.

Using cut formula (30), the stage-t subproblem (8) can be written as

$$\begin{cases} \min\limits_{x_t,\theta_{t+1}} & c_t^\top x_t + \theta_{t+1} \\ \text{s.t.} & W_t x_t = -T_{t-1}x_{t-1} + \xi_t \\ & \theta_{t+1} - \beta_{t+1}^\top x_t \geq \sum\limits_{\tau=t+1}^{T} \sum\limits_{\ell=1}^{L_\tau} \Big(\alpha_{r,t+1,\ell}^{(\tau)} \prod\limits_{k=t+1-p}^{t} \prod\limits_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Big), \quad r \in R_{t+1} \\ & x_t \geq 0. \end{cases}$$

Hence, the objective function of its dual is

$$-\pi_t^\top T_{t-1}x_{t-1} + \pi_t^\top \xi_t + \sum_{r\in R_{t+1}} \rho_{rt} \Big(\sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Big(\alpha_{r,t+1,\ell}^{(\tau)} \prod_{k=t+1-p}^{t} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Big) \Big).$$

For the next step, consider some fixed state $(\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]})$ and some arbitrary realization $\eta_t^{(j)}, j \in \{1,\dots,q_t\}$. Representing each occurrence of $\xi_{t\ell}$ by $b_{t\ell}^{(p)}(\bar{\xi}_{[t-p:t-1]}, \eta_t^{(j)})$ yields

$$\begin{aligned} &-\pi_t^\top T_{t-1}\bar{x}_{t-1} + \sum_{\ell=1}^{L_t} \pi_{t\ell} b_{t\ell}^{(p)}(\bar{\xi}_{[t-p:t-1]}, \eta_t^{(j)}) \\ &+ \sum_{r\in R_{t+1}} \rho_{rt} \Big(\sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Big(\alpha_{r,t+1,\ell}^{(\tau)} \prod_{\nu=1}^{L_t} \Big(b_{t\nu}^{(p)}(\bar{\xi}_{[t-p:t-1]}, \eta_t^{(j)}) \Big)^{\Theta(t+1,\tau,\ell,\nu,t)} \\ &\cdot \prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \bar{\xi}_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Big) \Big). \end{aligned}$$

Let a dual optimal solution for this case be given by π_{tj}^* and $\rho_{rtj}^*, r \in R_{t+1}$. Then, in accordance with the definition in (8), we obtain

$$\begin{aligned} &\underline{\widetilde{Q}}_t\big((\bar{x}_{t-1}, \bar{\xi}_{[t-p:t-1]}), \eta_t^{(j)}\big) \\ &= \underbrace{-(\pi_{tj}^*)^\top T_{t-1}}_{\overset{(27)}{=} \beta_{tj}^\top} \bar{x}_{t-1} + \sum_{\ell=1}^{L_t} \pi_{t\ell j}^* b_{t\ell}^{(p)}(\bar{\xi}_{[t-p:t-1]}, \eta_t^{(j)}) \\ &+ \sum_{r\in R_{t+1}} \rho_{rtj}^* \Big(\sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Big(\alpha_{r,t+1,\ell}^{(\tau)} \prod_{\nu=1}^{L_t} \Big(b_{t\nu}^{(p)}(\bar{\xi}_{[t-p:t-1]}, \eta_t^{(j)}) \Big)^{\Theta(t+1,\tau,\ell,\nu,t)} \\ &\cdot \prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Big) \Big). \end{aligned} \tag{40}$$

Exploiting that $\underline{\widetilde{Q}}_t(\cdot,\cdot)$ underestimates $\widetilde{Q}_t(\cdot,\cdot)$, and using the same reasoning on dual feasibility and optimality as in the base case, we can conclude that

$$\begin{aligned}
&\widetilde{Q}_t\big((x_{t-1}, \xi_{[t-p:t-1]}), \eta_t^{(j)}\big)\\
&\geq \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \pi_{t\ell j}^* b_{t\ell}^{(p)}(\xi_{[t-p:t-1]}, \eta_t^{(j)})\\
&+ \sum_{r\in R_{t+1}} \rho_{rtj}^* \Bigg(\sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Bigg(\alpha_{r,t+1,\ell}^{(\tau)} \prod_{\nu=1}^{L_t} \Big(b_{t\nu}^{(p)}(\xi_{[t-p:t-1]}, \eta_t^{(j)}) \Big)^{\Theta(t+1,\tau,\ell,\nu,t)}\\
&\cdot \prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\Bigg).
\end{aligned}$$

With the definitions in (24) and (27), the RHS can be reformulated, and we obtain

$$\begin{aligned}
&\widetilde{Q}_t\big((x_{t-1}, \xi_{[t-p:t-1]}), \eta_t^{(j)}\big)\\
&\geq \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \underbrace{\pi_{tj\ell}^* e^{\gamma_{t\ell}} e^{\psi_{t\ell}\eta_{t\ell}^{(j)}}}_{\overset{(27)}{=}\alpha_{t\ell j}^{(t)}} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\ell m}^{t-k}} \Bigg)\\
&+ \sum_{r\in R_{t+1}} \rho_{rtj}^* \Bigg(\sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Bigg(\alpha_{r,t+1,\ell}^{(\tau)} \prod_{\nu=1}^{L_t} \Big(e^{\gamma_{t\nu}} e^{\psi_{t\nu}\eta_{t\nu}^{(j)}} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\nu m}^{(t-k)}} \Big)^{\Theta(t+1,\tau,\ell,\nu,t)}\\
&\cdot \prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\Bigg)\\
&= \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\ell m}^{t-k}} \Bigg)\\
&+ \sum_{r\in R_{t+1}} \rho_{rtj}^* \Bigg(\sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Bigg(\alpha_{r,t+1,\ell}^{(\tau)} \prod_{\nu=1}^{L_t} e^{\gamma_{t\nu}\Theta(t+1,\tau,\ell,\nu,t)} e^{\psi_{t\nu}\eta_{t\nu}^{(j)}\Theta(t+1,\tau,\ell,\nu,t)}\\
&\cdot \Bigg(\prod_{\nu=1}^{L_k} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\nu m}^{(t-k)}\Theta(t+1,\tau,\ell,\nu,t)} \Bigg) \prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\Bigg). \qquad (41)
\end{aligned}$$

By distributivity and exchanging sums, the RHS of (41) is equivalent to

$$\begin{aligned}
&\beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\ell m}^{t-k}} \Bigg)\\
&+ \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \Bigg(\underbrace{\Bigg(\sum_{r\in R_{t+1}} \rho_{rtj}^* \alpha_{r,t+1,\ell}^{(\tau)} \Bigg) \prod_{\nu=1}^{L_t} e^{\gamma_{t\nu}\Theta(t+1,\tau,\ell,\nu,t)} e^{\eta_{t\nu}^{(j)}\Theta(t+1,\tau,\ell,\nu,t)}}_{\overset{(27)}{=}\alpha_{t\ell j}^{(\tau)}}\\
&\cdot \Bigg(\prod_{\nu=1}^{L_t} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\nu m}^{(t-k)}\Theta(t+1,\tau,\ell,\nu,t)} \Bigg) \Bigg(\prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\Bigg).
\end{aligned}$$

Therefore, using rules of powers and the definitions in (29), it follows

$$
\begin{aligned}
&\widetilde{Q}_t\big((x_{t-1}, \xi_{[t-p:t-1]}), \eta_t^{(j)}\big)\\
&\geq \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\ell m}^{(t-k)}} \Bigg)\\
&+ \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)} \Bigg(\prod_{\nu=1}^{L_t} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\nu m}^{(t-k)} \Theta(t+1,\tau,\ell,\nu,t)} \Bigg) \Bigg(\prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\\
&\overset{(29)}{=} \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,t,\ell,m,k)} \Bigg)\\
&+ \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)} \Bigg(\prod_{\nu=1}^{L_t} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\phi_{t\nu m}^{(t-k)} \Theta(t+1,\tau,\ell,\nu,t)} \Bigg) \Bigg(\prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\\
&\overset{(\#)}{=} \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,t,\ell,m,k)} \Bigg)\\
&+ \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\sum_{\nu=1}^{L_t} \phi_{t\nu m}^{(t-k)} \Theta(t+1,\tau,\ell,\nu,t)} \Bigg) \Bigg(\prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t+1,\tau,\ell,m,k)} \Bigg)\\
&\overset{(*)}{=} \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,t,\ell,m,k)} \Bigg) + \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)}\\
&\cdot \Bigg(\prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\sum_{\nu=1}^{L_t} \phi_{t\nu m}^{(t-k)} \Theta(t+1,\tau,\ell,\nu,t) + \Theta(t+1,\tau,\ell,m,k)} \Bigg) \Bigg(\prod_{m=1}^{L_{t-p}} \xi_{t-p,m}^{\sum_{\nu=1}^{L_t} \phi_{t\nu m}^{(p)} \Theta(t+1,\tau,\ell,\nu,t)} \Bigg)\\
&\overset{(29)}{=} \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,t,\ell,m,k)} \Bigg)\\
&+ \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)} \Bigg(\prod_{k=t+1-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,\tau,\ell,m,k)} \Bigg) \Bigg(\prod_{m=1}^{L_{t-p}} \xi_{t-p,m}^{\Theta(t,\tau,\ell,m,t-p)} \Bigg)\\
&= \beta_{tj}^\top x_{t-1} + \sum_{\ell=1}^{L_t} \alpha_{t\ell j}^{(t)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,t,\ell,m,k)} \Bigg)\\
&+ \sum_{\tau=t+1}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)} \Bigg(\prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,\tau,\ell,m,k)} \Bigg)\\
&= \beta_{tj}^\top x_{t-1} + \sum_{\tau=t}^{T} \sum_{\ell=1}^{L_\tau} \alpha_{t\ell j}^{(\tau)} \prod_{k=t-p}^{t-1} \prod_{m=1}^{L_k} \xi_{km}^{\Theta(t,\tau,\ell,m,k)}.
\end{aligned}
$$

By taking expectations on both sides with respect to η_t, we obtain cut formula (30), which proves the assertion. □

Remark A.1. *For completeness, let us analyze the proof of Theorem 4.2 in the light of Remark 4.1 (4). If p is not assumed constant, but depends on t, ℓ and m, then first, the two product operators in the process definition* (24) *have to be swapped due to the dependencies of p. More crucially, the steps in the proof marked by* ($\#$) *and* ($*$) *do* *not work anymore. In step (#), the product over ν cannot be reformulated to a sum in the exponents, as p depends on ν. In step $(*)$, factors from different product operators cannot be subsumed, as their index sets differ. Consequently, the obtained cut formulas become considerably more complicated. For this reason, as described in Remark 4.1, it is beneficial to assume a sufficiently large, constant lag order p and to introduce zero coefficients to compensate for it.*

B Notation

Tables 3 and 4 summarize the symbols that are used throughout the paper.

