Acta Numerica (2026), pp. 173–272
doi:10.1017/S0962492926100294

Nonlinear model reduction for transport-dominated problems

Jan S. Hesthaven
Karlsruhe Institute of Technology, Kaiserstr. 12, 76131 Karlsruhe, Germany
E-mail: jan.hesthaven@kit.edu

Benjamin Peherstorfer
Courant Institute of Mathematical Sciences, New York University,
251 Mercer Street, New York, NY 10012, USA
E-mail: pehersto@cims.nyu.edu

Benjamin Unger
Karlsruhe Institute of Technology, Englerstr. 2, 76131 Karlsruhe, Germany
E-mail: benjamin.unger@kit.edu

This article surveys nonlinear model reduction methods that remain effective in regimes where linear reduced-space approximations are intrinsically inefficient, such as transport-dominated problems with wave-like phenomena and moving coherent structures, which are commonly associated with the Kolmogorov barrier. The article organizes nonlinear model reduction techniques around three key elements – nonlinear parametrizations, reduced dynamics and online solvers – and categorizes existing approaches into transformation-based methods, online adaptive techniques, and formulations that combine generic nonlinear parametrizations with instantaneous residual minimization.

2020 Mathematics Subject Classification: 65M99, 41A46, 65F55, 68T07

CONTENTS

1. Introduction

We provide a brief, non-technical overview of model reduction and discuss the challenge of reducing transport-dominated problems.

1.1. Model reduction

In many applications in computational science and engineering, simulations with a numerical model appear within an outer loop, so that multiple numerical simulations are required, at many different parameter configurations, inputs and initial conditions. Examples include design, control, optimization, uncertainty quantification and inverse problems. The high cost of repeatedly performing numerical simulations is a major barrier to solving outer-loop and other many-query applications.[1] Model reduction seeks reduced models to perform numerical simulations at greatly reduced costs and has therefore emerged as a key technique for making outer-loop applications tractable; see Antoulas (2005), Rozza, Huynh and Patera (2008), Benner, Gugercin and Willcox (2015), Antoulas, Beattie and Gugercin (2020), Hesthaven, Pagliantini and Rozza (2022*b*), Peherstorfer (2022) and Kramer, Peherstorfer and Willcox (2024) for surveys.

Model reduction exploits two fundamental properties of outer-loop applications (Rozza *et al.* 2008). First, across the range of inputs, parameters and initial conditions relevant to an outer-loop application, the solutions of the numerical simulations typically vary in a structured, rather than arbitrary, manner across the generic solution space. The model reduction methods considered in this survey aim to identify this structured variability and exploit it to construct reduced models with reduced solutions that depend on only a small number of degrees of freedom, compared to the solutions of the original high-fidelity numerical model – henceforth

[1] See Peherstorfer, Willcox and Gunzburger (2018) for a discussion about the distinction of outer-loop and many-query applications in the context of multi-fidelity methods. In this survey we use the terms interchangeably.

the full model – and its corresponding full solutions. Second, the performance of outer-loop applications is governed primarily by the aggregate computational cost of all numerical simulations. This focus on aggregate costs allows model reduction to incur one-time high pre-processing costs in an offline (or training) phase to construct reduced models, provided that subsequent reduced-model simulations in an online (or test) phase are sufficiently inexpensive for the pre-processing costs to be amortized over the outer loop. Leveraging these two opportunities has enabled model reduction to play a key role in making outer-loop applications tractable.

Remark 1.1. Model reduction is an umbrella term that includes a wide range of approaches beyond the methods emphasized in this survey, including reduced-physics models and purely data-driven surrogate models. In this survey we focus on so-called projection-based reduced models that are built from full-model information and that represent a reduced solution with a low-dimensional state that evolves under reduced dynamics, even though for nonlinear parametrizations the resulting constructions are not necessarily projections in the classical vector space sense.

