



Globally optimal univariate spline approximations

Robert Mohr¹ · Maximilian Coblenz²  · Peter Kirst³

Received: 11 October 2022 / Accepted: 7 February 2023
© The Author(s) 2023

Abstract

We revisit the problem of computing optimal spline approximations for univariate least-squares splines from a combinatorial optimization perspective. In contrast to most approaches from the literature we aim at globally optimal coefficients as well as a globally optimal placement of a fixed number of knots for a discrete variant of this problem. To achieve this, two different possibilities are developed. The first approach that we present is the formulation of the problem as a mixed-integer quadratically constrained problem, which can be solved using commercial optimization solvers. The second method that we propose is a branch-and-bound algorithm tailored specifically to the combinatorial formulation. We compare our algorithmic approaches empirically on both, real and synthetic curve fitting data sets from the literature. The numerical experiments show that our approach to tackle the least-squares spline approximation problem with free knots is able to compute solutions to problems of realistic sizes within reasonable computing times.

Keywords Spline approximation · Least-squares spline approximation · Branch-and-bound · Global optimization · Curve fitting

✉ Maximilian Coblenz
maximilian.coblenz@hwg-lu.de

Robert Mohr
robert.mohr@web.de

Peter Kirst
peter.kirst@wur.nl

- ¹ Institute for Operations Research (IOR), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
- ² Department of Services and Consulting, Ludwigshafen University of Business and Society, Ludwigshafen, Germany
- ³ Operations Research and Logistics (ORL), Wageningen University and Research (WUR), Wageningen, The Netherlands

1 Introduction

In the univariate data approximation problem we are given n data points $(x_i, y_i) \in \mathbb{R}^2$ and a family of functions $h(\cdot, w) : \mathbb{R} \rightarrow \mathbb{R}$ parametrized with a vector $w \in \mathbb{R}^m$. The goal is to find a parameter \bar{w} such that $h(x_i, \bar{w}) \approx y_i$ for all $i \in \{1, \dots, n\}$. Thus, one aims to compute ideally a global minimizer of the finite-sum problem

$$\min_{w \in \mathbb{R}^m} \sum_{i=1}^n \ell(h(x_i, w), y_i),$$

where $\ell : \mathbb{R}^2 \rightarrow \mathbb{R}$ is an error function that quantifies the discrepancy between $h(x_i, w)$ and the actual value y_i corresponding to x_i .

In many applications it is clear from the context which family of functions $h(\cdot, w)$ should be used and the parameters w might even have a physical meaning. However, there are also many cases where no specific family of functions $h(\cdot, w)$ is dictated by the application. In those cases spline functions are a popular tool for data approximation, see [9].

A spline function consists of several polynomial functions of degree $m \in \mathbb{N}$ each defined on a segment of the approximation interval $[x_1, x_m]$. The joint points of those segments are called knots. Depending on the application, the polynomials may have to satisfy certain continuity restrictions at the knots where they join, e.g., continuity and smoothness up to the second derivative.

Another important choice concerns the type of error function. In this article, we focus on the least-squares criterion, i.e., $\ell(z, y) = (z - y)^2$, which is arguably the most widely-used error function in practice. Depending on the application, other choices can be more appropriate, e.g., $\ell(z, y) = |z - y|$. However, these alternative error functions and their implications for the solution of the spline approximation problem are for the most part beyond the scope of this paper.

The earliest algorithmic approach to determine an optimal knot placement with respect to the least-square error for a fixed given number of knots is a discrete Newton method proposed in [6], which can be used to approximate locally optimal solutions. Other local approaches include the use of Gauss-Newton-type methods, see, e.g., [11, 17, 30], as well as the Fletcher-Reeves nonlinear conjugate gradient (FR) method, see [9]. However, by applying these approaches it is only possible to determine locally optimal solutions, and commonly their quality is highly dependent on the initial solution provided to the algorithm.

In contrast, genetic algorithms as described, for instance, in [23, 27, 32], can escape low-quality locally optimal solutions, but the solution that is returned by these algorithms might be neither locally nor globally optimal, since these approaches are purely heuristic and cannot provide any certificate of optimality.

In [1] the so-called cutting angle method is used in order to compute globally optimal knot placements of the least-squares spline approximation problem with free knots. To the best of our knowledge, it is the only deterministic global optimization approach that was proposed specifically for the solution of such a problem. Furthermore, knots are treated as continuous variables here, which allows for great flexibility and makes

the problem even more challenging. However, as noted in the section on numerical experiments in [1], the computing time is prohibitive even for small problems.

In contrast to the methods reviewed so far, there exist many approaches in the literature which neither seek a locally nor a globally optimal solution with respect to the least-squares criterion. Instead, it is proposed to determine a knot placement that satisfies other quality measuring criteria. Some suggest looking at the data and place the knots according to some rule of thumbs, such as placing them near points of inflection or at specific quantiles, see [29, 36]. Other approaches are based on forward knot addition and/or backward knot deletion, see, e.g. [31, 34]. Moreover, Bayesian approaches are proposed in [8, 10], which employ a continuous random search methodology via reversible-jump Markov chain Monte Carlo methods. A more detailed review and comparison of these approaches can be found in [19, 35].

In this article, we propose to solve a combinatorial formulation of the least-squares spline approximation problem with free knots and show how this problem can be solved to global optimality in a reasonable amount of time. We assume the number k of free knots to be fixed and are only concerned with the optimal placement of the knots. The first approach that we present is the formulation of the combinatorial least-squares spline approximation problem as a mixed-integer quadratically constrained problem (MIQCP). Problems of this type can be solved using commercial optimization solvers such as Gurobi or CPLEX. Our second method is a branch-and-bound algorithm tailored specifically to the combinatorial least-squares spline approximation problem with free knots. Branch-and-bound methods have been successfully applied to a range of statistical problems like variable selection and clustering, see, for example, [3, 15].

This article is structured as follows. In the next section we formally introduce the relevant optimization problems. Moreover, we show that local optimization algorithms cannot be expected to yield satisfactory solutions for this problem if the initial point is not chosen sufficiently close to a globally optimal solution. However, since in typical applications, neither the dimension of the decision variable nor the number of data points is particularly large, it is possible to make use of the specific problem structure in order to devise algorithmic approaches to approximate the globally optimal solution of problem instances of relevant sizes. We suggest placing knots always exactly in the middle between two data points. Note, however, that there is no universal approach for this choice in the literature. In Sect. 3, as a first algorithmic approach for the solution of the latter, we present a convex mixed-integer formulation that can be solved using commercial optimization solvers. As an alternative algorithmic approach, we propose a new branch-and-bound method in Sect. 4, which is tailored specifically to the combinatorial formulation. In Sect. 5 we present numerical experiments on real and synthetic data which show that the combinatorial approach to the least-squares spline approximation problem with free knots makes it possible to compute high-quality solutions to problems of realistic sizes within reasonable computing times. Section 6 concludes the paper with some final remarks. The supplementary information of this article comprise a discussion on alternative error functions (Section A), a list of the used test functions (Section B), and more detailed numerical results (Sections C, D, E).

2 Least-squares spline regression

In this section, we start by formulating the least-squares spline approximation problem with fixed knots and describe how to efficiently compute its optimal solutions. Based on this we extend our consideration to problems with free knots and explain why the computation of globally optimal solutions is challenging. Finally, we describe how the least-squares spline approximation problem with free knots can be reformulated as a combinatorial optimization problem.

2.1 Least-squares spline regression with fixed knots

In order to simplify the exposition, we only consider cubic spline functions in this article, i.e., $m = 3$. The formal definition of a cubic spline function with k knots is

$$s(x, \beta^{(0)}, \dots, \beta^{(k)}, \xi) := \begin{cases} p(x, \beta^{(0)}), & \text{for } x_1 \leq x \leq \xi_1 \\ p(x, \beta^{(1)}), & \text{for } \xi_1 < x \leq \xi_2, \\ \vdots & \\ p(x, \beta^{(k)}), & \text{for } \xi_k < x \leq x_n, \end{cases}$$

with the ordered knot vector $\xi \in \mathbb{R}^k$, $x_1 < \xi_1 < \xi_2 < \dots < \xi_k < x_n$, and $k + 1$ cubic polynomials

$$p(x, \beta^{(j)}) := \beta_0^{(j)} + \beta_1^{(j)}x + \beta_2^{(j)}x^2 + \beta_3^{(j)}x^3$$

for $j \in \{0, \dots, k\}$. Note that we limit our exposition to cubic splines with the maximum number of three continuity restrictions (up to the second derivative) only for ease of presentation. It is straightforward to generalize our approach to piecewise polynomials of arbitrary degree and different continuity restrictions.

We first assume that the knot vector $\xi \in \mathbb{R}^k$ is fixed and the goal is to determine parameters optimal with respect to the least-squares criterion, i.e., we aim to solve the optimization problem

$$\min_{\beta \in \mathbb{R}^{4(k+1)}} \sum_{i=1}^n (s(x_i, \beta, \xi) - y_i)^2, \quad (1)$$

where $\beta \in \mathbb{R}^{4(k+1)}$ denotes the concatenation of the vectors $\beta^{(0)}, \dots, \beta^{(k)} \in \mathbb{R}^4$. Since the spline function s is linear in the decision variable β , the problem is a linear least-squares problem whose optimal solutions can be computed by solving a system of linear equations. This even holds in presence of constraints that ensure smoothness and continuity of the computed function s . The system of linear equations stems directly from the Karush-Kuhn-Tucker conditions. For more details and a derivation of this we refer to [11, 13, 28]. In this case, note that β can be directly obtained as soon as the knots are fixed. We will use this observation again in problem (3) in the next section.

A remark on our use of the power functions-based spline representation is due. In the spline literature there is also the B-spline representation, see, e.g., [7], which transforms the least-squares spline approximation problem into a different function space. Essentially, this reduces the number of parameters and offers a computational performance gain. Mathematically, both formulations are equivalent. However, we opted for the power functions-based spline representation for several reasons. Firstly, we think that this representation offers an easier approach to the problem from an optimization perspective. Secondly, the mixed-integer formulation in Sect. 3 can be straightforwardly derived if the power functions-based spline representation is used, and it is not clear whether an equally good mixed-integer formulation could be derived with a B-spline representation. Lastly, the computational gains from using the B-spline representation when computing the optimal knot placement with the branch-and-bound method we present in Sect. 4 turned out to be negligible.