Table 3: Notation used in the paper - Part 1.

Problem parameters	
T:	time horizon, last stage
t:	time index
$[t]$:	horizon from stages 1 to t
n_t:	dimension of state/decision space
c_t:	objective coefficients at stage t
T_{t-1}:	technology matrix at stage t
W_t:	recourse matrix at stage t
h_t:	RHS at stage t
$L_t, L_{[t]}$:	no. of constraints at stage t / from stages 1 to t
Uncertainty modeling	
$\mathcal{F}_t, \mathcal{F}$:	σ-algebra at stage t / end of horizon
Ω_t, Ω:	sample space at stage t / over whole horizon
$\mathbb{P}$:	probability measure
p_{tj}	probability of realization j at stage t
$\boldsymbol{\xi}_t, \xi_t$:	random vector / realization at stage t
$\boldsymbol{\eta}_t, \eta_t$:	stagewise independent random vector / realization at stage t
q_t:	no. of realizations of $\boldsymbol{\xi}_t/\boldsymbol{\eta}_t$
$\xi_t^{(j)}, \eta_t^{(j)}$:	j-th realization of $\boldsymbol{\xi}_t/\boldsymbol{\eta}_t$, with $j \in \{1, \dots, q_t\}$
$\xi_{[t]}, \xi_{[k:t]}$:	history of $(\boldsymbol{\xi}_t)_{t\in[T]}$ up to stage / between k and t
ξ_t^i:	sampled realization at iteration i
$b_t(\cdot,\cdot)$:	function in $\xi_{[t-1]}$ and η_t describing ξ_t
$\tilde{b}_t(\cdot,\cdot)$:	function in $\xi_{[t-1]}$ and η_t describing $\xi_{[t]}$
$\widehat{b}_t(\cdot,\cdot)$:	linear approximation of $b_t(\cdot,\cdot)$
$\gamma_t, \gamma_{t\ell}$:	intercept at stage t (for component ℓ)
$\psi_t, \psi_{t\ell}$:	coefficient at stage t for η_t (for component ℓ)
$\phi_{t\ell m}^{(k)}$:	coefficient at stage t for lag k, components ℓ and m
$\Phi_t^{(k)}$:	coefficient matrix for linear AR process at stage t for lag k
p:	lag order
Decision variables	
x_t:	state/decision variable at stage t
$\theta_{t+1}, \underline{\theta}_{t+1}$:	epigraph variable at stage t and initial lower bound
π_{tj}:	dual variable for for realization j at stage t
ρ_{rtj}:	dual variable for cut r for realization j at stage t

Table 4: Notation used in the paper - Part 2.

Optimal solutions	
v^*:	optimal value of MSLP
x_t^i:	optimal x_t at iteration i
π_{tj}^*, π_{tj}^i:	optimal π_{tj} (at iteration i)
$\rho_{rtj}^*, \rho_{rtj}^i$:	optimal ρ_{rtj} (at iteration i)
Value functions	
$Q_t(\cdot,\cdot)$:	stage-t value function
$\mathcal{Q}_t(\cdot,\cdot)$:	stage-t expected value function
$\widetilde{Q}_t(\cdot,\cdot)$:	stage-t value function (expanding the state space)
$\widehat{\mathcal{Q}}_t(\cdot,\cdot)$:	stage-t expected value function (Taylor approximation)
$\underline{Q}_t^i(\cdot,\cdot)$:	approx. stage-t value function at iteration i
$\underline{\widetilde{Q}}_t^i(\cdot,\cdot)$:	approx. stage-t value function at iteration i (expanding the state space)
$\underline{\mathcal{Q}}_t^i(\cdot)$:	expected approx. stage-t value function at iteration i
$\mathfrak{Q}_t^i(\cdot)$:	cut approximation for stage-t at iteration i
SDDP in general	
i:	iteration index
$\underline{v}^i, \overline{v}^i$:	lower bound / upper bound for v^* at iteration i
$\alpha_t, \alpha_{rt}, \alpha_{tj}$:	cut intercept for stage t (and cut r / realization j)
$\beta_t, \beta_{rt}, \beta_{tj}$:	cut gradient for stage t (and cut r / realization j)
R_{t+1}, R_{t+1}^i:	no. of cuts at stage t (and iteration i)
$\alpha_t^{\text{ind}}, \alpha_t^{\text{dep}}(\xi_{[t-1]})$:	scenario-independent / dependent part of incercept for stage t
SDDP with linear AR process	
$C_{tj}^{(\tau,t-k)}$:	intercept matrix for stage t, lag k and realization j
$C_t^{(\tau,t-k)}, C_{rt}^{(\tau,t-k)}$:	intercept matrix for stage t, lag k (and cut r)
$\omega_t^{(\tau)}, \omega_{rt}^{(\tau)}, \omega_{tj}^{(\tau)}$:	intercept vector for stage t (and cut r / realization j)
SDDP with log-linear AR process	
$\alpha_t^{(\tau)}, \alpha_{rt}^{(\tau)}, \alpha_{tj}^{(\tau)}$:	intercept factor for stage t and cut r / realization j)
$\Theta(t,\tau,\ell,m,k)$:	exponent for stage t, intercept factor τ, components ℓ, m and lag k
τ:	index for intercept factors
Operators	
$\odot$:	Hadamard product (componentwise product) of vectors
$\bigodot_{i=1}^k$:	indexed sequence of Hadamard products

References

[1] D. Ávila, A. Papavasiliou, and N. Löhndorf. Batch learning SDDP for long-term hydrothermal planning. *IEEE Transactions on Power Systems*, 39(1):614–627, 2024.

[2] R. E. Bellman. *Dynamic programming*. Princeton University Press, Princeton, New Jersey, 1957.

[3] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: a fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[4] V. L. de Matos and E. C. Finardi. A computational study of a stochastic optimization model for long term hydrothermal scheduling. *International Journal of Electrical Power & Energy Systems*, 43(1):1443–1452, 2012.

[5] V. L. de Matos, A. B. Philpott, and E. C. Finardi. Improving the performance of Stochastic Dual Dynamic Programming. *Journal of Computational and Applied Mathematics*, 290:196–208, 2015.

[6] A. Downward, O. Dowson, and R. Baucke. Stochastic dual dynamic programming with stagewise-dependent objective uncertainty. *Operations Research Letters*, 48(1):33–39, 2020.

[7] O. Dowson and L. Kapelevich. SDDP.jl: a Julia package for stochastic dual dynamic programming. *INFORMS Journal on Computing*, 33(1):27–33, 2020.

[8] I. Dunning, J. Huchette, and M. Lubin. JuMP: a modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[9] D. Duque and D. P. Morton. Distributionally robust stochastic dual dynamic programming. *SIAM Journal on Optimization*, 30(4):2841–2865, 2020.

[10] C. Füllner and S. Rebennack. Non-convex nested Benders decomposition. *Mathematical Programming*, 196:987–1024, 2022.

[11] C. Füllner and S. Rebennack. Stochastic dual dynamic programming and its variants – A review. Preprint, available at `http://www.optimization-online.org/DB_FILE/2021/01/8217.pdf`, 2023.

[12] P. Girardeau, V. Leclère, and A. B. Philpott. On the convergence of decomposition methods for multistage stochastic convex programs. *Mathematics of Operations Research*, 40(1):130–145, 2015.

[13] V. Guigues. SDDP for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57:167–203, 2014.

[14] V. Guigues. Inexact cuts in stochastic dual dynamic programming. *SIAM Journal on Optimization*, 30(1):407–438, 2020.

[15] V. Guigues, A. Shapiro, and Y. Cheng. Duality and sensitivity analysis of multistage linear stochastic programs. *European Journal of Operational Research*, 308(2):752–767, 2023.

[16] G. Infanger and D. P. Morton. Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75:241–256, 1996.

[17] T. Lohmann, A. S. Hering, and S. Rebennack. Spatio-temporal hydro forecasting of multireservoir inflows for hydro-thermal scheduling. *European Journal of Operational Research*, 255(1):243–258, 2016.

[18] N. Löhndorf and A. Shapiro. Modeling time-dependent randomness in stochastic dual dynamic programming. *European Journal of Operational Research*, 273(2):650–661, 2019.

[19] M. E. P. Maceira, D. D. J. Penna, A. L. Diniz, R. J. Pinto, A. C. G. Melo, C. V. Vasconcellos, and C. B. Cruz. Twenty years of application of stochastic dual dynamic programming in official and agent studies in Brazil: main features and improvements on the NEWAVE model. In *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018.

[20] M. V. F. Pereira and L. M. V. G. Pinto. Multi-stage stochastic optimization applied to energy planning. *Mathematical Programming*, 52:359–375, 1991.

[21] A. B. Philpott, V. L. de Matos, and E. Finardi. On solving multistage stochastic programs with coherent risk measures. *Operations Research*, 61(4):957–970, 2013.

[22] A. R. Queiroz and D. P. Morton. Sharing cuts under aggregated forecast when decomposing multi-stage stochastic programs. *Operations Research Letters*, 41(3):311–316, 2013.

[23] S. Rebennack. Combining sampling-based and scenario-based nested Benders decomposition methods: application to stochastic dual dynamic programming. *Mathematical Programming*, 156:343–389, 2016.

[24] A. Shapiro. Analysis of stochastic dual dynamic programming method. *European Journal of Operational Research*, 209(1):63–72, 2011.

[25] A. Shapiro, D. Dentcheva, and A. Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory, Second Edition.* Society for Industrial and Applied Mathematics, 2014.

[26] A. Shapiro and W. Tekaya. Report for technical cooperation between Georgia Institute of Technology and ONS – Operador Nacional do Sistema Eletrico – risk averse approach. Technical report, 2011.

[27] A. Shapiro, W. Tekaya, J. P. da Costa, and M. P. Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391, 2013.

[28] G. Steeger, T. Lohmann, and S. Rebennack. Strategic bidding for a price-maker hydroelectric producer: stochastic dual dynamic programming and Lagrangian relaxation. *IISE Transactions*, 50(11):929–942, 2018.

[29] W. van Ackooij, W. de Oliveira, and Y. Song. On level regularization with normal solutions in decomposition methods for multistage stochastic programming problems. *Computational Optimization and Applications*, 74:1–42, 2019.