1.2. Linear model reduction and its limitation

The classical approach to (projection-based) model reduction is approximating full-model solutions in a reduced space of low dimension, relative to the dimension of the generic full-solution space over which the full model is formulated. A reduced space is a problem-dependent low-dimensional approximation space, typically a subspace of the full-solution Hilbert space. It is meant to approximate the set of all full-model solutions over the inputs, parameters and initial conditions of interest in an outer-loop application – a set that is often referred to as the solution manifold. We refer to model reduction methods as linear if the reduced solutions are linear approximations of the full solutions in the reduced space. We stress that in linear model reduction a single, global reduced space is used. Examples of linear model reduction methods include balanced truncation (Mullis and Roberts 1976, Moore 1981), proper orthogonal decomposition (Sirovich 1987, Holmes, Lumley and Berkooz 1996), interpolation methods (Antoulas 2005, Baur, Beattie, Benner and Gugercin 2011) and the reduced basis method (Maday and Rønquist 2002, Rozza *et al.* 2008).

There are linear model reduction methods that are applicable to full models with nonlinear dynamics, meaning numerical models whose governing operators – and thus the time evolution – depend nonlinearly on the solution function. The term ‘linear’ in linear model reduction refers solely to the ansatz for the reduced solution, meaning that the reduced solution is constrained to lie in a low-dimensional space with a linear vector-space structure. Consequently, although the approximation ansatz is linear, the resulting reduced model can still exhibit nonlinear dynamics. This is analogous to finite element discretizations of nonlinear partial differential equations (PDEs), where the solution field is also represented by a linear combination of basis functions of a finite element solution space.

Tremendous progress has been made on linear model reduction to turn it into a mature and reliable technology, including effective ways to construct expressive reduced spaces, *a posteriori* error estimation, handling problems with nonlinear solution field dependence, and structure preservation. We refer to surveys (Antoulas, Sorensen and Gugercin 2001, Rozza *et al.* 2008, Benner *et al.* 2015, Hesthaven *et al.* 2022*b*, Kramer *et al.* 2024) and monographs (Antoulas 2005, Quarteroni and Rozza 2014, Hesthaven, Rozza and Stamm 2016, Quarteroni, Manzoni and Negri 2016, Benner, Cohen, Ohlberger and Willcox 2017, Antoulas *et al.* 2020, Benner *et al.* 2021*a*,*b*) that provide overviews of the progress in linear model reduction.

However, linear model reduction has a major, fundamental limitation: solutions of the full model must not only lie approximately on a smooth, low-dimensional manifold in the full-model solution space but have to be well approximated by a low-dimensional space with a linear vector space structure. The requirement that reduced solutions are linear approximations in a vector space imposes a strong structural restriction. Even if the solution manifold is smooth and of low dimension, it may be strongly curved or otherwise far from linear and therefore cannot be captured well with a low-dimensional vector space. For a class of problems related to well-behaved diffusion (elliptic/parabolic) problems, it has been shown that solution manifolds can be well approximated with reduced spaces, that is, there exist sequences of reduced spaces with increasing dimension n that achieve approximation errors that decay exponentially in the reduced dimension n in a suitable metric (Maday, Patera and Turinici 2002, Buffa *et al.* 2012). For this class of problems, linear model reduction is well suited, and this class has been the primary focus of model reduction for a long time. In contrast, transport-dominated problems – where coherent structures such as wave fronts or phase transitions travel through the spatial domain – can lead to solution manifolds that are rough in the sense that linearly approximating these solutions in a reduced space leads to a slow error decay with the reduced dimension (Rowley and Marsden 2000, Ohlberger and Rave 2013, Peherstorfer 2022). Linear model reduction is ineffective for such transport-dominated problems. For example, Ohlberger and Rave (2016), Greif and Urban (2019) and Arbes, Greif and Urban (2025) derive lower bounds on the error that behave as $1/\sqrt{n}$ when solution manifolds corresponding to instances of the linear advection and wave equation are approximated with linear approximations in reduced spaces of dimension n, a substantially slower error decay compared to the exponential decay obtained for some diffusion-dominated problems. This limitation is often referred to as the Kolmogorov barrier.