2.2 Least-squares spline regression with free knots

In contrast to the aforementioned description, if the knot vector $\xi \in \mathbb{R}^k$ is not fixed but enters the optimization problem as a decision variable, the problem is referred to as the least squares spline approximation problem with free knots, which reads

$$\begin{aligned}
 \min_{\beta, \xi} \quad & \sum_{x_i \leq \xi_1} \left(p(x_i, \beta^{(0)}) - y_i \right)^2 \\
 & + \sum_{j=1}^{k-1} \sum_{\xi_j < x_i \leq \xi_{j+1}} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \\
 & + \sum_{x_i > \xi_k} \left(p(x_i, \beta^{(k)}) - y_i \right)^2 \\
 \text{s.t.} \quad & p(\xi_j, \beta^{(j-1)}) = p(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & p'(\xi_j, \beta^{(j-1)}) = p'(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & p''(\xi_j, \beta^{(j-1)}) = p''(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & \xi_j \leq \xi_{j+1}, \quad j = 1, \dots, k - 1, \\
 & x_1 \leq \xi_1, \quad \xi_k \leq x_n, \\
 & \beta \in \mathbb{R}^{4(k+1)}, \quad \xi \in \mathbb{R}^k.
 \end{aligned} \tag{2}$$

Although, as already mentioned, it is not uncommon to restrict possible knot placements to a finite number of possibilities, in this section we prefer to consider the purely continuous version of this problem in order to illustrate some interesting aspects. Furthermore, it is important to note that the number of knots k is still a fixed parameter, despite the fact that the knot vector ξ enters the problem as a decision variable. Moreover, since the equality constraints are nonlinear in the decision variables ξ_j , problem (2) is, in fact, nonconvex. For this reason, there are typically many locally minimal

solutions that prevent local optimization algorithms from converging to globally optimal solutions.

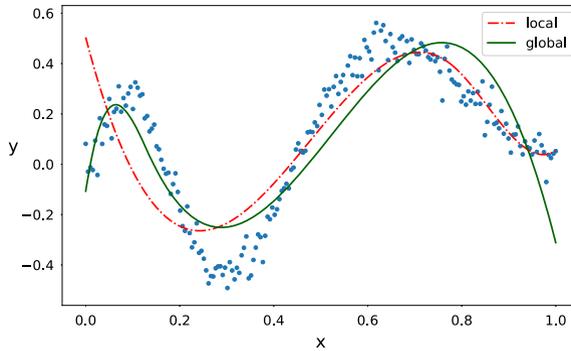
In order to visualize this, we reformulate the problem such that the vector ξ is the only decision variable. As discussed in the previous section, this is possible since we can obtain the optimal parameters $\beta(\xi)$ of a spline function with a fixed knot vector ξ by solving a linear system of equations, which is uniquely solvable under mild conditions. More details on this can be found in [11, 13, 28]. With this notation we rewrite problem (2) equivalently as

$$\begin{aligned}
 \min_{\xi} \quad & \sum_{x_i \leq \xi_1} \left(p(x_i, \beta^{(0)}(\xi)) - y_i \right)^2 \\
 & + \sum_{j=1}^{k-1} \sum_{\xi_j < x_i \leq \xi_{j+1}} \left(p(x_i, \beta^{(j)}(\xi)) - y_i \right)^2 \\
 & + \sum_{x_i > \xi_k} \left(p(x_i, \beta^{(k)}(\xi)) - y_i \right)^2 \\
 \text{s.t.} \quad & \xi_j \leq \xi_{j+1}, \quad j = 1, \dots, k-1, \\
 & x_1 \leq \xi_1, \quad \xi_k \leq x_n \\
 & \xi \in \mathbb{R}^k.
 \end{aligned} \tag{3}$$

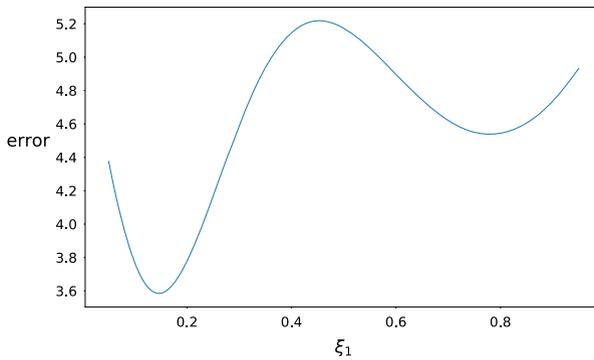
Note that the nonlinear equality constraints are eliminated from the problem since they are automatically satisfied due to the choice of the optimal parameter vector $\beta(\xi)$. However, the problem is still nonconvex since $\beta(\cdot)$ is nonlinear in the knot vector ξ .

Suboptimal local solutions of problem (3) are already present when the dimension k of the knot vector is small. We illustrate this for $k = 1$ and $k = 2$. The blue dots in Fig. 1a show a synthetic data set with 200 data points for which a spline approximation is computed. The objective function of problem (3) for this case and one free knot ξ_1 is illustrated in Fig. 1b. We observe that there exists a global and a local minimum. Moreover, a local optimization algorithm initialized with a random knot placement might approximate either one of these two locally optimal points. The spline functions corresponding to these solutions are shown in Fig. 1a in red and green. Based on these plots one could argue that the spline function corresponding to the globally optimal solution approximates the data set more accurately. Furthermore, it becomes apparent that a cubic spline function with one free knot does not yield a satisfying approximation of this data set at all.

The objective function of problem (3) for the same data set and two free knots ξ_1 and ξ_2 is depicted in Fig. 2. The function has one globally minimal point $\tilde{\xi} = (0.2632, 0.5166)^\top$ and two locally minimal points $\tilde{\xi} = (0.0939, 0.1003)^\top$ and $\tilde{\xi} = (0.8191, 0.8422)^\top$. Moreover, the spline functions corresponding to the local solutions approximate the data set significantly worse than the spline function corresponding to the globally optimal knot placement, as can be observed in Fig. 3.



(a) Locally (red) and globally (green) optimal spline functions with one knot.



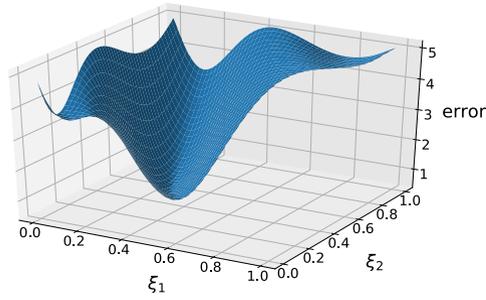
(b) Objective function of problem (3) for one free knot.

Fig. 1 Locally and globally optimal least-squares spline functions and the corresponding objective function of problem (3) with one free knot

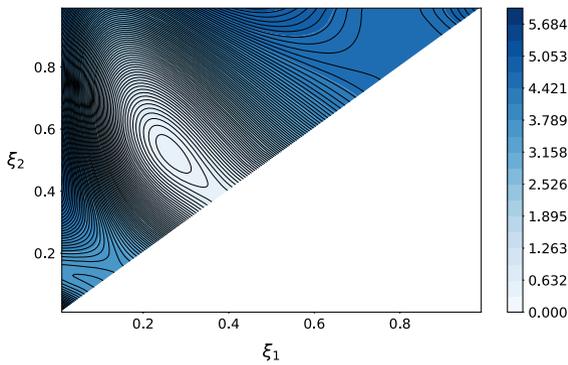
This demonstrates that local optimization algorithms are only useful for the solution of the least-squares spline approximation problem with free knots if they start from an initial solution sufficiently close to a globally optimal solution. Based on a set partitioning reformulation in the following subsection, we propose two novel possibilities in Sects. 3 and 4 to accomplish this.

2.3 Reformulation as a set partitioning problem

Since least-squares spline regression with free knots is an extraordinarily hard problem, we reformulate it as a type of set partitioning problem. To achieve this, we now restrict the possible knot locations to a finite set as already discussed. It will become clear that our new solution methods allow to do this in a rather natural way as we shall see in the following. Furthermore, globally optimal solutions of this combinatorial formulation may serve as promising initial points for local optimization algorithms and it might even be possible to further improve these approximations by relaxing the original problem.



(a) Objective function for two free knots.



(b) Contour plot of objective function for two free knots.

Fig. 2 Objective function of problem (3) with two free knots

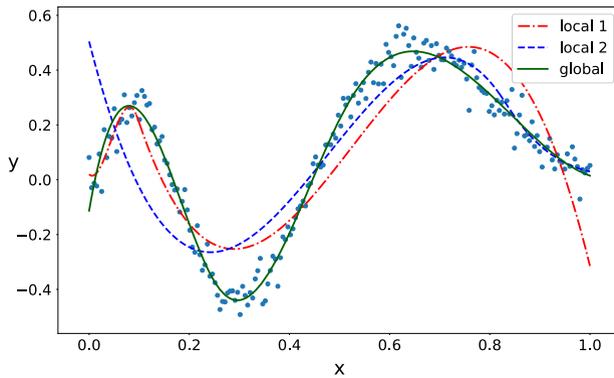


Fig. 3 Least-squares splines corresponding to the two locally minimal knot placements and the least-squares spline corresponding to the globally optimal knot placement $\hat{\xi}$. The red and blue curves correspond to the knot placements $\hat{\xi}_1$ and $\hat{\xi}_2$, respectively (Color figure online)

Since we want to approximate a given dataset with a spline with k knots, every data point needs to be assigned to exactly one of the $k + 1$ polynomials. Therefore, instead of computing the optimal placement of the knots, we aim to find an optimal partition $\{I_j \mid j = 0, \dots, k\}$ of the set of indices $\{1, \dots, n\}$.

Due to the assumed ordering of the design points $x_1 < \dots < x_n$, a partition is only feasible if for all $i, j \in \{0, \dots, k\}$ with $i < j$ we also have that $r < q$ for all $r \in I_i$ and $q \in I_j$. Since the spline function corresponding to a feasible partition $\{I_j \mid j = 0, \dots, k\}$ has to satisfy continuity conditions at the knots, the exact location of these knots has to be fixed based solely on the partition. Clearly, the knot ξ_j has to be placed between the values $\max_{i \in I_{j-1}} x_i$ and $\min_{i \in I_j} x_i$ for all $j = 1, \dots, k$. We propose to select the midpoint of these two values, i.e., we set $\xi_j = (\max_{i \in I_{j-1}} x_i + \min_{i \in I_j} x_i)/2$ for all $j = 1, \dots, k$. This simple and symmetric choice worked well in our numerical experiments. Thus, in order to determine the optimal feasible partition, we solve the combinatorial optimization problem

$$\begin{aligned}
 & \min_{\beta, I_0, \dots, I_k} \sum_{j=0}^k \sum_{i \in I_j} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \\
 & \text{s.t. } \bigcup_{j=0}^k I_j = \{1, \dots, n\}, \\
 & \quad I_j \neq \emptyset, \quad j = 0, \dots, k, \\
 & \quad s < i \text{ for all } s \in I_l \text{ and } i \in I_j \text{ with } l < j, \\
 & \quad p(\xi_j, \beta^{(j-1)}) = p(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & \quad p'(\xi_j, \beta^{(j-1)}) = p'(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & \quad p''(\xi_j, \beta^{(j-1)}) = p''(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & \quad \beta \in \mathbb{R}^{4(k+1)},
 \end{aligned} \tag{4}$$

where

$$\xi_j = \left(\max_{i \in I_{j-1}} x_i + \min_{i \in I_j} x_i \right) / 2,$$

for all $j \in \{1, \dots, k\}$.