[30] W. van Ackooij and X. Warin. On conditional cuts for stochastic dual dynamic programming. *EURO Journal on Computational Optimization*, 8(2):173–199, 2020.

[31] S. Zhang and X. A. Sun. Stochastic dual dynamic programming for multi-stage stochastic mixed-integer nonlinear optimization. *Mathematical Programming*, 196:935–985, 2022.

[32] X. Zhu and M.G. Genton. Short-term wind speed forecasting for power system operations. *International Statistical Review*, 80(1):2–23, 2012.

[33] J. Zou, S. Ahmed, and X. A. Sun. Stochastic dual dynamic integer programming. *Mathematical Programming*, 175:461–502, 2019.

E.1 Supplementary material

The following electronic companion was submitted to *Operations Research* together with the main manuscript.

Electronic Companion

D Standard SDDP

We provide a brief presentation of SDDP in its standard form, and refer to [3] for more details. For standard SDDP, the assumption of stagewise independence is crucial.

Assumption 3. *The stochastic process $(\boldsymbol{\xi}_t)_{t\in[T]}$ which appears in the RHS of problem (MSLP) is stagewise independent, that is, for all $t \in [T]$:*

$$\xi_t = b_t(\xi_{[t-1]}, \eta_t) \equiv \eta_t.$$

Under this assumption, all expectations in (5) become unconditional, and $Q_t(\cdot,\cdot)$ and $\mathcal{Q}_t(\cdot)$ do no longer depend on $\xi_{[t-1]}$. With x_{t-1} being the only argument of $\mathcal{Q}_t(\cdot)$, convexity is assured, and thus linear cuts are sufficient for a tight and valid approximation. We can then reformulate subproblems (6) as linear programs

$$\underline{Q}_t(x_{t-1}, \xi_t^{(j)}) = \begin{cases} \min\limits_{x_t,\theta_{t+1}} & c_t^\top x_t + \theta_{t+1} \\ \text{s.t.} & W_t x_t = -T_{t-1}x_{t-1} + h_t(\xi_t) \\ & -\beta_{r,t+1}^\top x_t + \theta_{t+1} \geq \alpha_{r,t+1}, \quad r \in R_{t+1} \\ & x_t \geq 0 \end{cases} \tag{41}$$

using the relation

$$\mathfrak{Q}_{t+1}(x_t) = \min\left\{\theta_{t+1} \mid \theta_{t+1} \geq \beta_{r,t+1}^\top x_t + \alpha_{r,t+1},\ r \in R_{t+1}\right\}$$

and $|R_{t+1}| \in \mathbb{N}$. Here, R_{t+1} denotes the index set of cuts that have been constructed to approximate $\mathcal{Q}_t(\cdot)$ so far. Quantities $\beta_{r,t+1}$ and $\alpha_{r,t+1}$ are cut-specific parameters whose computation we explain in detail in the remainder of this section.

Note that in this setting, cuts are shared between scenarios naturally because, independent of the history $\xi_{[t-1]}$, all nodes in the scenario tree share the same expected value functions.

SDDP consists of two main steps in each iteration i, a *forward pass* and a *backward pass* through the stages $t \in [T]$. In the forward pass, a sequence of *trial points* $(x_t^i)_{t\in[T]}$ is generated, at which then new cuts are constructed in the following backward pass. By this alternating procedure, SDDP iteratively improves the cut approximations $\mathfrak{Q}_t(\cdot)$ of $\mathcal{Q}_t(\cdot)$. Alg. 1 provides a pseudo-code for standard SDDP.

We now provide a more detailed and technical look at the algorithmic steps.

Algorithm 1 SDDP

```
Input: Problem (MSLP) satisfying Assumptions 1, 2, 3. Bounds $\underline{\theta}_t, t = 2, \dots, T$. Stopping criterion.
 1: Initialize cut approximations with $\theta_t \geq \underline{\theta}_t$ for all $t = 2, \dots, T$.
 2: Initialize lower bound with $\underline{v}^0 = -\infty$.
 3: Set iteration counter to $i \leftarrow 0$.
 4: while Stopping criterion not satisfied do
 5:     Set $i \leftarrow i + 1$.
        Forward Pass
 6:     Solve the first-stage problem (42) to obtain trial point $x_1^i$.
 7:     for stages $t = 2, \dots, T$ do
 8:         Sample one realization $\xi_t^i$ from $\left\{\xi_t^{(1)}, \dots, \xi_t^{(q_t)}\right\}$.
 9:         Solve the stage-$t$ subproblem (41) associated with $\underline{Q}_t^i(x_{t-1}^i, \xi_t^i)$ to obtain trial point $x_t^i$.
10:     end for
        Backward Pass
11:     for stages $t = T, \dots, 2$ do
12:         for realizations $j = 1, \dots, q_t$ do
13:             Solve the updated stage-$t$ subproblem (41) associated with $\underline{Q}_t^{i+1}(x_{t-1}^i, \xi_t^{(j)})$. Store the optimal dual variables $\pi_{tj}^i$ and $\rho_{rtj}^i, r \in R_{t+1}$.
14:         end for
15:         Use relations (43), (44), (45) to create an optimality cut for $\mathcal{Q}_t(\cdot)$.
16:         Update the cut approximation $\mathfrak{Q}_t^i(\cdot)$ to $\mathfrak{Q}_t^{i+1}(\cdot)$ using relation (46).
17:     end for
18:     Solve the updated first-stage problem (42) to obtain a lower bound $\underline{v}^i$.
19: end while
Output: (Approximately) optimal feasible policy for (MSLP) defined by $x_1^i$ and $\mathfrak{Q}_t^i(\cdot), t = 2, \dots, T$.
```

D.1 Forward Pass

At the start of each iteration i, the first-stage subproblem

$$\begin{cases} \min\limits_{x_1} & c_1^\top x_1 + \mathfrak{Q}_2^i(x_1) \\ \text{s.t.} & W_1 x_1 = h_1 \\ & x_1 \geq 0 \end{cases} \tag{42}$$

is solved, which yields the trial point x_1^i.

Afterwards, for each stage $t = 2, \dots, T$ we iteratively draw one sample ξ_t^i from the set $\left\{\xi_t^{(1)}, \dots, \xi_t^{(q_t)}\right\}$ of all realizations using random sampling, and then solve the subproblem (41) associated with $\underline{Q}_t^i(x_{t-1}^i, \xi_t^i)$; this subproblem contains the cut approximation $\mathfrak{Q}_{t+1}^i(\cdot)$. This way, we obtain trial points x_t^i that can be used as a parameter in the following stage. Over all stages, this yields a sequence of trial points $(x_t^i)_{t \in [T]}$.

Remark D.1. *In standard SDDP it is also common to sample more than one realization per stage, meaning that we run several forward passes and obtain several sequences of* *trial points per iteration ([3]). All our results in this paper naturally extend to this case, but for notational convenience, we restrict to the case of one sample per iteration.*

Remark D.2. *By evaluating the trial points $(x_t^i)_{t\in[T]}$ in the objective $\sum_{t\in[T]} c_t(\xi_t^i)^\top x_t^i$ for the chosen sample path, we obtain a simulated objective value. This is not guaranteed to provide reasonable information on v^*, though, as it depends on only one specific sample.*

By repeating the same evaluation for a large number of sample paths and taking the mean, at least an unbiased estimator of an upper bound for v^* can be computed, a so-called statistical upper bound. *This kind of simulation is often executed after SDDP has terminated to assess the quality of the obtained policy. It can either be done using the same realizations of $\boldsymbol{\eta}_t$ as in SDDP (in-sample simulation) or with realizations that were not considered within the algorithm (out-of-sample simulation).*

D.2 Backward Pass

In the backward pass, new cuts for the expected value functions $\mathcal{Q}_t(\cdot)$ are generated at the trial points x_{t-1}^i, thus improving their approximation from $\mathfrak{Q}_t^i(\cdot)$ to $\mathfrak{Q}_t^{i+1}(\cdot)$.

The backward pass starts at stage T. Here, we consider the subproblems (41) for the trial point x_{T-1}^i computed in the forward pass, but all possible noise realizations $\xi_T^{(j)}, j = 1, \dots, q_T$. That is, we consider functions $\underline{Q}_T^{i+1}(x_{T-1}^i, \xi_T^{(j)})$ for all $j = 1, \dots, q_T$.

As $\mathfrak{Q}_{T+1}^{i+1}(\cdot) = \mathcal{Q}_{T+1}(\cdot) \equiv 0$, we have $\underline{Q}_T^{i+1}(\cdot, \xi_T^{(j)}) = Q_T(\cdot, \xi_T^{(j)})$. The LP dual to the subproblem (41) associated with $\underline{Q}_T^{i+1}(x_{T-1}^i, \xi_T^{(j)})$ is

$$\begin{cases} \min\limits_{x_T} & \pi_T^\top \big(- T_{T-1} x_{T-1}^i + \xi_T^{(j)} \big) \\ \text{s.t.} & W_T^\top \pi_T = c_T \\ & \pi_T \geq 0. \end{cases}$$

Let π_{Tj}^* be a dual optimal solution. Then, by strong duality

$$Q_T(x_{T-1}^i, \xi_T^{(j)}) = (\pi_{Tj}^*)^\top \big(- T_{T-1} x_{T-1}^i + \xi_T^{(j)} \big).$$

As the dual feasible set is independent of x_{T-1}, π_{Tj}^* remains feasible, although not necessarily optimal, for any $x_{T-1} \in \mathbb{R}^{n_{T-1}}$. Therefore, we obtain the cut

$$\begin{aligned} Q_T(x_{T-1}, \xi_T^{(j)}) &\geq (\pi_{Tj}^*)^\top \big(- T_{T-1} x_{T-1} + \xi_T^{(j)} \big) \\ &= \underbrace{-(\pi_{Tj}^*)^\top T_{T-1}}_{=:\beta_{Tj}^\top} x_{T-1} + \underbrace{(\pi_{Tj}^*)^\top \xi_T^{(j)}}_{=:\alpha_{Tj}} \\ &= \beta_{Tj}^\top x_{T-1} + \alpha_{Tj}. \end{aligned}$$

β_{Tj} is called *cut gradient* and α_{Tj} is called *cut intercept* or *constant.*

Executing this for all $j = 1, \dots, q_T$, and then taking expectations over cuts (D.2), we obtain a cut

$$\mathcal{Q}_T(x_{T-1}) \geq \beta_T^\top x_{T-1} + \alpha_T$$

for the expected value function $\mathcal{Q}_T(\cdot)$, with

$$\beta_T := \sum_{j=1}^{q_T} p_{Tj} \beta_{Tj}, \qquad \alpha_T := \sum_{j=1}^{q_T} p_{Tj} \alpha_{Tj}.$$

With this new cut, the cut approximation $\mathfrak{Q}_T^i(\cdot)$ is updated to

$$\mathfrak{Q}_T^{i+1}(x_{T-1}) = \max\left\{\mathfrak{Q}_T^i(x_{T-1}),\ \beta_T^\top x_{T-1} + \alpha_T\right\},$$

and thus the updated cut index set satisfies $R_T^{i+1} = \{1,\dots,i+1\} = R_T^i \cup \{i+1\}$.