Besides these approximation-theoretic error bounds, the limitation of linear model reduction for transport- and wave-like problems has been empirically noted on a wide range of problems of interest in practice, including strongly advective and chemically reacting flows (Huang, Xu, Duraisamy and Merkle 2018, Huang and Duraisamy 2023), wildfire dynamics (Black, Schulze and Unger 2021*a*), pattern formation (Geelen and Willcox 2022), space weather predictions (Issan and Kramer 2023), pulsed detonation combustors (Schulze, Reiss and Mehrmann 2019), kinetic plasma

problems (Nicolini and Teixeira 2021) and rotating detonation waves (Mendible *et al.* 2020, Uy, Wentland, Huang and Peherstorfer 2024).

1.3. Nonlinear model reduction

Nonlinear model reduction aims to overcome the limited expressivity of linear model reduction by seeking nonlinear approximations of full-model solutions. Thus the reduced solution is allowed to depend nonlinearly on its degrees of freedom, which can provide a more efficient representation of full-model solutions than linear approximations in a reduced space. In other words, the trial manifold induced by the nonlinear approximations can represent the solution manifold accurately with fewer degrees of freedom than linear reduced-space approximations. From an approximation-theoretic perspective, allowing nonlinearity in the approximations can overcome the fundamental expressivity limitation of linear reduced-space approximations and so mitigate the Kolmogorov barrier. However, nonlinear approximations are just one ingredient of nonlinear model reduction. In contrast to linear model reduction, the reduced solutions given by the nonlinear approximations are no longer within a linear vector space. In particular, there is in general no global linear projection operator onto the trial manifold and no associated orthogonality principle, so standard Galerkin constructions and many standard stability arguments cannot be carried over verbatim from the full model. Nonlinear model reduction therefore requires revisiting the formulation of reduced models, their analysis, and numerical solution methods to account for the nonlinearity of the approximations, which is the topic of this survey.

Remark 1.2. Nonlinear approximations have a long history in numerical analysis, well predating recent nonlinear model reduction developments. For example, DeVore (1998) investigates the approximation-theoretic aspects of nonlinear approximations versus approximations in linear spaces as well as procedures to construct nonlinear approximations. Neural networks are investigated from an approximation-theoretic perspective in Pinkus (1999), Telgarsky (2016) and DeVore, Hanin and Petrova (2021). In finite element and related numerical methods for PDEs, adaptive mesh refinements lead to approximation spaces that are chosen in a solution-dependent way, so the resulting approximation is a nonlinear function of the underlying degrees of freedom (Babuvška and Rheinboldt 1978, Berger and Oliger 1984, Berger and Colella 1989). Likewise, approximations with rational functions are fundamentally nonlinear in their degrees of freedom, and this has been shown to be more effective than polynomial and fixed linear-space approximations for functions with sharp features or near-singular behaviour; see e.g. Braess (1986) and Trefethen (2019). While nonlinear model reduction can build on many of these advances, a key unique feature of nonlinear model reduction is that reduced models are not used to better approximate a single solution function but to build a nonlinear reduced model that can be repeatedly used for a range of parameters, inputs and initial conditions to accurately approximate full-model solutions.

1.4. Outline of the paper

After this introduction, we briefly discuss linear model reduction methods in Section 2 to set the stage. The limitation of linear model reduction to reduce transport- and wave-like problems is discussed in Section 3. Section 4 provides an overview of nonlinear model reduction and introduces nonlinear parametrizations, reduced dynamics and online solvers as the three elements of nonlinear model reduction. We then discuss three categories of nonlinear model reduction methods. First, methods based on nonlinear transformations in Section 5. Second, methods that adapt the reduced space online in Section 6. Third, methods that build on generic nonlinear parametrizations and employ instantaneous residual minimization in the online phase in Section 7. Conclusions and an outlook are given in Section 8.