The total number of feasible set partitions in problem (4) is $\binom{n-1}{k}$. Note that we do not allow empty index sets in our feasible partitions since basically this reduces the number of knots of the spline function.

Furthermore, let us stress that if the goal is to compute a piecewise polynomial, i.e., a spline function without any continuity restrictions, only the first three constraints in problem (4) are needed and the resulting combinatorial problem is an equivalent reformulation of the least-squares piecewise polynomial approximation problem with free knots.

3 Formulation as a convex mixed-integer quadratically constrained problem

A first possibility towards the solution of the combinatorial problem (4) is to reformulate it as a type of mixed-integer optimization problem that can be solved using commercial optimization solvers such as Gurobi or CPLEX. As will be explained in detail in the rest of this section, problem (4) can be equivalently reformulated as the following convex mixed-integer quadratically constrained problem (MIQCP):

$$\begin{aligned}
 & \min_{\alpha, \beta, z, q, w} \sum_{i=1}^n \alpha_i \\
 \text{s.t.} & [z_{ij} = 1] \implies [q_{ij} = p(x_i, \beta^{(j)}) - y_i], \forall (i, j) \in J_1 \times J_2, \\
 & q_{ij}^2 \leq \alpha_i, \forall (i, j) \in J_1 \times J_2, \\
 & \sum_{j=0}^k z_{ij} = 1, \forall i \in J_1, \\
 & \sum_{i=1}^n z_{ij} \geq 1, \forall j \in J_2, \\
 & \sum_{q=i+1}^n \sum_{r=0}^{j-1} (1 - z_{qr}) \geq (n - i) \cdot j \cdot z_{ij}, \forall i \in J_1 \setminus \{n\}, j \in J_2 \setminus \{0\}, \\
 & w_{ij} \leq z_{ij}, \forall (i, j) \in \bar{J}, \\
 & w_{ij} \leq z_{i+1, j+1}, \forall (i, j) \in \bar{J}, \\
 & w_{ij} \geq z_{ij} + z_{i+1, j+1} - 1, \forall (i, j) \in \bar{J}, \\
 & [w_{ij} = 1] \implies [p(\gamma_i, \beta^{(j)}) - p(\gamma_i, \beta^{(j+1)}) = 0], \forall (i, j) \in \bar{J}, \\
 & [w_{ij} = 1] \implies [p'(\gamma_i, \beta^{(j)}) - p'(\gamma_i, \beta^{(j+1)}) = 0], \forall (i, j) \in \bar{J}, \\
 & [w_{ij} = 1] \implies [p''(\gamma_i, \beta^{(j)}) - p''(\gamma_i, \beta^{(j+1)}) = 0], \forall (i, j) \in \bar{J}, \\
 & \alpha \in \mathbb{R}^n, \beta \in \mathbb{R}^{4(k+1)}, z \in \{0, 1\}^{n(k+1)}, q \in \mathbb{R}^{n(k+1)}, w \in \{0, 1\}^{(n-1)k}
 \end{aligned} \tag{5}$$

where $J_1 := \{1, \dots, n\}$, $J_2 := \{0, \dots, k\}$, $\bar{J} := (J_1 \setminus \{n\}) \times (J_2 \setminus \{k\})$ and $\gamma_i := (x_i + x_{i+1})/2$, for all $i \in J_1 \setminus \{n\}$. In total, the problem contains $(n + 4) \cdot (k + 1) + n$ continuous variables, $2nk + n - k$ binary variables and $9n(k + 1)$ constraints. Among those constraints are $n(k + 1)$ convex quadratic inequality constraints and $4n(k + 1)$ so-called indicator constraints. This type of constraints have the general structure

$$[c = \delta] \implies a^T x \leq b, \tag{6}$$

where $x \in \mathbb{R}^n$ and $c \in \{0, 1\}$ are variables whereas $\delta \in \{0, 1\}$, $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are fixed parameters. The interpretation is that the inequality is only enforced if the condition $c = \delta$ is true. Otherwise, the inequality may be ignored by the solver.

Indicator constraints are an alternative to big-M formulations and can be handled directly by commercial optimization solvers such as Gurobi or CPLEX [4, 14]. The big-M formulation corresponding to (6) that could be used instead reads

$$d \cdot M \leq a^T x - b \leq d \cdot M, \tag{7}$$

where $M > 0$ is sufficiently large and d is a binary variable that satisfies

$$d = \begin{cases} 0, & \text{if } c = \delta, \\ 1, & \text{otherwise.} \end{cases}$$

In general, the existence of a sufficiently large value M , such that (6) can be equivalently replaced with (7), is dependent on the objective function as well as other constraints of the optimization problem. If M is chosen too small, the optimal solution might be excluded from the problem. On the other hand, large big-M constants can lead to numerical difficulties and excessively long run times due to weak relaxations in branch-and-cut algorithms. In contrast, indicator constraints are handled internally by the solvers and usually do not lead to numerical difficulties. However, they can lead to longer run times compared to well chosen big-M constants in some cases [4]. In Sect. 3.2, we will discuss whether a big-M formulation could be an appropriate alternative to the indicator constraints in problem (5).

The constraints in problem (5) can be grouped into three blocks. The first two constraints are needed to shift the least-squares objective function of the aforementioned problems into the constraints, thus, leading to a linear objective function. The next three constraints ensure that the partition of the data points encoded in the variables z_{ij} is feasible, and the remaining constraints enforce the continuity restrictions at the knots. In the following subsections we explain each of these blocks in detail.

3.1 Feasible partitions

For every index tuple $(i, j) \in J_1 \times J_2$ the binary variable z_{ij} encodes whether the i th design point x_i is assigned to the j th polynomial, or, equivalently, if the index i is contained in the index set I_j , i.e., for all $(i, j) \in J_1 \times J_2$ it holds

$$z_{ij} = \begin{cases} 1, & \text{if } i \in I_j, \\ 0, & \text{if } i \notin I_j. \end{cases} \tag{8}$$

The constraints

$$\sum_{j=0}^k z_{ij} = 1, \quad \forall i \in J_1, \tag{9}$$

$$\sum_{i=1}^n z_{ij} \geq 1, \quad \forall j \in J_2, \tag{10}$$

$$\sum_{q=i+1}^n \sum_{r=0}^{j-1} (1 - z_{qr}) \geq (n - i) \cdot j \cdot z_{ij}, \quad \forall i \in J_1 \setminus \{n\}, j \in J_2 \setminus \{0\}, \quad (11)$$

ensure that each design point is assigned to exactly one polynomial, and that at least one design point is assigned to each polynomial. Moreover, constraint (11) states that a design point x_i can only be assigned to the j th polynomial if all the following design points are not assigned to prior polynomials.

Constraints (9), (10) and (11) represent

$$\bigcup_{j=0}^k I_j = \{1, \dots, n\}, \quad (12)$$

$$I_j \neq \emptyset, \quad j = 0, \dots, k, \quad (13)$$

$$s < i \text{ for all } s \in I_l \text{ and } i \in I_j \text{ with } l < j, \quad (14)$$

in problem (4). Clearly, conditions (10) and (13) are equivalent. The equivalence of the remaining conditions is shown in the following lemma.

Lemma 3.1 *If the binary vector $z \in \{0, 1\}^{n(k+1)}$ is defined as in (8), then conditions (9) and (11) are equivalent to conditions (12) and (14).*

Proof We first show that if conditions (12) and (14) are satisfied, then conditions (9) and (11) are satisfied as well. Condition (12) implies that $\sum_{j=0}^k z_{ij} \geq 1$ holds for all $i \in J_1$ and since condition (14) implies that every index is assigned to at most one index set, i.e., $\sum_{j=0}^k z_{ij} \leq 1$ for all $i \in J_1$, we arrive at condition (9).

Next, we choose some $i \in J_1 \setminus \{n\}$ and $j \in J_2 \setminus \{0\}$. If $z_{ij} = 0$ then the inequality in condition (11) is satisfied trivially. On the other hand, if $z_{ij} = 1$, then $i \in I_j$ holds. We define the set $K := \{i + 1, \dots, n\} \times \{0, \dots, j - 1\}$ and suppose there exist $(q, r) \in K$ with $z_{qr} = 1$. This would imply $q \in I_r, r < j$ and $q > i$, in contradiction to condition (14). Thus, $z_{qr} = 0$ and $(1 - z_{qr}) = 1$ holds for all $(q, r) \in K$. Consequently, we obtain

$$\sum_{q=i+1}^n \sum_{r=0}^{j-1} (1 - z_{qr}) = \sum_{(q,r) \in K} (1 - z_{qr}) = |K| = (n - i) \cdot j = (n - i) \cdot j \cdot z_{ij}$$

and condition (11) is satisfied.

For the other direction, we assume that condition (12) or (14) is violated and show that this implies that condition (9) or (11) is violated. If condition (12) is violated, then there exists an index $i \in \{1, \dots, n\}$ which is not assigned to any index set. Therefore, we have $\sum_{j=0}^k z_{ij} = 0$ and condition (9) is violated. On the other hand, if (14) is violated, then there exist $l, j \in \{1, \dots, k\}$ with $l < j$ and indices $s \in I_l$ and $i \in I_j$ such that $s \geq i$. Consequently, we have that $z_{sl} = 1$ and $z_{ij} = 1$. Thus, either condition (9) is violated, or it follows that $s \neq i$ and therefore $s > i$ must be true. Now, since

$(s, l) \in K$ and $z_{sl} = 1$ we obtain the equality

$$\sum_{q=i+1}^n \sum_{r=0}^{j-1} (1 - z_{qr}) = |K \setminus \{(s, l)\}| = |K| - 1 = (n - i) \cdot j - 1.$$

However, if condition (11) was satisfied, the inequality

$$\sum_{q=i+1}^n \sum_{r=0}^{j-1} (1 - z_{qr}) \geq (n - i) \cdot j$$

would have to be satisfied as well. Since this is not the case, condition (11) is violated. □

To summarize the above, a feasible binary vector z in problem (5) corresponds to a feasible partition of the data points, in the sense that conditions (12) to (14) are satisfied.