In the same way, for stages $t = T-1,\dots,2$, cuts for $\mathcal{Q}_t(\cdot)$ can be constructed by solving subproblems (41) for the trial points x_{t-1}^i and all realizations $\xi_t^{(j)}, j = 1,\dots,q_t$. Importantly, by going backwards through the stages, at stage t we can already factor in the cuts that have been constructed at the following stage $t+1$, thus using a better approximation as the basis to construct a new cut. This means that we consider $\mathfrak{Q}_{t+1}^{i+1}(\cdot)$ and by that $\underline{Q}_t^{i+1}(\cdot,\cdot)$ with index $i+1$ in the backward pass of iteration i.

The objective function to the dual of the stage-t subproblem (41) associated with $\underline{Q}_t^{i+1}(x_{t-1}^i, \xi_t^{(j)})$ is

$$\pi_t^\top\big(-T_{t-1}x_{t-1}^i + \xi_t^{(j)}\big) + \sum_{r\in R_{t+1}^{i+1}} \rho_{rt}\alpha_{r,t+1}.$$

Let $(\pi_{tj}^*, \rho_{tj}^*)$ be a dual optimal solution. By strong duality, we obtain

$$\underline{Q}_t^{i+1}(x_{t-1}^i, \xi_t^{(j)}) = (\pi_{tj}^*)^\top\big(-T_{t-1}x_{t-1}^i + \xi_t^{(j)}\big) + \sum_{r\in R_{t+1}^{i+1}} \rho_{rtj}^*\alpha_{r,t+1}.$$

As for stage T, the dual feasible set is independent of x_{t-1}. Therefore, $(\pi_{tj}^*, \rho_{tj}^*)$ remains feasible, although not necessarily optimal, for any $x_{t-1} \in \mathbb{R}^{n_{t-1}}$. We obtain

$$\underline{Q}_t^{i+1}(x_{t-1}, \xi_t^{(j)}) \geq (\pi_{tj}^*)^\top\big(-T_{t-1}x_{t-1} + \xi_t^{(j)}\big) + \sum_{r\in R_{t+1}^{i+1}} \rho_{rtj}^*\alpha_{r,t+1}.$$

Finally, as $\underline{Q}_t^{i+1}(\cdot, \xi_t^{(j)})$ underestimates $Q_t(\cdot, \xi_t^{(j)})$, we obtain the cut

$$\begin{aligned} Q_t(x_{t-1}, \xi_t^{(j)}) &\geq (\pi_{tj}^*)^\top\big(-T_{t-1}x_{t-1} + \xi_t^{(j)}\big) + \sum_{r\in R_{t+1}^{i+1}} \rho_{rtj}^*\alpha_{r,t+1} \\ &= \underbrace{-(\pi_{tj}^*)^\top T_{t-1}}_{=:\beta_{tj}^\top} x_{t-1} + \underbrace{(\pi_{tj}^*)^\top \xi_t^{(j)} + \sum_{r\in R_{t+1}^{i+1}} \rho_{rtj}^*\alpha_{r,t+1}}_{=:\alpha_{tj}} \qquad (43) \\ &= \beta_{tj}^\top x_{t-1} + \alpha_{tj}. \end{aligned}$$

Executing this for all $j = 1,\dots,q_t$, and then taking expectations over cuts (43), we obtain a cut

$$\mathcal{Q}_t(x_{t-1}) \geq \beta_t^\top x_{t-1} + \alpha_t \qquad (44)$$

for the expected value function $\mathcal{Q}_t(\cdot)$, with

$$\beta_t := \sum_{j=1}^{q_t} p_{tj}\beta_{tj}, \qquad \alpha_t := \sum_{j=1}^{q_t} p_{tj}\alpha_{tj}. \qquad (45)$$

With this new cut, the cut approximation $\mathfrak{Q}_t^i(\cdot)$ is updated to

$$\mathfrak{Q}_t^{i+1}(x_{t-1}) = \max\left\{\mathfrak{Q}_t^i(x_{t-1}),\ \beta_t^\top x_{t-1} + \alpha_t\right\}. \qquad (46)$$

At the first stage, the updated first-stage subproblem (42) is solved with optimal value $\underline{v}^i$. As $\mathfrak{Q}_2^{i+1}(\cdot)$ is a lower approximation of $\mathcal{Q}_2(\cdot)$, $\underline{v}^i$ is a valid lower bound to the optimal value v^* of (MSLP). In contrast, we are not guaranteed to obtain a valid upper bound for v^* in SDDP, as we only consider one sampled scenario per iteration instead of evaluation over the entire scenario tree.

After each iteration of SDDP, one or several stopping criteria are checked, such as statistical stopping criteria, reaching a predefined number of iterations or stalling of the lower bounds $\underline{v}^i$ ([3]). If SDDP does not stop, a new iteration $i+1$ is started.

As mentioned before, the polyhedral dual feasible sets at each stage t are independent of x_{t-1}, thus they possess finitely many extreme points. Using this reasoning inductively starting at stage T and assuming that we restrict to optimal dual basic solutions in the cut generation process, it can be shown that only finitely many different cuts can be constructed. Additionally, the cuts are *tight*, *i.e.*, evaluating the RHS of (44) at x_{t-1}^i, yields $\underline{\mathcal{Q}}_t^{i+1}(x_{t-1}^i)$, which implies that the approximations eventually become exact. Combining this with the properties of the sampling procedure, almost sure finite convergence of SDDP to an optimal policy can be proven [10].

E Long-term Hydrothermal Scheduling Problem

The LTHS problem with 4 EERs and 95 generators that we consider in our computational experiments in Sect. 5 can be formulated as follows:

$$\begin{aligned}
\min \quad & \sum_{t=1}^{T} \beta^{t-1} \left(\sum_{k\in\mathcal{K}} \left(\sum_{j\in\mathcal{G}_k} c_{tj} g_{tj} \right) + \delta_{tk} r_{tk} \right) & \\
\text{s.t.} \quad & v_{tk} = v_{t-1,k} + a_{tk} + q_{tk} + s_{tk}, & t = 1,\dots,T, k\in\mathcal{K} \\
& q_{tk} + \sum_{j\in\mathcal{G}_k} g_j + r_{tk} + \sum_{\ell\in\mathcal{K}} (f_{t\ell k} - f_{tk\ell}) = d_{tk}, & t = 1,\dots,T, k\in\mathcal{K}, \\
& 0 \le v_{tk} \le \overline{v}_k, & t = 1,\dots,T, k\in\mathcal{K} \\
& 0 \le q_{tk} \le \overline{q}_k, & t = 1,\dots,T, k\in\mathcal{K} \\
& s_{t5} = 0, 0 \le s_{tk}, & t = 1,\dots,T, k\in\mathcal{K} \\
& \underline{g}_{tj} \le g_{tj} \le \overline{g}_{tj}, & t = 1,\dots,T, j\in\mathcal{G} \\
& 0 \le r_{tk} \le \overline{r}_{tk}, \underline{f}_{tk\ell} \le f_{tk\ell} \le \overline{f}_{tk\ell}, & t = 1,\dots,T, k\in\mathcal{K}, \ell\in\mathcal{K}.
\end{aligned}$$

Here $\mathcal{K}$ denotes the set of five sub-systems in the overall power system. The variables v_{tk}, a_{tk}, q_{tk} and s_{tk} denote the water level, inflow, hydro power generation and spillage for the reservoirs associated with systems $k\in\mathcal{K}$ at stage t. Note that only four of these systems contain a hydro reservoir, so for the fifth one the bounds $\overline{v}_k$ and $\overline{q}_k$ are zero, and the spillage variable is always set to zero as well. The first set of contraints describes the water balance of the hydro reservoirs. The initial levels v_{0k} are given, while the inflow vector a_t is considered uncertain and described by a PAR process.

The set $\mathcal{G}_k$ contains all the thermal generators in system $k\in\mathcal{K}$. The variables g_{tj} denote the thermal power generation of generator $j\in\mathcal{G}$ at stage t, with lower limits $\underline{g}_{tj}$ and upper limits $\overline{g}_{tj}$. The second set of constraints ensures the load balance for all systems $k\in\mathcal{K}$ at all stages t by satisfying the load d_{tk} using thermal generation, hydro power generation and energy exchange between systems which is represented by variables $f_{t\ell k}$ for $k,\ell\in\mathcal{K}$. Additionally, we may allow for a load deficit, represented by variable r_{tk}. Both types of variables are bounded as well.

In the objective function we minimize costs, which are discounted using the annual discount rate β. The costs include the generation costs by thermal generators, taken into account using specific costs c_{tj}, as well as the penalization of load curtailment, *i.e.*, a positive deficit r_{tk}. We assume that the latter costs are piecewise linear so that they increase with the amount of curtailment. This is modeled using the piecewise linear function δ_{tk}. It is assumed that hydro generation does not cause any costs.

F Inflow Modeling

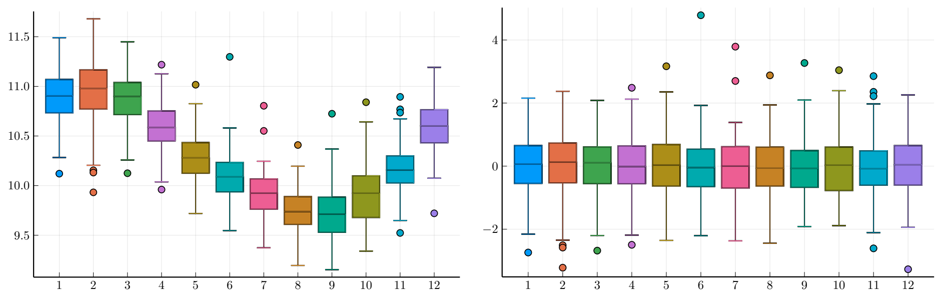
The following procedure is applied to all four EERs (SE, S, N, NE) independently, as we do not take spatial inflow dependencies into account. The performed model fitting and validation steps are standard in time series analysis as part of the Box-Jenkins method [1]. However, we follow an adapted variant for PAR models, which is mostly based on [6]. In that regard, our inflow modeling deviates from [12], where the same historical data is used, but the model selection (*i.e.*, identifying the lag order of the process) and model validation are executed for a single non-seasonal AR model and not specifically for a PAR model. For details, we refer to our GitHub project `https://github.com/ChrisFuelOR/LogLinearSDDP.jl`, which also contains the preparation of the inflow data.