2. Linear model reduction

This section briefly reviews linear model reduction techniques, which seek linear approximations of full-model solutions. Linear model reduction is typically separated into an offline phase, where a reduced model is constructed, and an online phase, where simulations are performed with the reduced model. We discuss the three steps of the offline phase of linear model reduction, which are (i) collecting training data, (ii) training a reduced space, and (iii) constructing a reduced model. We then discuss the lifting bottleneck, which hinders an efficient online phase, and present empirical interpolation as a way to circumvent the lifting bottleneck and so enable efficient online reduced-model solutions. We conclude with a numerical experiment that demonstrates the limited effectiveness of linear model reduction for transport-dominated problems.

2.1. Full model

We consider time- and parameter-dependent PDEs of the form

$$\partial_t q_{\mathcal{Q}}(t, \boldsymbol{x}; \boldsymbol{\mu}) = f(\boldsymbol{x}, q_{\mathcal{Q}}(t, \cdot; \boldsymbol{\mu}); \boldsymbol{\mu}), \tag{2.1}$$

where $t \in \mathcal{T} = [0, T]$ is the temporal variable, $\boldsymbol{x} \in \Omega \subseteq \mathbb{R}^d$ denotes the spatial variable, and $\boldsymbol{\mu} \in \mathcal{D} \subseteq \mathbb{R}^{d'}$ is a parameter. Problem (2.1) is equipped with an appropriate initial condition $q_{\mathcal{Q}}(0, \cdot; \boldsymbol{\mu}) \colon \Omega \to \mathbb{R}$ for all $\boldsymbol{\mu} \in \mathcal{D}$. The solution function $q_{\mathcal{Q}}(t, \cdot; \boldsymbol{\mu}) \colon \Omega \to \mathbb{R}$ is an element of a suitable Hilbert space $\mathcal{Q}$ that imposes boundary conditions so that the PDE problem is well-posed. In this work, we typically consider situations where the dimension d of the spatial domain is low, $d \in \{1, 2, 3\}$, and the dimension d' of the parameter is at most moderate. Note that several extensions to the PDE problem (2.1) are possible: the problem may also depend on a control input, the right-hand side function f may depend on time t, and the solution may be vector-valued, as in systems of PDEs. While model reduction methods exist for these settings, we restrict our attention to the formulation in (2.1) to ease exposition.

Let the solution function $q_{\mathcal{Q}}(t,\cdot;\boldsymbol{\mu})\colon \Omega \to \mathbb{R}$ be approximated in a finite-dimensional subspace $\mathcal{Q}_N \subseteq \mathcal{Q}$ at all times $t \in \mathcal{T}$ and parameters $\boldsymbol{\mu} \in \mathcal{D}$. The space $\mathcal{Q}_N$ has dimension $N \in \mathbb{N}$ and is spanned by the basis functions $\phi_1, \ldots, \phi_N : \Omega \to \mathbb{R}$. For example, the finite-dimensional space $\mathcal{Q}_N$ could be obtained with a finite element approach (Ern and Guermond 2004). An approximation of $q_{\mathcal{Q}}(t,\cdot;\boldsymbol{\mu})$ in $\mathcal{Q}_N$ is a linear combination

$$q(t,\boldsymbol{x};\boldsymbol{\mu}) = \sum_{i=1}^{N} q_i(t,\boldsymbol{\mu})\phi_i(\boldsymbol{x}), \tag{2.2}$$