3.2 Continuity restrictions

The interpretation of the variable w_{ij} is that $w_{ij} = 1$ holds if and only if $i \in I_j$ and $i + 1 \in I_{j+1}$, i.e., two successive indices are assigned to different (successive) index sets. In other words, $w_{ij} = 1$ holds if and only if there is a knot between the data points x_i and x_{i+1} . Consequently, the constraints

$$[w_{ij} = 1] \implies [p(\gamma_i, \beta^{(j)}) - p(\gamma_i, \beta^{(j+1)}) = 0], \forall (i, j) \in \bar{J}, \tag{15}$$

$$[w_{ij} = 1] \implies [p'(\gamma_i, \beta^{(j)}) - p'(\gamma_i, \beta^{(j+1)}) = 0], \forall (i, j) \in \bar{J}, \tag{16}$$

$$[w_{ij} = 1] \implies [p''(\gamma_i, \beta^{(j)}) - p''(\gamma_i, \beta^{(j+1)}) = 0], \forall (i, j) \in \bar{J} \tag{17}$$

from problem (5) enforce the continuity conditions at a value $\gamma_i = (x_i + x_{i+1})/2$ only if x_i and x_{i+1} belong to different polynomials. Note that, as in problem (4), knots are midpoints between two consecutive design points that are assigned to different polynomials.

An intuitive way to model the variable w is via the nonlinear constraints

$$w_{ij} = z_{ij} \cdot z_{i+1, j+1}, \forall (i, j) \in \bar{J}. \tag{18}$$

Fortunately, non-linear constraints of this type can be replaced by linear ones due to the fact that w and z both are binary decision variables in problem (5). Thus, (18) is rewritten equivalently by

$$\begin{aligned} w_{ij} &\leq z_{ij}, \forall (i, j) \in \bar{J}, \\ w_{ij} &\leq z_{i+1, j+1}, \forall (i, j) \in \bar{J}, \\ w_{ij} &\geq z_{ij} + z_{i+1, j+1} - 1, \forall (i, j) \in \bar{J}. \end{aligned}$$

As mentioned previously, an alternative to model restrictions of type (15) to (17) is the widely used big- M reformulation. However, in this case it is not clear how to obtain a reasonable value for the constant M without knowledge of the optimal solution. The numerical tests that we conducted showed that constants M that are large enough so that the optimal solution is not excluded from the feasible set often lead to numerical difficulties and long run times. In contrast to that, indicator constraints neither increased the computation time nor lead to numerical issues on the problem instances that we tested. Moreover, the aforementioned state-of-the-art solvers allow to incorporate this type of constraints conveniently.

3.3 Generalized epigraph reformulation

In this subsection we explain how the nonlinear objective function of problem (4) can be rewritten by a linear one in problem (5). This reformulation technique is sometimes called epigraph reformulation in the literature, see, e.g., [33]. To this end, first of all note that the objective function of problem (4) can be rewritten as

$$\sum_{j=0}^k \sum_{i \in I_j} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 = \sum_{i=0}^n \sum_{j=0}^k z_{ij} \left(p(x_i, \beta^{(j)}) - y_i \right)^2.$$

In combination with the results from the previous subsections, it is clear that problem (4) is equivalent to

$$\begin{aligned} & \min_{\beta, z, w} \sum_{i=0}^n \sum_{j=0}^k z_{ij} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \\ \text{s.t. } & \sum_{j=0}^k z_{ij} = 1, \quad \forall i \in J_1, \\ & \sum_{i=1}^n z_{ij} \geq 1, \quad \forall j \in J_2, \\ & \sum_{q=i+1}^n \sum_{r=0}^{j-1} (1 - z_{qr}) \geq (n - i) \cdot j \cdot z_{ij}, \quad \forall i \in J_1 \setminus \{n\}, j \in J_2 \setminus \{0\}, \quad (19) \\ & w_{ij} \leq z_{ij}, \quad \forall (i, j) \in \bar{J}, \\ & w_{ij} \leq z_{i+1, j+1}, \quad \forall (i, j) \in \bar{J}, \\ & w_{ij} \geq z_{ij} + z_{i+1, j+1} - 1, \quad \forall (i, j) \in \bar{J}, \\ & [w_{ij} = 1] \implies [p(\gamma_i, \beta^{(j)}) - p(\gamma_i, \beta^{(j+1)}) = 0], \quad \forall (i, j) \in \bar{J}, \\ & [w_{ij} = 1] \implies [p'(\gamma_i, \beta^{(j)}) - p'(\gamma_i, \beta^{(j+1)}) = 0], \quad \forall (i, j) \in \bar{J}, \\ & [w_{ij} = 1] \implies [p''(\gamma_i, \beta^{(j)}) - p''(\gamma_i, \beta^{(j+1)}) = 0], \quad \forall (i, j) \in \bar{J}, \\ & \beta \in \mathbb{R}^{4(k+1)}, z \in \{0, 1\}^{n(k+1)}, w \in \{0, 1\}^{(n-1)k}. \end{aligned}$$

This objective function, however, has a difficult nonlinear and nonconvex structure. In order to improve upon this formulation, we introduce the additional decision variables $\alpha \in \mathbb{R}^n$, replace the objective function with $\sum_{i=0}^n \alpha_i$ and add the constraints

$$\sum_{j=0}^k z_{ij} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \leq \alpha_i, \quad \forall i \in J_1. \tag{20}$$

Due to the first constraint in problem (19), we have that for each $i \in J_1$ there only exists a single index $j \in J_2$ such that $z_{ij} = 1$. Thus, the constraints (20) are equivalent to the quadratic indicator constraints

$$[z_{ij} = 1] \implies \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \leq \alpha_i, \quad \forall i \in J_1.$$

Since the quadratic functions are convex in the decision variables, the resulting problem is a convex MIQCP. However, so far common optimization solvers can only handle linear indicator constraints [4, 14]. By introducing an additional decision variable $q \in \mathbb{R}^{n(k+1)}$ we arrive at the first two constraints of problem (5)

$$[z_{ij} = 1] \implies [q_{ij} = p(x_i, \beta^{(j)}) - y_i], \quad \forall (i, j) \in J_1 \times J_2, \tag{21}$$

$$q_{ij}^2 \leq \alpha_i, \quad \forall (i, j) \in J_1 \times J_2, \tag{22}$$

and thus at the convex MIQCP (5), which can be solved with standard solvers.

In our numerical experiments, the computing time decreased significantly when bounds were added to the decision variable q , i.e., when the additional constraints $-\bar{q} \leq q \leq \bar{q}$ were imposed for some $\bar{q} > 0$. However, this offers the risk of excluding optimal solutions from the feasible set if the optimal solution is not known. Yet, bounds on q only lead to the exclusion of the optimal solution of problem (5) if in this optimal solution there exist $(i, j) \in J_1 \times J_2$ with $z_{ij} = 1$ such that $|p(x_i, \beta^{(j)}) - y_i| > \bar{q}$. This means that the y component of a data point (x_i, y_i) that is assigned to the j th polynomial in the optimal solution has to have a distance to the polynomial function evaluated at x_i that is larger than \bar{q} . So, by setting $\bar{q} := \max_i y_i - \min_i y_i$, these bounds should not affect the optimal solution in typical applications. Indeed, if the optimal solution does not satisfy these bounds, this might indicate that outliers are present in the data or that the spline function with the chosen number of free knots is not appropriate to approximate the given data set. Note that this reasoning is invalid for the bounds imposed by big-M constants since they have to hold for all $(i, j) \in J_1 \times J_2$, and not only for those that satisfy $z_{ij} = 1$.

Furthermore, we mention that, due to the convexity of the quadratic constraints, it is straightforward to linearize these constraints and approximate the convex MIQCP with a mixed-integer linear problem (MILP). Since we already impose bounds on q in our MIQCP formulation, we can simply choose m points $r_t \in [-\bar{q}, \bar{q}]$, compute for

$t = 1, \dots, m - 1$, the parameters

$$a_t = \frac{r_{t+1}^2 - r_t^2}{r_{t+1} - r_t},$$

$$b_t = r_t^2 - a_t \cdot r_t,$$

of the secants passing through the points (r_t, r_t^2) , $t = 1, \dots, m$, and replace the quadratic constraints

$$q_{ij}^2 \leq \alpha_i, \quad \forall (i, j) \in J_1 \times J_2,$$

with the linear constraints

$$a_t q_{ij} + b_k \leq \alpha_i, \quad \forall (i, j) \in J_1 \times J_2, \quad t = 1, \dots, m - 1.$$

However, some preliminary numerical tests for different choices of m and points $r_t \in [-\bar{q}, \bar{q}]$ indicate that it is not possible to decrease the computing time without significantly worsening the quality of the obtained solution. Therefore, we will not consider these linearization technique further.

Finally, alternative error functions can be used. A discussion of the impact of this on the considered problem can be found in Section A of the supplementary information.

4 A branch-and-bound algorithm for the combinatorial formulation

In this section, we describe a branch-and-bound algorithm for the solution of the combinatorial formulation (4) of the least-squares spline approximation problem with free knots. We start by sketching the main idea of branch-and-bound algorithms before going into the details our algorithm. A general description of branch-and-bound methods in combinatorial optimization can be found, e.g., in the monographs [2, 24, 25].

The main idea of branch-and-bound algorithms is to subsequently divide the problem into subproblems (*branching*) and compute lower bounds of the minimal values of those subproblems. All problems are stored in a list together with the corresponding lower bound. In case there are subproblems in the list whose lower bound exceeds a known upper bound for the optimal value of problem (4), these can be excluded from the list without further consideration as none of these subproblems can contain a globally minimal point. This is also known as *fathoming* or *pruning*. The algorithm terminates with the optimal solution after a finite number of steps.

4.1 Branching strategy

As a first step in describing our branching strategy, we define the subproblems into which the problem is subsequently divided in our branch-and-bound algorithm. To this end, for given index sets $\Lambda_0, \dots, \Lambda_k \subseteq \{1, \dots, n\}$ we consider the optimization problem

$$\begin{aligned}
 S(\Lambda_0, \dots, \Lambda_k) : \quad & \min_{\beta, I_0, \dots, I_k} \sum_{j=0}^k \sum_{i \in I_j} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \\
 \text{s.t.} \quad & \Lambda_j \subseteq I_j, \quad j = 0, \dots, k, \\
 & \bigcup_{j=0}^k I_j = \{1, \dots, n\}, \\
 & I_j \neq \emptyset, \quad j = 0, \dots, k, \\
 & r < q \text{ for all } r \in I_i \text{ and } q \in I_j \text{ with } i < j, \\
 & p(\xi_j, \beta^{(j-1)}) = p(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & p'(\xi_j, \beta^{(j-1)}) = p'(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & p''(\xi_j, \beta^{(j-1)}) = p''(\xi_j, \beta^{(j)}), \quad j = 1, \dots, k, \\
 & \beta \in \mathbb{R}^{4(k+1)},
 \end{aligned}$$

where

$$\xi_j = \left(\max_{i \in I_{j-1}} x_i + \min_{i \in I_j} x_i \right) / 2,$$

for all $j \in \{1, \dots, k\}$. We call the index sets $\Lambda_0, \dots, \Lambda_k$ restriction sets, since by adding indices to these sets we restrict the set of feasible partitions in problem $S(\Lambda_0, \dots, \Lambda_k)$.