F.1 Data Preparation

Let X_t denote the true historical inflow time series and let $Y_t = \log(X_t)$. The monthly box-plot diagram in Figure 1a shows that Y_t follows a seasonal, *i.e.*, monthly pattern. To account for this pattern, we perform a deseasonalization using monthly means μ_t and standard deviations σ_t, which yields a time series of the form

$$Z_t := \frac{Y_t - \mu_t}{\sigma_t}, \tag{47}$$

see also Remark 4.1 (e). The monthly means and standard deviations satisfy $\mu_t = \mu_{t+12}, \sigma_t = \sigma_{t+12}$. The box-plot diagram in Figure 1b for Z_t shows that the monthly pattern is removed.



(a) Before detrending. (b) After detrending.

Figure 1: Boxplot of Y_t before and after detrending for system SE.

F.2 Stationarity Analysis

To evaluate the time series Z_t for stationarity, we analyze its partial autocorrelation function (PACF) and perform an augmented Dickey-Fuller (ADF) test. The PACF shows a fast decline, which is an indicator for stationarity. The ADF test rejects the null hypothesis of a unit root at a 5% significance level, which again implies stationarity. Note that stationarity is a prerequisite for most of the following fitting and validation techniques.

F.3 Model Selection and Fitting

Next, we split the data into a training data set (from 1931 to 1994) and a test data set (from 1995 to 2009). The model selection, fitting and validation via hypothesis tests are only using the training data. Later on, we use the test data for out-of-sample validation.

As we consider a log-linear PAR model of type (22) to model X_t, the time series Y_t follows a standard PAR model. We assume a monthly resolution, so the coefficients of the AR model may vary with each month. This can be interpreted as fitting separate models for each month, only considering the corresponding lagged data, see [6, 8] for details. To identify a reasonable lag order for each of these models, for each month, we first estimate the *periodic* autocorrelation function (ACF) [6]. Additionally, for each month, we fit AR models with lag orders between 1 and 12 and use the Bayesian Information Criterion (BIC) to compare these models. The latter approach yields the following lag orders for the components of the model, summarized in Table 1.

Table 1: Lag orders identified using the BIC.

System	Month											
	1	2	3	4	5	6	7	8	9	10	11	12
SE	1	2	1	1	1	1	3	1	1	1	1	1
S	1	2	1	2	1	1	1	1	1	1	1	1
NE	1	2	1	1	1	3	10	1	1	1	2	1
N	1	4	1	1	1	10	3	4	5	1	1	1

In conformity with Sect. 5, we refer to the model using these lag orders as `LOG-BIC`. For comparison, we also consider a PAR model of lag order 1 for Y_t, which we refer to as `LOG-1`. Moreover, we consider a linear model that is obtained by linearizing the log-linear PAR model for X_t using a first-order Taylor approximation (cf. Sect. 4.2). Varying coefficients have been reported for this model and the given inflow data in the literature [2, 9, 11]. We therefore apply two different variants: For `LIN-SHA` we use the model coefficients as reported in [11], for `LIN-FIT` we use coefficients obtained using the same fitting and validation steps that we describe for the log-linear models below. We should mention that in contrast to our approach, `LIN-SHA` uses a multivariate normal distribution for the error terms.

Recall that we assume a constant lag order in our cut formulas, see Sect. 4, so the values in Table 1 imply the choice $p = 10$, whereas all other models satisfy $p = 1$.

F.4 Model Validation – Diagnostic

For validation of the monthly models constituting `LOG-BIC` (or `LOG-1`, `LIN-FIT`, respectively), we perform several diagnostic tests and analyses.

We observe that all monthly models exhibit a reasonable goodness of fit (adjusted $\mathbb{R}^2 > 0.35$) and autoregression coefficients that are significantly different from zero (as validated using a t-test for lag-order $p = 1$ or an F-test for lag orders $p > 1$, respectively).

We then analyze the residuals of the PAR model. First, we check for heteroscedasticity using a scatter plot of the residuals (this can be done on a monthly basis or for all residuals of the PAR model combined). The plots give no indication of variance changes in the residuals.

Second, we check the residuals for autcorrelation. To this end, we estimate the periodic residual autocorrelation function (RACF) [6]. For all months and all systems, we see no indication of significant autocorrelation. Still, we perform a Portmanteau test, as proposed in [6], to statistically assess the hypothesis of no autcorrelation. In most cases, this hypothesis cannot be rejected.

Finally, we check the residuals for normal distribution. Not only does the validity of the previous test results rely on normally distributed residuals, we also may assume the stagewise independent error term $\boldsymbol{\eta}_t$ appearing in formula (22) to be normally distributed in this case. For our analysis, we use histograms, qq-plots and different hypothesis tests (Jarque-Bera test, Anderson-Darling test, Kolmogorov-Smirnov test). It turns out that in many cases, the hypothesis of normality is debatable and can be rejected statistically. Importantly, we make the same observation when we fit the linearized model by Shapiro et al. [12] to the data (note that in [12] and the related technical report [11], model validation is only performed for the full time series, but not for the actual periodical model).

Since using normally distributed error terms $\boldsymbol{\eta}_t$ still yields reasonable forecasting and simulation results, see the following subsections, in our computational tests in Sect. 5, we nevertheless follow the normality hypothesis. In the future, a more elaborate PAR model may be applied to the LTHS problem.

F.5 Model Validation – Forecasting

To further validate our fitted models, we use them for forecasts and compare them to the historical data. More precisely, we use the fitted models (without error term) together with the historical data to compute one-step ahead forecasts for the training horizon (in-sample forecast) and the test horizon (out-of-sample forecast). For the obtained point forecasts we remove the deseasonalization and take the exponential function to obtain forecasts for the original time series X_t. Note that in general, simply taking the exponential function of forecasts for a logarithmized time series yields biased forecasts for the original time series, which can be avoided using a correction factor [5]. However, without this correction we observe better forecasting results. The mean absolute forecasting errors for the test set data are presented in Table 2. For comparison, also the errors for `LIN-FIT` and `LIN-SHA` are presented. As we can see, the linearized models show smaller one-step ahead forecasting errors than the log-linear ones, but relative to the mean value of the original time series, the differences are subtle.

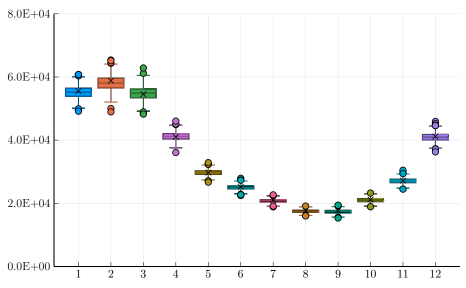
F.6 Model Validation – Simulation

As a last step, that is most relevant for our application of SDDP, we use the fitted models and $N(0, (\hat{\sigma}_t^e)^2)$-distributed errors $\boldsymbol{\eta}_t$ together with some historical starting values to generate 1000 alternative scenarios (sample paths) of 79 years length. Here, $(\hat{\sigma}_t^e)^2$ denotes the sample variance of the residuals. As before, we remove the deseasonalization

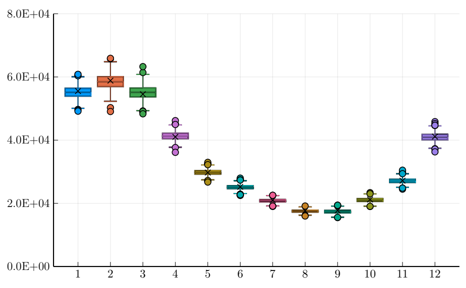
Table 2: One-step-ahead forecasting MAE (mean absolute error) on the test data.

	SE	S	NE	N
`LOG-BIC`	4888.3	4085.0	1309.7	978.1
`LOG-1`	4879.7	4106.3	1309.7	1009.5
`LIN-FIT`	4903.5	4047.8	1336.5	968.7
`LIN-SHA`	4853.3	3913.4	1392.2	958.2
Mean historical value	34805.9	10118.5	6496.1	6041.4

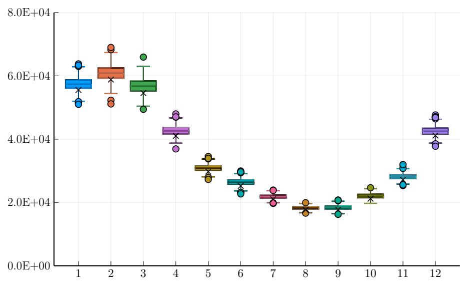
and take the exponential function for each value in the obtained time series. Then, we compare the statistical properties of the obtained series with the historical data. More precisely, we compare the monthly means and standard deviations. This is exemplified in Figures 2 to 5, which show box plots of the means and standard deviations for the systems SE and S. The historical monthly values are indicated by black crosses in each plot. Again, we consider all four different types of models.



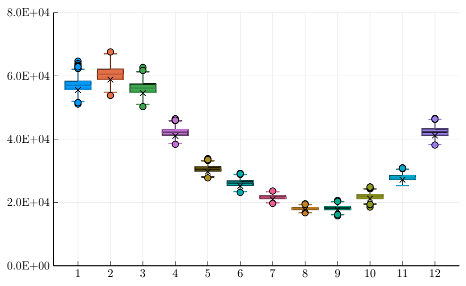
(a) `LOG-BIC`.



(b) `LOG-1`.



(c) `LIN-FIT`.

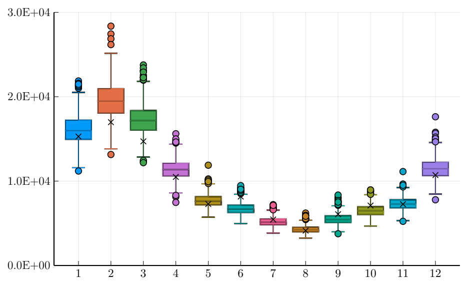


(d) `LIN-SHA`.

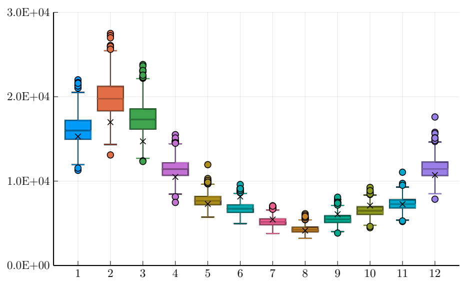
Figure 2: Boxplot of monthly means for system SE over 1000 sample paths.