with N coefficients $q_1(t,\boldsymbol{\mu}), \ldots, q_N(t,\boldsymbol{\mu}) \in \mathbb{R}$ that depend on time t and parameter $\boldsymbol{\mu}$. The ansatz (2.2) can then be inserted into a suitable weak (variational) formulation of the PDE problem (2.1) and (Petrov–) Galerkin conditions can be imposed so that the residual vanishes when tested against a test space. It is important to note that in model reduction it is typically assumed that the error of the solution (2.2) in $\mathcal{Q}_N$ compared to the corresponding solution $q_{\mathcal{Q}}$ in the space $\mathcal{Q}$ can be controlled with standard numerical analysis tools and is negligible for the purpose of model reduction. Thus, whenever we construct a reduced model via model reduction, it is sufficient to derive a reduced model that is accurate with respect to the solution q given in (2.2) in $\mathcal{Q}_N$. Correspondingly, we refer to the solution q in $\mathcal{Q}_N$ as the full-model solution.

Collecting the coefficients of the full-model solution (2.2) into

$$\boldsymbol{q}(t,\boldsymbol{\mu}) = \big[q_1(t,\boldsymbol{\mu}), \ldots, q_N(t,\boldsymbol{\mu})\big]^{\mathsf{T}} \in \mathbb{R}^N \tag{2.3}$$

gives a vector that describes a function $q(t,\cdot;\boldsymbol{\mu})$ in the space $\mathcal{Q}_N$. Such a coordinate-based representation $\boldsymbol{q}(t,\boldsymbol{\mu})$ of the full-model solution $q(t,\cdot;\boldsymbol{\mu})$ leads to the semi-discrete full model, which is a system of ordinary differential equations (ODEs),

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{q}(t,\boldsymbol{\mu}) &= \boldsymbol{f}(\boldsymbol{q}(t,\boldsymbol{\mu});\boldsymbol{\mu}), \\ \boldsymbol{q}(0;\boldsymbol{\mu}) &= \boldsymbol{q}_0(\boldsymbol{\mu}), \end{aligned} \tag{2.4}$$

with initial condition $\boldsymbol{q}_0(\boldsymbol{\mu}) \in \mathbb{R}^N$. Correspondingly, the vector $\boldsymbol{q}(t,\boldsymbol{\mu})$ is often referred to as the state of the semi-discrete full model. There could be a mass matrix resulting from the spatial discretization, which we formally include in the vector field $\boldsymbol{f}$. The mass matrix needs to be appropriately treated in actual implementations.

Remark 2.1. Throughout this survey we consider parametric full models without explicitly formulating an input–output map. Nevertheless, an important class of model reduction methods originates from systems and control theory, where the explicit definition of inputs and outputs is central. In that setting, the full model is viewed as an input–output dynamical system, and reduction can be posed in terms of approximating the induced input–output map. Representative approaches include balanced truncation, moment matching and interpolation, and related projection-based techniques, which provide reduced models that approximate the full system's

input–output response while often admitting stability and error guarantees, at least in the case of linear system dynamics. We refer to Antoulas (2005), Benner *et al.* (2015) and Antoulas *et al.* (2020) for overviews of system- and control-theoretic model reduction.

2.2. *Overview of linear model reduction*

Linear model reduction seeks approximations of the full-model solutions in a low-dimensional subspace $\mathcal{V} \subseteq \mathcal{Q}_N$ of dimension $n \ll N$.

The reduced space $\mathcal{V}$ has to be sufficiently expressive to well approximate the full-model solutions over the time interval $\mathcal{T}$ and the parameter domain $\mathcal{D}$. Formally, the full-model solutions induce a so-called solution manifold,

$$\mathcal{M} = \{q(t, \cdot; \boldsymbol{\mu}) \mid t \in \mathcal{T}, \boldsymbol{\mu} \in \mathcal{D}\}, \tag{2.5}$$

which is the set of all full-model solutions corresponding to the time and parameter range of interest. We follow the convention of referring to $\mathcal{M}$ as the solution manifold, even though the set $\mathcal{M}$ does not necessarily have a manifold structure; see e.g. Haasdonk (2017, Ex. 2.9). The solution manifold could be defined as the set of solutions in the space $\mathcal{Q}$ instead of the N-dimensional full-model space $\mathcal{Q}_N$.