Definition 4.1 Restriction sets $\Lambda_0, \dots, \Lambda_k \subseteq \{1, \dots, n\}$ are called *valid*, if there exists a partition I_0, \dots, I_k of the set $\{1, \dots, n\}$ such that the conditions

$$\Lambda_j \subseteq I_j, \quad j = 0, \dots, k, \tag{23}$$

$$I_j \neq \emptyset, \quad j = 0, \dots, k, \tag{24}$$

$$r < q \text{ for all } r \in I_i \text{ and } q \in I_j \text{ with } i < j, \tag{25}$$

are satisfied.

The basic idea of our branch-and-bound algorithm is to subsequently assign indices to the restriction sets in order to split problem (4) into subproblems $S(\Lambda_0, \dots, \Lambda_k)$ that allow the efficient computation of nontrivial lower bounds. Without loss of generality, we assume that the first point x_1 is assigned to the first polynomial, i.e., $1 \in I_0$, and the last point x_n is assigned to the last polynomial, i.e., $n \in I_k$. Thus, we can start the branch-and-bound algorithm with the initial restriction sets $\Lambda_0 := \{1\}$, $\Lambda_1 := \emptyset, \dots, \Lambda_{k-1} := \emptyset$, $\Lambda_k := \{n\}$ and the set of unassigned indices $I = \{2, \dots, n - 1\}$.

In order to divide a subproblem $S(\Lambda_0, \dots, \Lambda_k)$ further into subproblems, we choose an index $i \in I$ and assign it to one of the restriction sets $\Lambda_0, \dots, \Lambda_k$. In case we add

an index i to a restriction set Λ_j for which $i < \min(\Lambda_j)$ holds, then we also add the indices $\{i + 1, \dots, \min(\Lambda_j) - 1\}$ to Λ_j , since those indices cannot be assigned to another restriction set without making $S(\Lambda_0, \dots, \Lambda_k)$ infeasible. Analogously, if $i > \max(\Lambda_j)$, then we add the indices $\{\max(\Lambda_j) + 1, \dots, i - 1\}$ to Λ_j . Note that this implies that an index which is added to a restriction set always becomes the smallest or largest index in this restriction set. The following lemma gives simple conditions that ensure that the resulting restriction sets are valid, i.e., the new subproblem still permits feasible partitions.

Lemma 4.2 *Let $\Lambda_0, \dots, \Lambda_k \subset \{1, \dots, n\}$ be valid restriction sets, $1 \in \Lambda_0, n \in \Lambda_k$ and $j \in \{0, \dots, k\}$. In addition, let $i \in \{1, \dots, n\}$ be an index such that $i \notin \Lambda_m$ for all $m \in \{0, \dots, k\}$. Then the sets $\bar{\Lambda}_j := \Lambda_j \cup \{i\}$, $\bar{\Lambda}_m := \Lambda_m, \forall m \neq j$, are valid restriction sets if and only if the following conditions are satisfied:*

$$i < \min(\Lambda_j) \implies \max(\Lambda_\ell) \leq i + \ell - j, \tag{26}$$

$$i > \max(\Lambda_j) \implies \min(\Lambda_r) \geq i + r - j, \tag{27}$$

where $\ell := \max\{m \mid \Lambda_m \neq \emptyset, m \in \{0, \dots, j - 1\}\}$ and $r := \min\{m \mid \Lambda_m \neq \emptyset, m \in \{j + 1, \dots, k\}\}$.

Proof Since the sets $\Lambda_0, \dots, \Lambda_k$ are valid restriction sets, there exists a partition I_0, \dots, I_k of the set $\{1, \dots, n\}$ such that the conditions (23) to (25) are satisfied.

We show that if $i < \min(\Lambda_j)$, then condition (26) is satisfied if and only if the sets $\bar{\Lambda}_0, \dots, \bar{\Lambda}_k$ are valid restriction sets. The line of argument that if $i > \max(\Lambda_j)$ condition (27) is satisfied if and only if the sets $\bar{\Lambda}_0, \dots, \bar{\Lambda}_k$ are valid restriction sets is analogous and, therefore, omitted.

Thus, we assume that $i < \min(\Lambda_j)$ is true. Since $i < \min(\Lambda_j)$ and $1 \in \Lambda_0$ it follows that $j \geq 1$ and the set $\{m \mid \Lambda_m \neq \emptyset, m \in \{0, \dots, j - 1\}\}$ is nonempty, i.e., the index ℓ is well defined.

If we assume that condition (26) is satisfied, we can define the sets

$$\bar{I}_m := \begin{cases} \{\min(\Lambda_\ell), \dots, i + \ell - j\}, & \text{if } m = \ell, \\ \{i + m - j\}, & \text{if } \ell < m < j, \\ \{i, \dots, \max(\Lambda_j)\}, & \text{if } m = j, \\ I_m, & \text{if } m < \ell \text{ or } j < m, \end{cases}$$

for all $m \in \{0, \dots, k\}$. The sets $\bar{I}_0, \dots, \bar{I}_k$ form a partition of the set $\{1, \dots, n\}$ such that the conditions (23) to (25) are satisfied for the restriction sets $\bar{\Lambda}_0, \dots, \bar{\Lambda}_k$. Consequently, these restriction sets are valid.

On the other hand, assume that condition (26) is violated, i.e., $\max(\Lambda_\ell) \geq i + \ell - j + 1$. If $\max(\Lambda_\ell) > i$, it is clear that there exists no partition I_0, \dots, I_k of the set $\{1, \dots, n\}$ such that condition (25) is satisfied. If $\max(\Lambda_\ell) < i$, then $i + \ell - j + 1 \leq \max(\Lambda_\ell) < i$ and thus there are $j - \ell - 1 > 0$ empty restriction sets between the sets Λ_ℓ and Λ_j . In every partition I_0, \dots, I_k of the set $\{1, \dots, n\}$ that satisfies conditions (23) and (25) the $j - \ell - 1$ index sets I_m with $\ell < m < j$ cannot all be nonempty, since there are

Algorithm 4.1 Branching strategy

- 1: **Input:** Restriction sets $\Lambda_0, \dots, \Lambda_k$;
 - 2: Choose an unassigned index $i \in \{1, \dots, n\}$ (i.e., an index which is not yet contained in one of the restriction sets $\Lambda_0, \dots, \Lambda_k$);
 - 3: **for** $j = 0, \dots, k$ **do**
 - 4: **if** i and j satisfy conditions (26) and (27) **then**
 - 5: **if** $i < \min(\Lambda_j)$ **then**
 - 6: Set $\tilde{\Lambda}_j := \Lambda_j \cup \{i, \dots, \min(\Lambda_j) - 1\}$;
 - 7: **else**
 - 8: Set $\tilde{\Lambda}_j := \Lambda_j \cup \{\max(\Lambda_j) + 1, \dots, i\}$;
 - 9: **end if**
 - 10: Add $S(\Lambda_0, \dots, \Lambda_{j-1}, \tilde{\Lambda}_j, \Lambda_{j+1}, \dots, \Lambda_k)$ to list of subproblems \mathcal{P} ;
 - 11: **end if**
 - 12: **end for**
 - 13: **Output:** List of new subproblems \mathcal{P} ;
-

only $i - \max(\Lambda_\ell) - 1 \leq j - l - 2$ unassigned indices that can be assigned to these sets without violating condition (23) or (25). However, then condition (24) is violated and the restriction sets $\tilde{\Lambda}_0, \dots, \tilde{\Lambda}_k$ are not valid. □

Our branching strategy is described more formally in Algorithm 4.1. Note that there are different possibilities to choose an unassigned index in Step 2 of Algorithm 4.1. We propose to take the midpoint of the (ordered) set of currently unassigned indices.

If one obtains restriction sets $\Lambda_0, \dots, \Lambda_k$ with Algorithm 4.1 such that between two nonempty sets Λ_ℓ and Λ_r there are $\min(\Lambda_r) - \max(\Lambda_\ell) - 1 > 0$ empty sets $\Lambda_m = \emptyset$ with $\ell < m < r$, then one can set $\Lambda_m := \{\max(\Lambda_\ell) + m - \ell\}$ for $\ell < m < r$, since this is the only way to assign these indices such that one obtains valid restriction sets. Although this is used in our implementation, it is not explicitly mentioned in Algorithm 4.1.

Example 4.3 Suppose we are given the restriction sets $\Lambda_0 = \{1, 2\}$, $\Lambda_1 = \emptyset$, $\Lambda_2 = \emptyset$ and $\Lambda_3 = \{9\}$. Assume that in Algorithm 4.1 the so far unassigned index $i = 5$ is assigned to Λ_3 . The restriction sets obtained with Algorithm 4.1 are then $\Lambda_0 = \{1, 2\}$, $\Lambda_1 = \emptyset$, $\Lambda_2 = \emptyset$ and $\tilde{\Lambda}_3 = \{5, 6, 7, 8, 9\}$. In addition to that, the empty index sets $\tilde{\Lambda}_1$ and $\tilde{\Lambda}_2$ can be filled and we obtain the restriction sets $\Lambda_0 = \{1, 2\}$, $\tilde{\Lambda}_1 = \{3\}$, $\tilde{\Lambda}_2 = \{4\}$ and $\tilde{\Lambda}_3 = \{5, 6, 7, 8, 9\}$.