For system SE, the simulated paths provide a good fit for the statistical properties observed in the historical data. The log-linear models seem to match the true monthly means slightly better. Especially in the first three months, the historical inflow means are consistently exceeded by the inflows from the linearized models, which is not the case for the log-linear models.

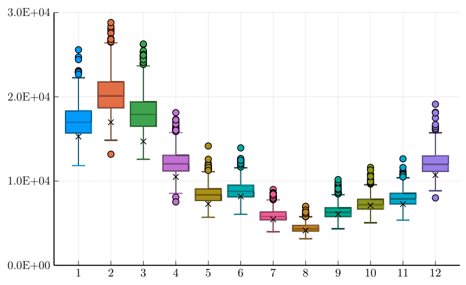
For system S, fitting seems most difficult among all four systems. Even though the linearized models provide a reasonable fit to the mean and standard deviation observed in the historical data, the historical values are again matched better using the log-linear



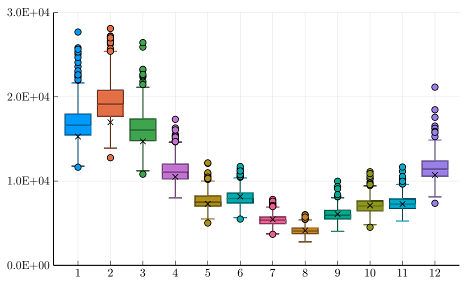
(a) `LOG-BIC`.



(b) `LOG-1`.

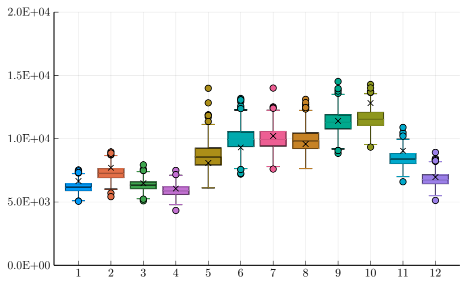


(c) `LIN-FIT`.

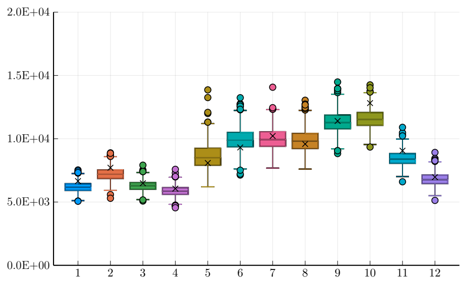


(d) `LIN-SHA`.

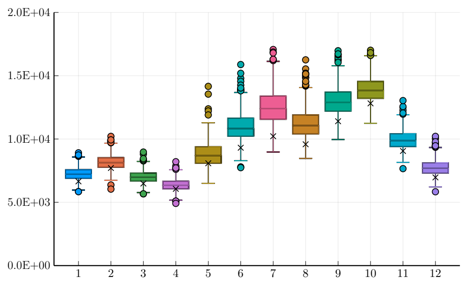
Figure 3: Boxplot of monthly standard deviations for system SE over 1000 sample paths.



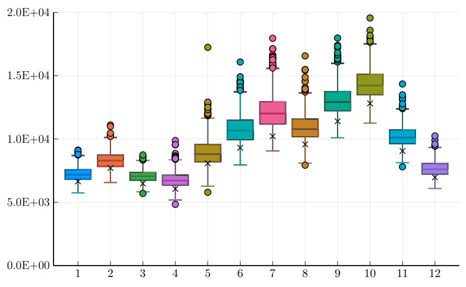
(a) `LOG-BIC`.



(b) `LOG-1`.

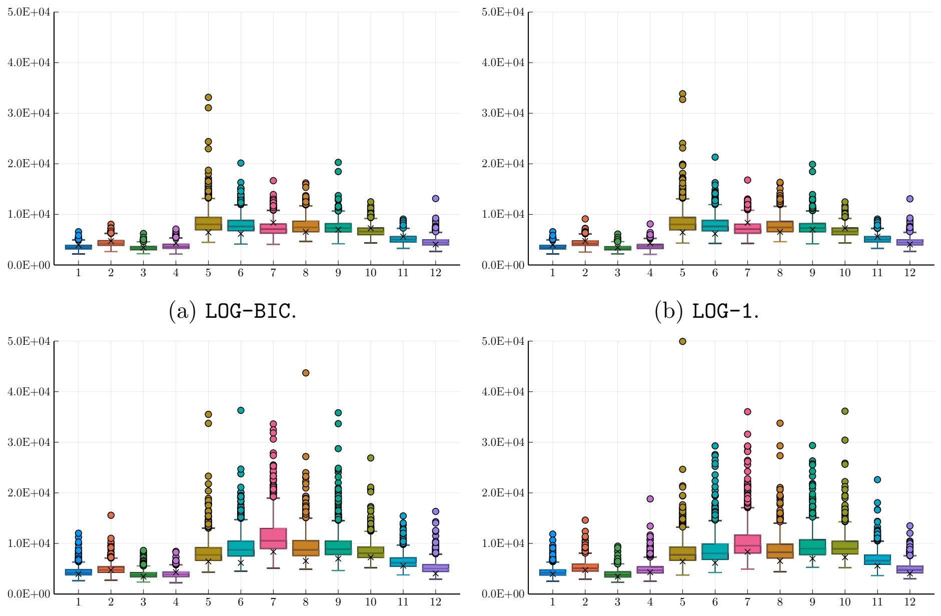


(c) `LIN-FIT`.



(d) `LIN-SHA`.

Figure 4: Boxplot of monthly means for system S over 1000 sample paths.



(a) `LOG-BIC`. (b) `LOG-1`.

(c) `LIN-FIT`. (d) `LIN-SHA`.

Figure 5: Boxplot of monthly standard deviations for system S over 1000 sample paths.

models. In particular, we do not observe the systematic underestimation of the historical mean that is apparent for the linearized case. Additionally, the log-linear models lead to less variability in the observed means and standard deviations over all sample paths.

In Tables 3 and 4 we provide some additional information on our simulation comparison for all four systems. Table 3 addresses the monthly averages and their deviation from the historical monthly averages. It presents the average of these deviations. As from the boxplots, we can see that the log-linear models on average provide a more accurate fit for the historical time series, except for system NE where the deviations are similar.

Table 4 presents similar results, but for deviations from the monthly averages obtained using `LOG-BIC`. Hence, it gives a hint on the relative differences in the inflow levels between the log-linear and the linearized models. We observe a considerable difference in the average inflow level, especially for systems SE and S. As SE is by far the reservoir with the largest capacity, we conclude that the linearized models yield significantly larger *total* inflows as well.

Table 3: Averages of deviation (in %) of monthly averages from historical monthly averages over 1000 sample paths.

	SE	S	NE	N
`LOG-BIC`	-0.6	-1.8	5.6	1.6
`LOG-1`	-0.4	-2.1	5.6	1.8
`LIN-FIT`	3.8	11.4	5.2	4.7
`LIN-SHA`	2.9	3.2	5.9	4.1

Table 4: Averages of deviation (in %) of monthly averages from `LOG-BIC` monthly averages over 1000 sample paths.

	SE	S	NE	N
`LOG-1`	0.2	-0.3	0.0	0.2
`LIN-FIT`	4.5	13.6	-0.4	3.1
`LIN-SHA`	3.5	5.3	0.3	2.4

Importantly, these difference in the inflow levels, as well as the accuracy of simulating inflows with similar statistical properties to the historical data, do still occur if we use the exact same parameters in the log-linear and linearized models (instead of those obtained in the fitting process). These differences seem to be directly linked to the difference between the log-linear model formulas (24) and their Taylor approximations. We should note that similar observations have been already made by Löhndorf and Shapiro in [9].

G Additional Computational Results

G.1 Standard Runs

Table 5 summarizes the computational requirements of different steps in SDDP.

Table 5: Average time requirements in computational tests (in % of total computation time of SDDP).

Algorithmic step	`LOG-BIC`	`LOG-1`
Set-up log-linear AR process initially	0	0
Compute exponent factors $\Theta(t,\tau,\ell,m,k)$ in (29)	0	0
Compute cut intercept factors (27)	38	44
thereof: Compute pre-factor $\left(\sum_{r\in R_{t+1}} \rho^*_{rtj}\big(\alpha^{(\tau)}_{r,t+1,\ell}\big)\right)$	32	37
Compute deterministic part of intercept	6	1
Evaluate cut intercepts in (30)	36	34
Solver calls	11	13
Other steps	9	8

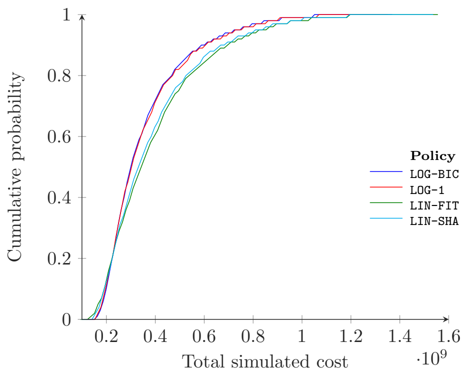
Fig. 6 illustrates the cumulative distribution of the obtained statistical upper bounds during the out-of-sample simulations for the LTHS problem.

Fig. 7 present different statistics for the temporal development of the reservoir filling levels observed in the out-of-sample simulations.

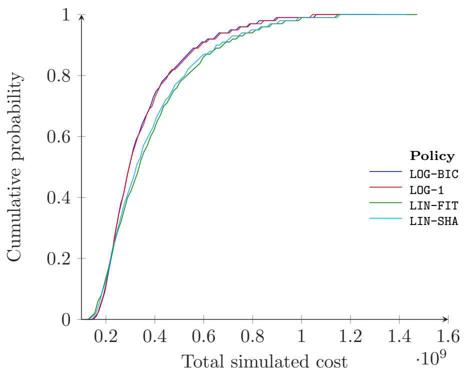
Fig. 8 presents the average costs caused by a supply deficit compared to the demand in the out-of-sample simulations.