A reduced solution is denoted as a function $\hat{q}\colon \mathcal{T} \times \Omega \times \mathcal{D} \to \mathbb{R}$. For a given t and $\boldsymbol{\mu}$, the function $\hat{q}(t, \cdot; \boldsymbol{\mu})\colon \Omega \to \mathbb{R}$ is an element of $\mathcal{V}$ and approximates the full-model solution $q(t, \cdot; \boldsymbol{\mu})$. Because $\hat{q}(t, \cdot; \boldsymbol{\mu})$ is in the reduced space $\mathcal{V}$, it can be represented as a linear combination

$$\hat{q}(t, \boldsymbol{x}; \boldsymbol{\mu}) = \sum_{i=1}^{n} \hat{q}_i(t, \boldsymbol{\mu}) \varphi_i(\boldsymbol{x}) \tag{2.6}$$

of basis functions $\varphi_1, \ldots, \varphi_n\colon \Omega \to \mathbb{R}$ of the reduced space $\mathcal{V}$. The coefficients $\hat{q}_1(t, \boldsymbol{\mu}), \ldots, \hat{q}_n(t, \boldsymbol{\mu}) \in \mathbb{R}$ of the linear combination (2.6) depend on time t and parameter $\boldsymbol{\mu}$. In contrast, the basis functions that span the space $\mathcal{V}$ are fixed over all times t and parameters $\boldsymbol{\mu}$. Thus, at all times t and parameters $\boldsymbol{\mu}$, the reduced solution (2.6) is a linear approximation in $\mathcal{V}$ in the sense that the coefficients (degrees of freedom) $\hat{q}_1(t, \boldsymbol{\mu}), \ldots, \hat{q}_n(t, \boldsymbol{\mu})$ enter linearly in $\hat{q}$ when $\hat{q}$ is interpreted as a function of the coefficients. The coefficients of the reduced solution (2.6) can be obtained via a variational formulation of the PDE problem (2.1) and Galerkin or Petrov–Galerkin projection onto $\mathcal{V}$, analogous to how the full-model solution (2.2) in $\mathcal{Q}_N$ is obtained from the PDE problem. In particular, the reduced solution $\hat{q}(t, \cdot; \boldsymbol{\mu})$ at all times t and parameters $\boldsymbol{\mu}$ is computed in $\mathcal{V}$ so that it depends only on n degrees of freedom, which are the n coefficients in the representation. This is in contrast to the full-model solution (2.2) that depends on N coefficients with typically $N \gg n$.

2.3. *The three steps of the offline phase of linear model reduction*

We identify three main steps of the offline phase of linear model reduction. Step 1 is collecting information about the full model, which is used to construct a suitable

reduced space $\mathcal{V}$ that can well approximate the full solutions lying on the solution manifold $\mathcal{M}$. Within the scope of this survey, the information is typically obtained in the form of full-model solutions at selected times and parameters, which constitute the training data. Step 2 is training a reduced space $\mathcal{V}$ on the training data to well approximate $\mathcal{M}$. Step 3 is constructing a reduced model that describes a system of equations for the coefficients of reduced solutions (2.6) in the reduced space $\mathcal{V}$. Once a reduced model has been constructed with these three steps in the offline phase, it can then be used in the online phase to compute reduced solutions within an outer-loop application. We will now discuss the three steps of the offline phase in more detail, and then the online phase in Section 2.4.

2.3.1. Offline Step 1: Collecting training data

A standard approach to collecting training data is to solve the full model (2.4) for the parameters in a set of training parameters $\mathcal{D}_{\text{train}} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_M\} \subseteq \mathcal{D}$, which results in a training data set

$$\mathcal{Q}_{\text{train}} = \{\boldsymbol{q}(t, \boldsymbol{\mu}_i) \mid i = 1, \ldots, M,\ t \in [0, T]\}, \tag{2.7}$$

where we use the semi-discrete representation given in (2.3) of full-model solutions.