4.2 Computation of lower bounds

In order to obtain a nontrivial lower bound of the optimal value of problem $S(\Lambda_0, \dots, \Lambda_k)$ for given restriction sets $\Lambda_0, \dots, \Lambda_k$, we compute the optimal value of the optimization problem

$$\begin{aligned}
 L(\Lambda_0, \dots, \Lambda_k) : \quad & \min_{\beta} \sum_{j=0}^k \sum_{i \in \Lambda_j} \left(p(x_i, \beta^{(j)}) - y_i \right)^2 \\
 \text{s.t.} \quad & p(\xi_j, \beta^{(j-1)}) = p(\xi_j, \beta^{(j)}), \quad j \in J(\Lambda_0, \dots, \Lambda_k), \\
 & p'(\xi_j, \beta^{(j-1)}) = p'(\xi_j, \beta^{(j)}), \quad j \in J(\Lambda_0, \dots, \Lambda_k), \\
 & p''(\xi_j, \beta^{(j-1)}) = p''(\xi_j, \beta^{(j)}), \quad j \in J(\Lambda_0, \dots, \Lambda_k), \\
 & \beta \in \mathbb{R}^{4(k+1)},
 \end{aligned}$$

where

$$\xi_j := \left(\max_{i \in I_{j-1}} x_i + \min_{i \in I_j} x_i \right) / 2,$$

for all $j \in J(\Lambda_0, \dots, \Lambda_k)$ and

$$J(\Lambda_0, \dots, \Lambda_k) := \{j \in \{1, \dots, k\} \mid 1 + \max(I_{j-1}) = \min(I_j)\}.$$

Note that $L(\Lambda_0, \dots, \Lambda_k)$ is a least-squares spline approximation problem with fixed knots, and can be solved via a linear system of equations. The only difference is that only knots ξ_j with $j \in J(\Lambda_0, \dots, \Lambda_k)$ and the corresponding continuity restrictions are included in the problem. Note that the index set $J(\Lambda_0, \dots, \Lambda_k)$ contains only those indices of knots which are not subject to change when the problem $S(\Lambda_0, \dots, \Lambda_k)$ is further divided into subproblems in subsequent branching steps.

Lemma 4.4 *Let v_L be the optimal value of problem $L(\Lambda_0, \dots, \Lambda_k)$ and v_S the optimal value of problem $S(\Lambda_0, \dots, \Lambda_k)$. Then the inequality $v_L \leq v_S$ holds.*

Proof Let β^* and I_0^*, \dots, I_k^* denote an optimal solution of $S(\Lambda_0, \dots, \Lambda_k)$. Since the partition I_0^*, \dots, I_k^* is feasible for problem $S(\Lambda_0, \dots, \Lambda_k)$, it must hold that $\Lambda_j \subseteq I_j^*$ for all $j \in \{0, \dots, k\}$. Therefore the inequality

$$\sum_{j=0}^k \sum_{i \in \Lambda_j} \left(p(x_i, \beta^{(j),*}) - y_i \right)^2 \leq \sum_{j=0}^k \sum_{i \in I_j^*} \left(p(x_i, \beta^{(j),*}) - y_i \right)^2$$

is satisfied. Due to $J(\Lambda_0, \dots, \Lambda_k) \subseteq \{1, \dots, k\}$ we have that every vector β that is feasible for problem $S(\Lambda_0, \dots, \Lambda_k)$ is also feasible for problem $L(\Lambda_0, \dots, \Lambda_k)$. Consequently, the optimal solution β^* is feasible for problem $L(\Lambda_0, \dots, \Lambda_k)$ and we obtain

$$v_L \leq \sum_{j=0}^k \sum_{i \in \Lambda_j} \left(p(x_i, \beta^{(j),*}) - y_i \right)^2 \leq \sum_{j=0}^k \sum_{i \in I_j^*} \left(p(x_i, \beta^{(j),*}) - y_i \right)^2 = v_S.$$

□

Algorithm 4.2 Branch-and-bound algorithm for problem (4)

1: **Input:** n data points $(x_i, y_i) \in \mathbb{R}^2$, number of free knots k ;

2: Initialize list of subproblems as $\mathcal{L} := \{(l_S, S(\Lambda_0, \dots, \Lambda_k))\}$, where $l_S := 0$ and $\Lambda_0 := \{1\}$, $\Lambda_1 := \emptyset, \dots, \Lambda_{k-1} := \emptyset, \Lambda_k := \{n\}$;

3: Use a heuristic method to determine initial partition (I_0^*, \dots, I_k^*) and initial upper bound $u \in [0, \infty)$;

4: **while** $\mathcal{L} \neq \emptyset$ **do**

5: Select $(l_S, S(\Lambda_0, \dots, \Lambda_k)) \in \mathcal{L}$ with smallest lower bound l_S ;

6: Create set \mathcal{P} of new subproblems using Algorithm 4.1;

7: **for** $S(\Lambda_0, \dots, \Lambda_k) \in \mathcal{P}$ **do**

8: Compute the optimal value l_S of problem $L(\Lambda_0, \dots, \Lambda_k)$;

9: **if** $l_S < u$ **then**

10: Add $(l_S, S(\Lambda_0, \dots, \Lambda_k))$ to \mathcal{L} ;

11: **if** $\bigcup_{j=0}^k \Lambda_j = \{1, \dots, n\}$ **then**

12: Set $u := l_S$ and $I_0^* := \Lambda_0, \dots, I_k^* := \Lambda_k$;

13: **end if**

14: **end if**

15: **end for**

16: **for** $(l_S, S(\Lambda_0, \dots, \Lambda_k)) \in \mathcal{L}$ **with** $l_S \geq u$ **do**

17: Remove $(l_S, S(\Lambda_0, \dots, \Lambda_k))$ from \mathcal{L} ;

18: **end for**

19: **end while**

20: **Output:** Globally optimal partition (I_0^*, \dots, I_k^*) for problem (4);

The optimal value of $L(\Lambda_0, \dots, \Lambda_k)$ is increasing as more indices are added to the restriction sets $\Lambda_0, \dots, \Lambda_k$. When the set of unassigned indices is empty, then the optimal value of $L(\Lambda_0, \dots, \Lambda_k)$ coincides with the optimal value of the subproblem $S(\Lambda_0, \dots, \Lambda_k)$. Consequently, we can use this technique to efficiently compute lower bounds for the subproblems and we obtain a branch-and-bound algorithm that terminates with an optimal solution after a finite number of steps.

4.3 Outline of the branch-and-bound algorithm

In Algorithm 4.2 our branch-and-bound method to solve the combinatorial formulation (4) of the least-squares spline problem with free knots is described formally.

The initial partition and initial upper bound can be determined by any heuristic method, or one could simply choose equally spaced knots. Moreover, in addition to the upper bounds obtained in Algorithm 4.2 when the set of unassigned indices is empty, every subproblem $S(\Lambda_0, \dots, \Lambda_k)$ can be used to compute an upper bound for the globally minimal value of problem (4). To achieve this, one may simply generate new restriction sets $\bar{\Lambda}_0, \dots, \bar{\Lambda}_k$ such that all remaining indices are assigned, i.e., $\bigcup_{j=0}^k \bar{\Lambda}_j = \{1, \dots, n\}$, and $\Lambda_j \subseteq \bar{\Lambda}_j$ holds for all $j \in \{0, \dots, k\}$. Then the optimal value of problem $L(\bar{\Lambda}_0, \dots, \bar{\Lambda}_k)$ serves as an upper bound for the globally minimal value of problem (4).

In our numerical tests those upper bounds only had a minor influence on the runtime of the algorithm. Therefore, we did not compute additional upper bounds during the

Table 1 Least-squares errors of the solutions obtained with the FR, CF and CF+FR methods when applied to the *titanium heat* data set

Number of knots	FR	CF	CF+FR
2	2.1157	2.0741	2.0703
3	1.9209	0.5006	0.4651
4	0.0904	0.0681	0.0654
5	0.0962	0.0093	0.0088

Lowest errors are marked in bold

optimization process. However, upper bounds can help to keep the list of subproblems \mathcal{L} short and ensure that the main memory does not become a limiting factor.

5 Numerical experiments

This section comprises numerical experiments for the algorithms developed. First, we compare our combinatorial formulation of Sect. 2.3 that can be solved to global optimality by using our new approaches with a purely local method applied to problem (2). Furthermore, we propose the combination of both approaches. Here we are mainly concerned with the quality of the solution, since, in fact, two slightly different problems are considered. In Sect. 5.2 we compare our new methods for the global solution of the spline approximation problem in order to see, which of these methods performs better.

5.1 Combination with a local approach

We consider three different approaches in this section. First, we consider a variant of the Fletcher-Reeves nonlinear conjugate gradient method (FR) presented in [9] in order to solve problem (2) locally. Second, the combinatorial formulation (CF) presented in Sect. 2.3 is solved to global optimality by using our newly developed approaches. Finally, a combination of the combinatorial formulation with the Fletcher-Reeves nonlinear conjugate gradient method (CF+FR) is used, which means that we try to improve the solution of our global approach by using it as a starting point for the local solver for the relaxed problem (2) where the knots can be placed continuously.

We use two real data sets from the literature to empirically compare the quality of the solutions obtained with each of these approaches. The FR method is always initialized with the equidistant knot placement.

5.1.1 Titanium heat data

The *titanium heat* data set is introduced in [6] as an example of a data set that is difficult to approximate with classical methods. It is actual data that expresses a thermal property of titanium. It consists of 49 data points and contains a significant amount of noise.

Table 1 shows the least-squares errors of the spline functions corresponding to the knot placements computed with the three approaches for different numbers of

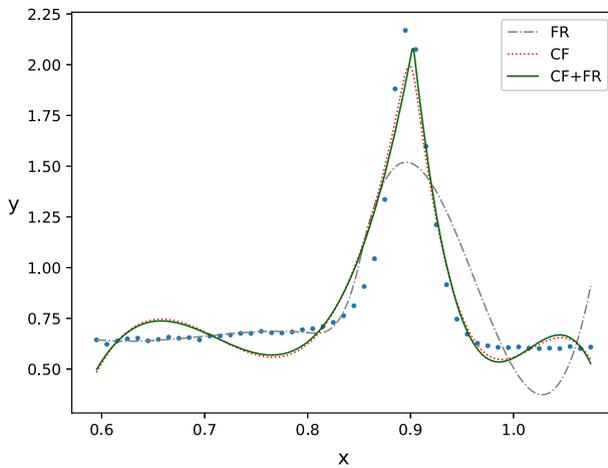


Fig. 4 Spline functions with three free knots corresponding to the solutions obtained with the FR, CF and CF+FR methods when applied to the *titanium heat* data set

free knots. We can observe that the combination of the CF and FR methods leads to substantially better solutions than the FR method alone. Compared to the improvement of the CF over the FR method, the local improvement of the CF method with the FR method seems marginal.

Clearly, any heuristic method could be combined with the FR method in order to try to find high-quality local solutions. However, there is a theoretically and practically relevant difference. Whereas there is no optimality guarantee except local optimality if we use a heuristic knot placement as an initial solution of the FR method, the CF+FR approach guarantees that the approximated local solution is at least as good as the optimal solution of the combinatorial formulation.

In Fig. 4 the spline functions corresponding to the knot placements computed with the three methods are depicted for the case of three free knots. Clearly, the solution obtained by the FR method is not satisfactory. The other two methods yield similarly good approximations with the only visible difference being the approximation quality near the peak in the data set.

5.1.2 Angular displacement data

The *angular displacement* data set is introduced in [26] for the evaluation of differentiation methods of film data of human motion. We use the modification of this data set proposed in [18], which contains larger errors and is supposed to be more realistic than the original data set. It consists of 142 data points.