G.2 Experiments with Scaled Inflows

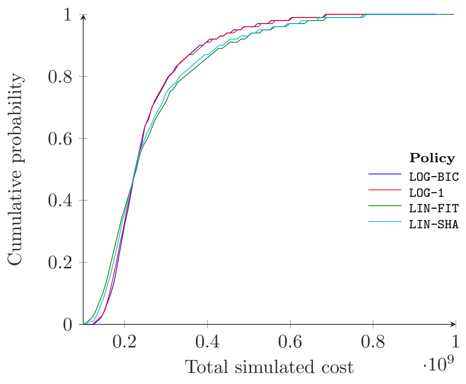
Our presented results seem heavily affected by the difference of inflows levels of the log-linear and linearized models. To get a better comparison between standard SDDP and our proposed version, we run a second batch of experiments. In this batch, even if contradicting our fitting results, we scale the inflows from the log-linear models by



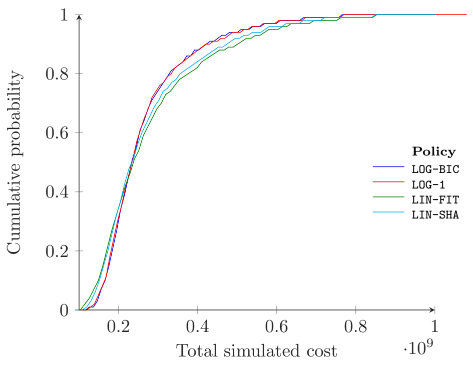
(a) `LOG-BIC` inflow data.



(b) `LOG-1` inflow data.

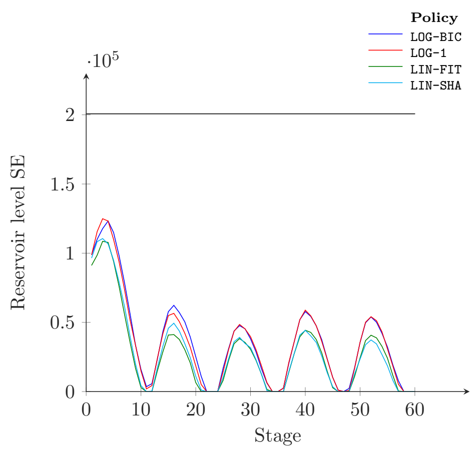


(c) `LIN-FIT` inflow data.

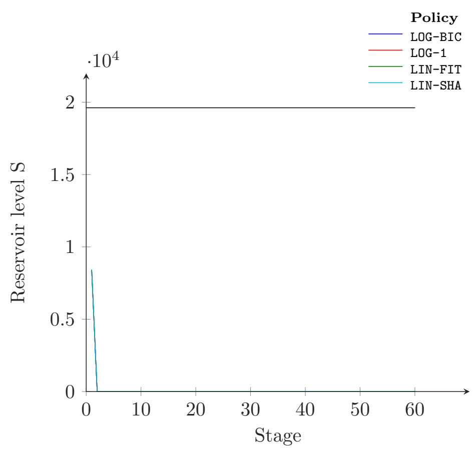


(d) `LIN-SHA` inflow data.

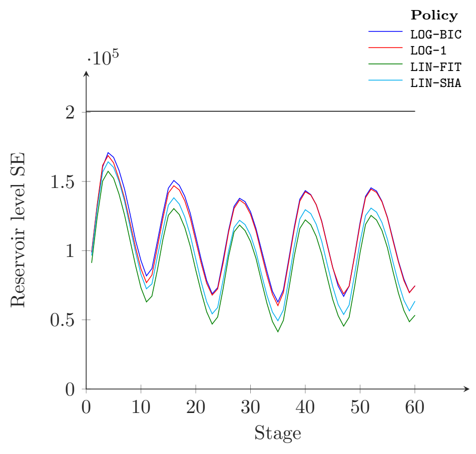
Figure 6: Cumulative distribution of upper bounds in out-of-sample simulations.



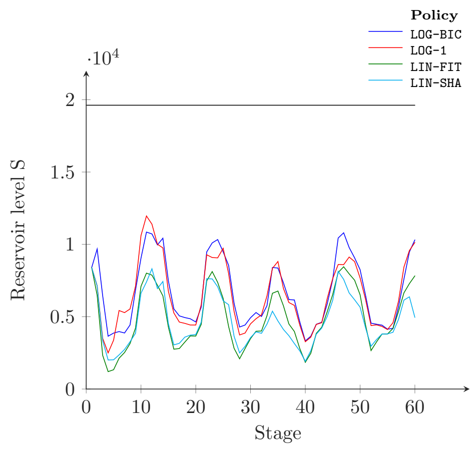
(a) System SE - 5% quantile.



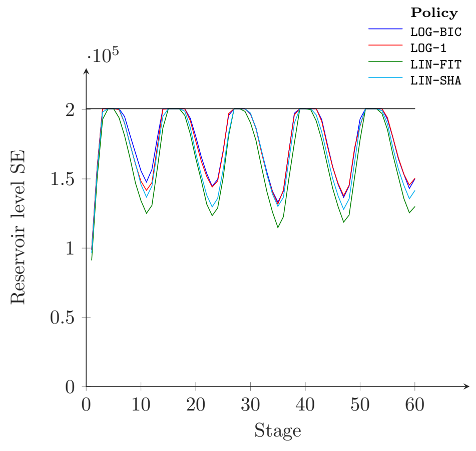
(b) System S - 5% quantile.



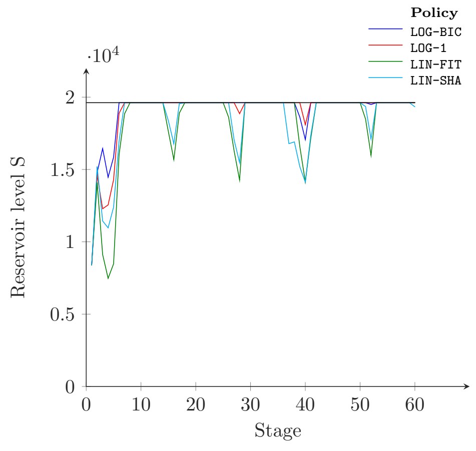
(c) System SE - average.



(d) System S - average.

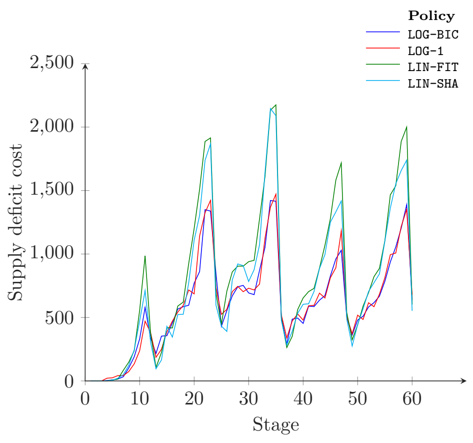


(e) System SE - 95% quantile.

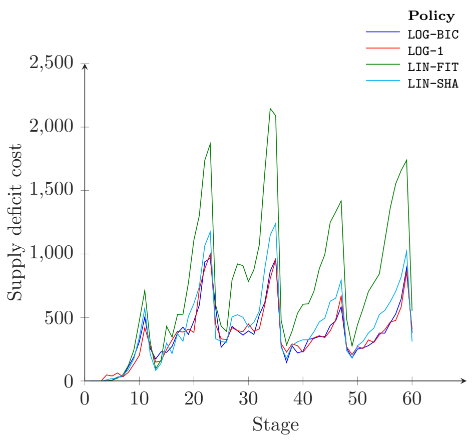


(f) System S - 95% quantile.

Figure 7: Statistics of reservoir volumes for systems SE and S over the first 60 stages for 2000 out-of-sample simulations using `LOG-1` inflow data.



(a) Using LOG-1 data.



(b) Using LOG-1 data.

Figure 8: Average supply deficit costs over the first 60 stages for 2000 out-of-sample simulations.

1.04, lifting them to a similar level as those from the linear models. In this case, the differences in the out-of-sample performance are far less pronounced, as illustrated in Fig. 9.

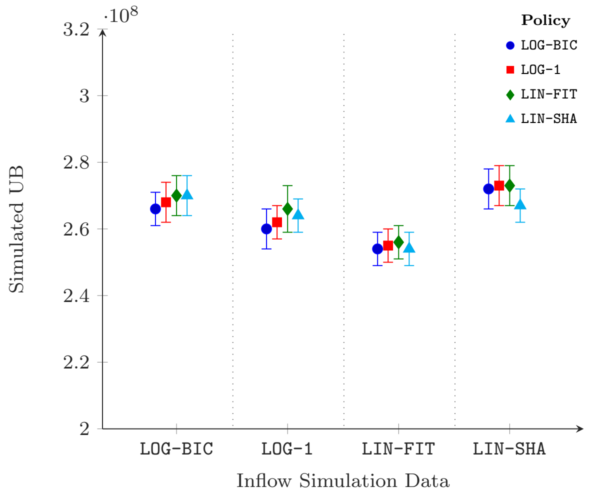


Figure 9: Average statistical upper bounds (and confidence intervals) in out-of-sample simulations over 5 independent runs with scaled inflows for log-linear models.

H Cut Formulas under Linearization Breakpoint

In Sect. 4.4, we detected that using our proposed non-convex cuts in SDDP may come with a substantial computational overhead. On the other hand, it allows for a more accurate representation of uncertainty in situations that warrant a log-linear PAR process.

In this section, we show that our proposed method can be straightforwardly extended to a combined approach where the uncertainty in earlier stages is modeled with more accuracy and complexity, while using a more granular model for later stages. More precisely, assume that we approximate the nonlinear function $b_t(\cdot,\cdot)$ defining the AR process $(\boldsymbol{\xi}_t)_{t\in[T]}$ by a linear function $\widehat{b}_t(\cdot)$ (see the idea by Shapiro et al. [12] described in Sect. 4.2) for later stages $\widehat{T}+1,\dots,T$, thus computing linear cuts for the expected value functions of these stages, while only considering the more accurate nonlinear function $b_t(\cdot,\cdot)$ for stages $1,\dots,\widehat{T}$, with $\widehat{T}\in\mathbb{N}, \widehat{T}\in[1,T]$. We call $\widehat{T}$ the *breakpoint stage*. If we choose $\widehat{T}=T$, then we consider the cases discussed in Sect. 4, whereas for $\widehat{T}=1$, only linear cuts are constructed following the idea from [12] (recall that ξ_1 is assumed deterministic and that no cuts are computed for stage 1). For $\widehat{T}\in(1,T)$, we obtain a trade-off between model accuracy and computational efficiency. This trade-off can be used to reduce the computational burden associated with the generation and evaluation of nonlinear cuts.

We now analyze how the incorporation of a breakpoint stage $\widehat{T}\in(1,T)$ affects the cut formulas for the nonlinear cuts. For simplicity, we restrict to the lag-one process (26) in this context, however, the same ideas can be applied to processes (24) and (25).