It is challenging to select suitable training parameters $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_M$ so that the training data set $\mathcal{Q}_{\text{train}}$ is informative for constructing a reduced space. A large body of work addresses this problem by using greedy procedures that construct the training set incrementally: starting from an initial parameter sample, the reduced model is repeatedly evaluated over a candidate parameter set, an error indicator identifies the parameter where the reduced model performs worst, and this parameter is then added to the training set. This procedure is repeated until the error indicator is below a user-specified tolerance (Prud'homme *et al.* 2001, Veroy and Patera 2005, Haasdonk and Ohlberger 2008, Binev *et al.* 2011, Hesthaven, Stamm and Zhang 2014, Cohen, Dahmen, DeVore and Nichols 2020). Such greedy strategies often critically build on advances in *a posteriori* error estimation (Grepl and Patera 2005, Veroy and Patera 2005, Haasdonk and Ohlberger 2011, Urban and Patera 2012). Greedy methods can also explicitly target quantities of interest by constructing reduced spaces so that reduced solutions accurately approximate these quantities of interest; see e.g. Bui-Thanh, Willcox, Ghattas and van Bloemen Waanders (2007).

2.3.2. Offline Step 2: Training a reduced space

The training data $\mathcal{Q}_{\text{train}}$ from Offline Step 1 can be used to train a low-dimensional space $\mathcal{V}$ of dimension $n \ll N$, which is typically achieved by constructing basis functions $\varphi_1, \ldots, \varphi_n$ that span $\mathcal{V} \subseteq \mathcal{Q}_N$. Consequently, the basis functions $\varphi_1, \ldots, \varphi_n$ of $\mathcal{V}$ can be represented as linear combinations of the basis functions $\phi_1, \ldots, \phi_N$ that span the full-model solution space $\mathcal{Q}_N$,

$$\varphi_i = \sum_{j=1}^{N} v_i^{(j)} \phi_j, \quad i = 1, \ldots, n. \tag{2.8}$$

We collect the coefficients into the vectors

$$\boldsymbol{v}_i = \left[v_i^{(1)}, \ldots, v_i^{(N)}\right]^\top \in \mathbb{R}^N, \quad i = 1, \ldots, n.$$

The coefficient vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ span an n-dimensional subspace of $\mathbb{R}^N$. Thus we can represent the reduced space by the basis functions $\varphi_1, \ldots, \varphi_n$ that span a subspace of $\mathcal{Q}_N$ or, equivalently, by the basis vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ that span a subspace of $\mathbb{R}^N$.

One approach to constructing a basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$, and thus implicitly basis functions $\varphi_1, \ldots, \varphi_n$ of the form (2.8) that span $\mathcal{V}$, is given by the singular value decomposition (SVD). For this, consider K discrete time points $0 = t_0 < t_1 < \cdots < t_K = T$ and the corresponding snapshot matrix of the training data set $\mathcal{Q}_{\text{train}}$,

$$\boldsymbol{Q}_{\text{train}} = [\boldsymbol{q}(t_0, \boldsymbol{\mu}_1), \ldots, \boldsymbol{q}(t_K, \boldsymbol{\mu}_1), \boldsymbol{q}(t_0, \boldsymbol{\mu}_2), \ldots, \boldsymbol{q}(t_K, \boldsymbol{\mu}_M)] \in \mathbb{R}^{N \times (K+1)M}. \quad (2.9)$$

Computing the SVD of $\boldsymbol{Q}_{\text{train}}$ and using the first n left-singular vectors corresponding to the n largest singular values gives a basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ that spans a subspace of $\mathbb{R}^N$. In model reduction, this process of computing a basis matrix based on the SVD of the snapshot matrix is known as proper orthogonal decomposition (POD); see e.g. Sirovich (1987) or Holmes *et al.* (1996) for early usage in the context of fluid mechanics and Gubisch and Volkwein (2017) for a numerical analysis point of view. There is a wide range of techniques other than POD to construct reduced spaces, including greedy methods (Prud'homme *et al.* 2001, Veroy and Patera 2005, Bui-Thanh *et al.* 2007, Binev *et al.* 2011, Cohen *et al.* 2020), interpolation-based methods (Antoulas 2005, Baur *et al.* 2011, Antoulas *et al.* 2020) and balancing (Mullis and Roberts 1976, Moore 1981).