In Table 2 the least-squares errors of the spline functions corresponding to the knot placements computed with the three approaches for different numbers of free knots are presented. The overall picture is similar to the one obtained from the *titanium heat* data set. The FR method seems to approximate the globally minimal point for two and three free knots, even when it is initialized with the equidistant knot placement.

Table 2 Least-squares errors of the solutions obtained with the FR, CF and CF+FR methods when applied to the *angular displacement* data set

Number of knots	FR	CF	CF+FR
2	11.4483	11.4496	11.4483
3	1.5298	1.5384	1.5297
4	1.5266	0.1493	0.1465
5	1.3663	0.0552	0.0545

Lowest errors are marked in bold

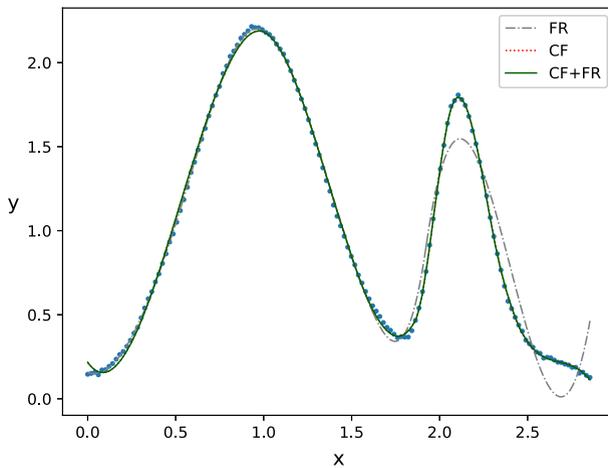


Fig. 5 Spline functions with five free knots corresponding to the solutions obtained with the FR, CF and CF+FR methods when applied to the *angular displacement* data set

However, the least-squares error of the locally minimal point approximated in case of four and five free knots is much higher than the result obtained with the other methods. Again, the CF method is only slightly inferior to the combined CF+FR approach.

The similarity of the results obtained with the CF and CF+FR approaches is also visible in Fig. 5, where only two curves can be distinguished, i.e. the suboptimal fit obtained with the FR method initialized with the equidistant knot placement, and the high quality approximation obtained with the other methods.

5.2 Comparison of MIQCP and Branch-and-Bound Approaches

In the previous section, it already became clear that it is possible to obtain high-quality solutions of the least-squares spline approximation problem with free knots by using our global optimization approach and this can even be improved using local solvers if we aim at a solution with continuous knot placement as done in the formulation of problem (2). In this section we investigate whether the MIQCP or the branch-and-bound approach is more efficient for the solution of the combinatorial formulation. To this end, once again we consider the two real data sets from the previous section to gain some preliminary insights into the relative performance of the approaches.

Table 3 Solve times (in seconds) and least-squares errors of the solutions obtained with the branch-and-bound and MIQCP approaches for the *titanium heat* data set

Number of knots	Branch-and-bound		MIQCP	
	Time	Error	Time	Error
2	0.07	2.0741	0.88	2.0741
3	0.40	0.5006	2.00	0.5006
4	1.99	0.0681	9.31	0.0681
5	5.55	0.0093	12.09	0.0093

Lowest errors and computation times are marked in bold

In addition, we use a larger testbed of synthetic data sets in order to decide which approach should be preferred in practice.

The branch-and-bound algorithm is implemented in Python 3.6. We use the commercial optimization solver Gurobi (version 8.1.1) and its Python API to model and solve the MIQCPs. The computations are performed on an Intel Core i7-9700K with 32 GB of main memory. The time limit for each algorithm is set to 3600 s.

5.2.1 Titanium heat and angular displacement data sets

Tables 3 and 4 show the least-squares errors of the knot placements computed via the branch-and-bound and MIQCP approaches for the *titanium heat* and *angular displacement* data sets and the corresponding solve times. As expected, both algorithms are able to determine the globally optimal value, except for the case of five free knots and the displacement data set, where the MIQCP approach does not terminate within the given time limit. The main observation is that the branch-and-bound algorithm needs significantly less time than the MIQCP approach. It is consistently faster by an order of magnitude.

We can also make some preliminary observations concerning the growth in computation time when the number of data points or free knots is increased. The *angular displacement* data set contains almost exactly three times as many data points as the *titanium heat* data set. This leads to about a tenfold increase in computation time for the branch-and-bound method. In contrast to that, the computation time of the MIQCP method increases a hundredfold. However, increasing the number of free knots by one leads to an increase in the computing time of the branch-and-bound algorithm by a factor between 5 and 10, whereas the computing time of the MIQCP approach increases only by a factor between 1.5 and 5. We further investigate the dependency of the computation time on the number of data points and knots in the next section.

5.2.2 Synthetic data sets

In this section, we present the results of more extensive numerical experiments with synthetic data sets. The functions used to generate the synthetic data sets are listed in Table 8 in Section B of the supplementary information, along with further details on the nature and magnitude of the added noise and the sources that first used these synthetic data sets in the literature on free knot spline approximation. Each of these functions is used to generate five synthetic data sets of different sizes. In combination with varying

Table 4 Solve times (in seconds) and least-squares errors of the solutions obtained with the branch-and-bound and MIQCP approaches for the *angular displacement* data set

Number of knots	Branch-and-bound		MIQCP	
	Time	Error	Time	Error
2	0.61	11.4496	78.77	11.4496
3	8.14	1.5384	379.31	1.5384
4	44.95	0.1493	1250.79	0.1493
5	446.37	0.0552	3600.00	0.0563

Lowest errors and computation times are marked in bold

Table 5 Combinations of data set sizes and number of free knots that are considered for generating instances of the spline approximation problem with the functions in Table 8 in Section B of the supplementary information

Number of data points	Number of free knots
50	{2, 3, 4, 5, 6, 7}
100	{2, 3, 4, 5, 6}
200	{2, 3, 4, 5}
400	{2, 3, 4}
800	{2, 3}

numbers of free knots, we obtained 20 problem instances for each of the 12 functions, yielding 240 problem instances in total. Table 5 shows all the combinations of data set sizes and number of free knots that are considered.

The detailed results of the experiments are listed in Section C of the supplementary information of this paper. However, in the following we briefly comment on the main observations. Concerning the solve times required by the approaches, the branch-and-bound algorithm is faster than the MIQCP approach on all but three problem instances. Of the 240 problem instances, the branch-and-bound approach is able to solve 218 (90.8%) within the specified time limit of 3600 s. In contrast to that, the MIQCP approach is only able to solve 141 (58.8%) of the problem instances in the given time limit.

In particular, when the size of the data set is large, the MIQCP approach frequently fails to compute the optimal solution within the time limit. Among 24 problem instances with data sets containing 800 points, only one could be solved to optimality with the MIQCP approach, whereas the branch-and-bound algorithm computed the optimal solution for 23 of these problem instances. These findings support the preliminary observations made in the previous section that the computing time required by the branch-and-bound algorithm grows slower in the size of the data set than the computing time of the MIQCP approach.

In order to further investigate the dependency of the computing time on the size of the data set, we consider problem instances with three free knots and compare the solve times required by each method for different data set sizes. Figure 6 depicts the solve times as box plots on a log scale. From this figure one can observe that the required solve times of the branch-and-bound and MIQCP approaches increase exponentially in the number of data points, but the growth constant is smaller for the

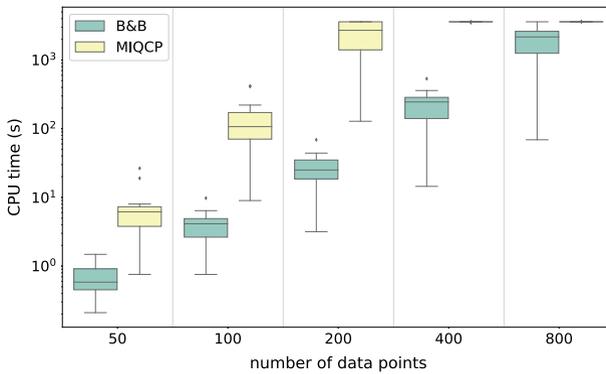


Fig. 6 Solve times (in seconds) required by the branch-and-bound and MIQCP approaches in order to compute the optimal spline functions with three free knots. There are 12 synthetic data sets for each data set size and the time limit for each algorithm is set to 3600 s

Table 6 Percentage of solved instances and average times (in seconds) required by the branch-and-bound and MIQCP approaches to compute the optimal spline functions with three free knots

Number of data points	Branch-and-bound		MIQCP	
	Time	Solved (%)	Time	Solved (%)
50	0.71	100	7.89	100
100	4.06	100	152.13	100
200	27.18	100	2425.97	66.7
400	222.81	100	3591.73	8.3
800	1927.69	91.7	3600.00	0

There are 12 synthetic data sets for each data set size and the time limit for each algorithm is set to 3600 s

branch-and-bound algorithm. For most problems, the times needed by the branch-and-bound algorithm are an order of magnitude smaller than the times required by the MIQCP approach. This can also be observed in Table 6, which shows the average times and the percentages of solved problem instances. Note that unsolved problem instances enter the average times with the maximal computing time of 3600 s. Thus, if there are many unsolved instances, the average time shown in the table do not give an accurate value for the average time in absence of any time limit, which should be expected to be much higher.

Next, we investigate the dependence of the computing time on the number of free knots. In order to visualize this, we fix the size of the data set to 50 and compare the solve times required by each method for problem instances with different numbers of free knots. From the box plots in Fig. 7 we conclude that the time required by both methods grows exponentially in the number of free knots, and the growth constants seem to be roughly similar. This can also be observed in Table 7, where the average time required by the MIQCP approach is consistently 8–10 times larger than the time required by the branch-and-bound algorithm, as long as both methods solve all problem instances in the given time limit, which is the case for up to five knots.

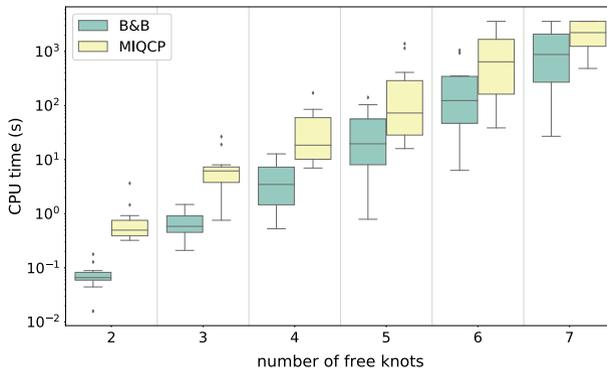


Fig. 7 Solve times (in seconds) required by the branch-and-bound and MIQCP approaches in order to compute the optimal spline functions for 50 data points. There are 12 synthetic data sets for each data set size and the time limit for each algorithm is set to 3600 s

Table 7 Percentage of solved instances and average times (in seconds) required by the branch-and-bound and MIQCP approaches to compute the optimal spline functions for 50 data points

Number of free knots	Branch-and-bound		MIQCP	
	Time	Solved (%)	Time	Solved (%)
2	0.08	100	0.85	100
3	0.71	100	7.88	100
4	4.79	100	42.08	100
5	37.76	100	307.34	100
6	283.99	100	1222.19	75.0
7	1287.84	83.3	2265.58	58.3

There are 12 synthetic data sets for each data set size and the time limit for each algorithm is set to 3600 s

In summary, both approaches enable the computation of globally optimal solutions to least-squares spline approximation problems with free knots of practically relevant sizes. Yet, the branch-and-bound algorithm is superior to the MIQCP approach in terms of computing time.