We assume that the linear cuts generated in stages $t=\widehat{T}+1,\dots,T$ have the form

$$\mathcal{Q}_t(x_{t-1},\xi_{t-1}) \geq \beta_t^\top x_{t-1} + \mu_t^\top \xi_{t-1} + w_t \tag{48}$$

with μ_t the cut gradient for ξ_{t-1} and w_t the scenario-independent cut intercept (see [4, 7], Sect. D and Sect. 3.1).

For stages $t=\widehat{T}-1,\dots,2$, let the cut coefficients be defined as in (35)-(36). Additionally, we define stage-$\widehat{T}$ factors

$$\begin{aligned} \alpha_{\widehat{T}j}^{(\widehat{T})} &:= \Big(\pi_{\widehat{T}j}^* + \sum_{r\in R_{t+1}} \rho_{r\widehat{T}j}\mu_{r,\widehat{T}+1}\Big) \odot e^{\gamma_{\widehat{T}}} \odot e^{\psi_{T\ell}\eta_{\widehat{T}}^{(j)}}, \\ \alpha_{\widehat{T}}^{(\widehat{T})} &:= \sum_{j=1}^{q_t} p_{tj}\alpha_{\widehat{T}j}^{(\widehat{T})} \end{aligned} \tag{49}$$

and for all $t=2,\dots,\widehat{T}$ the intercepts

$$\begin{aligned} w_{tj} &:= \sum_{r\in R_{t+1}} \rho_{rtj}^* w_{r,t+1}, \\ w_t &:= \sum_{j=1}^{q_t} p_{tj} w_{tj}. \end{aligned} \tag{50}$$

Then, we obtain the following cut result.

Theorem H.1. *Suppose that the RHS uncertainty in problem (MSLP) is defined by process (26) for stages $2,\dots,\widehat{T}$ and by a linear approximation $\widehat{b}_t(\cdot)$ for stages $\widehat{T}+1,\dots,T$. For any $t=2,\dots,\widehat{T}$ and any state (x_{t-1},ξ_{t-1}), a nonlinear cut for the expected value function $\mathcal{Q}_t(\cdot,\cdot)$ is then given by*

$$\mathcal{Q}_t(x_{t-1},\xi_{t-1}) \geq \beta_t^\top x_{t-1} + \sum_{\tau=t}^{\widehat{T}} \big(\alpha_{tj}^{(\tau)}\big)^\top \xi_{t-1}^{\prod_{k=t}^{\tau}\phi_k} + w_t, \tag{51}$$

with $\alpha_t^{(\tau)}$ defined as in (35)-(36) for stages $\widehat{T}-1,\dots,2$ and as in (49) for stage $\widehat{T}$, and with w_t defined as in (50).

Proof. We consider an arbitrary backward pass in SDDP and assume that for stages $t = \widehat{T}+1,\dots,T$ already R_{t+1} linear cuts of form (48) have been generated.

For some fixed state $(\bar{x}_{t-1},\bar{\xi}_{t-1})$ and some arbitrary realization $\eta_t^{(j)}, j \in \{1,\dots,q_t\}$, the dual objective function to the stage-$\widehat{T}$ subproblem is

$$\pi_{\widehat{T}}^\top\Big(b_{\widehat{T}}(\bar{\xi}_{\widehat{T}-1},\eta_{\widehat{T}}) - T_{\widehat{T}-1}\bar{x}_{\widehat{T}-1}\Big) + \sum_{r\in R_{\widehat{T}+1}} \rho_{r\widehat{T}}\Big(\mu_{r,\widehat{T}+1}^\top b_{\widehat{T}}(\bar{\xi}_{\widehat{T}-1},\eta_{\widehat{T}}) + w_{r,\widehat{T}+1}\Big).$$

Let a dual optimal solution be given by $\pi^*_{\widehat{T}j}$ and $\rho^*_{r\widehat{T}j}, r \in R_{\widehat{T}+1}$. Then, using the same reasoning as in the previous proofs, we obtain

$$\begin{aligned}
&\widetilde{Q}_{\widehat{T}}\big((x_{\widehat{T}-1},\xi_{\widehat{T}-1}),\eta_{\widehat{T}}^{(j)}\big)\\
&\geq (\pi^*_{\widehat{T}j})^\top\Big(b_{\widehat{T}}(\xi_{\widehat{T}-1},\eta_{\widehat{T}}) - T_{\widehat{T}-1}x_{\widehat{T}-1}\Big) + \sum_{r\in R_{\widehat{T}+1}} \rho^*_{r\widehat{T}j}\Big(\mu_{r,\widehat{T}+1}^\top b_{\widehat{T}}(\xi_{\widehat{T}-1},\eta_{\widehat{T}}) + w_{r,\widehat{T}+1}\Big)\\
&= \underbrace{-(\pi^*_{\widehat{T}j})^\top T_{\widehat{T}-1}}_{\overset{(35)}{=}\beta_{\widehat{T}j}^\top} x_{\widehat{T}-1} + (\pi^*_{\widehat{T}j})^\top\Big(e^{\gamma_{\widehat{T}}} \odot e^{\psi_{\widehat{T}}\eta_{\widehat{T}}^{(j)}} \odot \xi_{\widehat{T}-1}^{\phi_{\widehat{T}}}\Big)\\
&\quad + \sum_{r\in R_{\widehat{T}+1}} \rho^*_{r\widehat{T}j}\Big(\mu_{r,\widehat{T}+1}^\top\Big(e^{\gamma_{\widehat{T}}} \odot e^{\psi_{\widehat{T}}\eta_{\widehat{T}}^{(j)}} \odot \xi_{\widehat{T}-1}^{\phi_{\widehat{T}}}\Big) + w_{r,\widehat{T}+1}\Big)\\
&= \beta_{\widehat{T}j}^\top x_{\widehat{T}-1} + \underbrace{\sum_{r\in R_{\widehat{T}+1}} \rho^*_{r\widehat{T}j} w_{r,\widehat{T}+1}}_{\overset{(50)}{=} w_{\widehat{T}j}} + \Big(\pi^*_{\widehat{T}j} + \sum_{r\in R_{\widehat{T}+1}} \rho^*_{r\widehat{T}j}\mu_{r,\widehat{T}+1}\Big)^\top\Big(e^{\gamma_{\widehat{T}}} \odot e^{\psi_{\widehat{T}}\eta_{\widehat{T}}^{(j)}} \odot \xi_{\widehat{T}-1}^{\phi_{\widehat{T}}}\Big)\\
&= \beta_{\widehat{T}j}^\top x_{\widehat{T}-1} + w_{\widehat{T}j} + \Bigg(\underbrace{\Big(\pi^*_{\widehat{T}j} + \sum_{r\in R_{\widehat{T}+1}} \rho^*_{r\widehat{T}j}\mu_{r,\widehat{T}+1}\Big) \odot e^{\gamma_{\widehat{T}}} \odot e^{\psi_{\widehat{T}}\eta_{\widehat{T}}^{(j)}}}_{\overset{(49)}{=}\alpha_{\widehat{T}j}^{(\widehat{T})}}\Bigg)^\top \xi_{\widehat{T}-1}^{\phi_{\widehat{T}}}\\
&= \beta_{\widehat{T}j}^\top x_{\widehat{T}-1} + w_{\widehat{T}j} + \big(\alpha_{\widehat{T}j}^{(\widehat{T})}\big)^\top \xi_{\widehat{T}-1}^{\phi_{\widehat{T}}}
\end{aligned}$$

Taking expectations over all $j = 1,\dots,q_{\widehat{T}}$, we obtain

$$\mathcal{Q}_{\widehat{T}}(x_{\widehat{T}-1},\xi_{\widehat{T}-1}) \geq \beta_{\widehat{T}}^\top x_{\widehat{T}-1} + w_{\widehat{T}} + \big(\alpha_{\widehat{T}}^{(\widehat{T})}\big)^\top \xi_{\widehat{T}-1}^{\phi_{\widehat{T}}}$$

Except for the additional scenario-independent constant $w_{\widehat{T}}$, this looks exactly like a cut (37) for stage T had we not used a breakpoint stage. Following the inductive proof of Theorem 4.2, for all the earlier stages $t = 2,\dots,\widehat{T}-1$, the cuts can also be expressed by formula (37), but with $\widehat{T}$ in the role of T and by adding the constant w_t as defined in (50). □

References

[1] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Time series analysis: forecasting and control.* Wiley, Hoboken, New Jersey, fifth edition, 2016.

[2] L. Ding, S. Ahmed, and A. Shapiro. A Python package for multi-stage stochastic programming. Preprint, available online at `http://www.optimization-online.org/DB_FILE/2019/05/7199.pdf`, 2019.

[3] C. Füllner and S. Rebennack. Stochastic dual dynamic programming and its variants – A review. Preprint, available at `http://www.optimization-online.org/DB_FILE/2021/01/8217.pdf`, 2023.

[4] V. Guigues. SDDP for some interstage dependent risk-averse problems and application to hydro-thermal planning. *Computational Optimization and Applications*, 57:167–203, 2014.

[5] D. N. Gujarati and D. C. Porter. *Basic econometrics.* McGraw-Hill, New York, NY, fifth edition, 2009.

[6] K. W. Hipel and A. I. McLeod. *Time series modelling of water resources and environmental systems.* 2006. available online at `https://www.eng.uwaterloo.ca/~kwhipel/Time%20Series%20Book.htm`.

[7] G. Infanger and D. P. Morton. Cut sharing for multistage stochastic linear programs with interstage dependency. *Mathematical Programming*, 75:241–256, 1996.

[8] T. Lohmann, A. S. Hering, and S. Rebennack. Spatio-temporal hydro forecasting of multireservoir inflows for hydro-thermal scheduling. *European Journal of Operational Research*, 255(1):243–258, 2016.

[9] N. Löhndorf and A. Shapiro. Modeling time-dependent randomness in stochastic dual dynamic programming. *European Journal of Operational Research*, 273(2):650–661, 2019.

[10] A.B. Philpott and Z. Guan. On the convergence of stochastic dual dynamic programming and related methods. *Operations Research Letters*, 36(4):450–455, 2008.

[11] A. Shapiro and W. Tekaya. Report for technical cooperation between Georgia Institute of Technology and ONS – Operador Nacional do Sistema Eletrico – risk averse approach. Technical report, 2011.

[12] A. Shapiro, W. Tekaya, J. P. da Costa, and M. P. Soares. Risk neutral and risk averse stochastic dual dynamic programming method. *European Journal of Operational Research*, 224(2):375–391, 2013.