Finally, we want to point out that we ran the experiments also for spline functions with no continuity restrictions, which corresponds to fitting piecewise polynomials to the data. The results for this can be found in Sections D and E of the supplementary information of this paper.

6 Conclusions

We propose two different possibilities for the computation of globally optimal least-squares spline approximations with free knots. Although we assume possible knot placements to be discrete, we demonstrate that solutions computed by using our approach commonly do not differ much from the continuous case and can, therefore, serve as starting points for local optimization algorithms. Our numerical experiments confirm that both approaches, the MIQCP formulation and a tailored branch-and-

bound method, enable the solution of problems of practically relevant sizes. Although both methods work well, the branch-and-bound algorithm is even superior in terms of the required time.

There are several possible directions for future research. One idea is to accelerate the solution of the linear systems of equations in the branch-and-bound approach by observing that the linear systems, which are solved in order to determine lower bounds for the subproblems, become banded and therefore the method of Cholesky for positive band matrices could be used, see [22]. Going even further, one could make use of the fact that the restriction sets $\bar{\Lambda}_0, \dots, \bar{\Lambda}_k$ obtained with Algorithm 4.1 are oftentimes very similar to the sets $\Lambda_0, \dots, \Lambda_k$ that were provided as input to the algorithm. In many cases, only a single index is added to one of the restriction sets. Thus, in order to compute a lower bound for the subproblem $S(\bar{\Lambda}_0, \dots, \bar{\Lambda}_k)$ one typically has to solve a linear system of equations that is rather similar to the linear system that is solved in a prior iteration when a lower bound of the subproblem $S(\Lambda_0, \dots, \Lambda_k)$ is computed. Thus, we expect that the efficiency of the branch-and-bound algorithm could be improved significantly by employing the orthogonalization method described in [5], saving the QR decomposition for each subproblem and solving the modified linear systems with Givens rotations, see, e.g., [12].

Another idea for further research is an extension to the multivariate case, i.e., having data $(x_{i1}, \dots, x_{id}, y_i) \in \mathbb{R}^{d+1}$ for $d > 1$. In this case, computing globally optimal approximations is expected to be much more challenging for several reasons. For instance, there are several possibilities to partition the search domain into elements such as a triangulation or a grid. The idea of placing knots to represent boundaries as done in this article naturally leads to a certain kind of grid. However, there are different possibilities to define such a grid. For instance, we may allow for different hyperrectangles, on which the spline functions live, which can be of any form and different from each other. Another possibility is to restrict the number of knots in each dimension, e.g. k_1 knots partitioning the x_1 line, k_2 knots partitioning the x_2 line, etc., such that the splines live on the intersections of these knot placements. This immediately leads to $k = \sum_i k_i$ knots and $\prod_i (k_i + 1)$ splines that have to be estimated. Unfortunately, the number of partition elements is likely to increase exponentially in the number of dimensions, which holds for different forms of the elements. We suspect that this can easily lead to prohibitive computation times.

In addition, it is not entirely clear how such a situation can be tackled in a feasible way using our approaches. Depending on the specific statement of the problem, we think that it is possible to derive an analogous MIQCP formulation that can be solved using state-of-the-art solvers. Moreover, especially if the search domain is supposed to be partitioned into hyperrectangles, we think that a similar branch-and-bound algorithm tailored to this problem can be developed. At least, the concept of valid restriction sets can be transferred to this case. Moreover, a similar lower bounding procedure might be possible by omitting data points that are not assigned to specific partition elements yet.

A natural idea to create a suitable partition might be to start with a coarse tessellation that is subsequently refined by further subdividing elements on which there are large approximation errors. Although this idea is similar to what is done in so-called spatial branch-and-bound algorithms in global optimization, we think that such an approach

will at most lead to locally optimal solutions of the spline approximation problem and, actually, we do not see a possibility to drive such a procedure towards globally optimal points.

Another topic for future research could be the development of a branch-and-bound algorithm that is able to compute a global solution of the continuous problem with free knots. However, we suspect that such an algorithm might only be able to solve smaller problem instances. Furthermore, our numerical results presented in this article indicate that there might be at most small improvements on most instances so that it is in question whether such an algorithm is of practical interest at all and whether the additional time needed to solve such problems is worth the effort.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s10589-023-00462-7>.

Acknowledgements We thank Oliver Grothe, Tim Ortkamp, Oliver Stein, and an anonymous reviewer for helpful comments. Parts of this work were completed while RM was a PhD student at Karlsruhe Institute of Technology.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data availability All data and code to reproduce the data are included in the supplementary information of this article.

Declarations

Conflict of interest No funds, grants, or other support was received. The authors have no competing interests to declare that are relevant to the content of this article.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Beliakov, G.: Least squares splines with free knots: global optimization approach. *Appl. Math. Comput.* **149**(3), 783–798 (2004)
2. Bertsimas, D., Tsitsiklis, J.N.: *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA (1997)
3. Brusco, M.J., Stahl, S.: *Branch-and-Bound Applications in Combinatorial Data Analysis*. Statistics and Computing. Springer, New York, NY (2005)
4. IBM ILOG CPLEX V12.7, Users manual for CPLEX. International Business Machines Corporation (2017)
5. Cox, M.G.: The least squares solution of overdetermined linear equations having band or augmented band structure. *IMA J. Numer. Anal.* **1**(1), 3–22 (1981)
6. de Boor, C., Rice, J.R.: Least squares cubic spline approximation, II—variable knots. Technical Report No. CSD TR 21, Purdue University (1968)

7. de Boor, C.: A practical guide to splines. Springer, Berlin (2001)
8. Denison, D.G.T., Mallick, B.K., Smith, A.F.M.: Automatic Bayesian curve fitting. *J. R. Stat. Soc.: Ser. B (Methodol.)* **60**(3), 333–350 (1998)
9. Dierckx, P.: Curve and surface fitting with splines. Monographs on Numerical Analysis. Clarendon Press, Oxford (1996)
10. DiMatteo, I., Genovese, C.R., Kass, R.E.: Bayesian curve-fitting with free-knot splines. *Biometrika* **88**(4), 1055–1071 (2001)
11. Ronald Gallant, A., Fuller, W.A.: Fitting segmented polynomial regression models whose join points have to be estimated. *J. Am. Stat. Assoc.* **68**(341), 144–147 (1973)
12. Morven Gentleman, W.: Least squares computations by givens transformations without square roots. *IMA J. Appl. Math.* **12**(3), 329–336 (1973)
13. Golub, G.H., Pereyra, V.: The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM J. Numer. Anal.* **10**(2), 413–432 (1973)
14. Gurobi Optimization, LLC Gurobi Optimizer Reference Manual (2020)
15. Hand, D.J.: Branch and bound in statistical data analysis. *J. R. Stat. Soc. Ser. D (The Statistician)* **30**(1), 1–13 (1981)
16. Yingkang, H.: An algorithm for data reduction using splines with free knots. *IMA J. Numer. Anal.* **13**, 365–381 (1993)
17. Jupp, D.L.: Approximation to data by splines with free knots. *SIAM J. Numer. Anal.* **15**(2), 328–343 (1978)
18. Lanshammar, H.: On practical evaluation of differentiation techniques for human gait analysis. *J. Biomech.* **15**(2), 99–105 (1982)
19. Lee, T.C.: On algorithms for ordinary least squares regression spline fitting: a comparative study. *J. Stat. Comput. Simul.* **72**(8), 647–663 (2002)
20. Lindstrom, M.J.: Penalized estimation of free knot splines. *J. Comput. Graph. Stat.* **8**(2), 333–352 (1999)
21. Mammen, E., van de Geer, S.: Locally adaptive regression splines. *Ann. Stat.* **25**(1), 387–413 (1997)
22. Martin, R.S., Peters, G., Wilkinson, J.H.: Symmetric decomposition of a positive definite matrix. *Numer. Math.* **7**(5), 362–383 (1965)
23. Miyata, S., Shen, X.: Adaptive free-knot splines. *J. Comput. Graph. Stat.* **12**(1), 197–213 (2003)
24. Nemhauser, G.L., Wolsey, L.A.: Integer and Combinatorial Optimization. Wiley-Interscience, New York, NY (1988)
25. Papadimitriou, C.H., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall Inc, Upper Saddle River, NJ (1982)
26. Pezzack, J.C., Norman, R.W., Winter, D.A.: An assessment of derivative determining techniques used for motion analysis. *J. Biomech.* **10**(5–6), 377–382 (1977)
27. Pittman, J., Murthy, C.A.: Fitting optimal piecewise linear functions using genetic algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(7), 701–718 (2000)
28. Poirier, D.J.: Piecewise regression using cubic splines. *J. Am. Stat. Assoc.* **68**(343), 515–524 (1973)
29. Ruppert, D.: Selecting the number of knots for penalized splines. *J. Comput. Graph. Stat.* **11**(4), 735–757 (2002)
30. Schwetlick, H., Schütze, T.: Least squares approximation by splines with free knots. *BIT Numer. Math.* **35**(3), 361–384 (1995)
31. Smith, P.L.: Curve fitting and modeling with splines using statistical variable selection techniques. Report NASA 166034, NASA Langley Research Center, Hampton (1982)
32. Spiriti, S., Eubank, R., Smith, P.W., Young, D.: Knot selection for least-squares and penalized splines. *J. Stat. Comput. Simul.* **83**(6), 1020–1036 (2013)
33. Stein, O.: Grundzüge der Globalen Optimierung. Springer Spektrum, Berlin (2018)
34. Stone, C.J., Hansen, M.H., Kooperberg, C., Truong, Y.K.: Polynomial splines and their tensor products in extended linear modeling. *Ann. Stat.* **25**(4), 1371–1425 (1997)
35. Wand, M.P.: A comparison of regression spline smoothing procedures. *Comput. Stat.* **15**, 443–462 (2000)
36. Wold, S.: Spline functions in data analysis. *Technometrics* **16**(1), 1–11 (1974)