The outcome of Offline Step 2 is a basis of an n-dimensional reduced space $\mathcal{V}$, represented in terms of either the basis functions $\varphi_1, \ldots, \varphi_n \in \mathcal{Q}_N$ or the coefficient vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{R}^N$.

2.3.3. *Offline Step 3: Constructing a reduced model*

The third offline step of linear model reduction is constructing a reduced model, which means deriving a system of equations that determines the coefficients $\hat{q}_1(t, \boldsymbol{\mu}), \ldots, \hat{q}_n(t, \boldsymbol{\mu})$ of a reduced solution $\hat{q}$ as represented in (2.6).

Analogous to how the full model was derived, the reduced model can be obtained with a variational formulation and a Galerkin or Petrov–Galerkin projection, but now onto the reduced space $\mathcal{V}$ rather than the full-model space $\mathcal{Q}_N$. In the semi-discrete setting, the reduced model becomes a system of ODEs: the representation (2.8) allows us to identify a function $\hat{q}(t, \cdot; \boldsymbol{\mu}) \in \mathcal{V}$ as given in (2.6) with the coefficient vector

$$\hat{\boldsymbol{q}}(t, \boldsymbol{\mu}) = [\hat{q}_1(t, \boldsymbol{\mu}), \ldots, \hat{q}_n(t, \boldsymbol{\mu})]^\top \in \mathbb{R}^n, \quad (2.10)$$

which represents the coordinates of an element of the n-dimensional subspace of $\mathbb{R}^N$ that is spanned by the columns of the basis matrix $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n] \in \mathbb{R}^{N \times n}$. We note that in many cases the reduced basis vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ (and correspondingly the reduced basis functions $\varphi_1, \ldots, \varphi_n$) are constructed so that they are orthonormal.

In particular, in these cases the basis matrix $\boldsymbol{V}$ has orthonormal columns. If not otherwise noted, we assume in the following that $\boldsymbol{V}$ has orthonormal columns.

Building on the basis matrix $\boldsymbol{V}$ and the semi-discrete full model (2.4), the semi-discrete reduced model is

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{q}}(t,\boldsymbol{\mu}) = \hat{\boldsymbol{f}}(\hat{\boldsymbol{q}}(t,\boldsymbol{\mu});\boldsymbol{\mu}), \tag{2.11}$$

where the right-hand side function $\hat{\boldsymbol{f}}$ is obtained via Galerkin projection as

$$\hat{\boldsymbol{f}}(\hat{\boldsymbol{q}}(t,\boldsymbol{\mu});\boldsymbol{\mu}) = \boldsymbol{V}^{\top}\boldsymbol{f}(\boldsymbol{V}\hat{\boldsymbol{q}}(t,\boldsymbol{\mu});\boldsymbol{\mu}). \tag{2.12}$$

Analogously to the state $\boldsymbol{q}(t,\boldsymbol{\mu})$ of the semi-discrete full model, the vector $\hat{\boldsymbol{q}}(t,\boldsymbol{\mu})$ is often referred to as the reduced state. The initial condition $\hat{\boldsymbol{q}}(0;\boldsymbol{\mu}) \in \mathbb{R}^n$ can be obtained, for example, via the projection $\hat{\boldsymbol{q}}(0;\boldsymbol{\mu}) = \boldsymbol{V}^{\top}\boldsymbol{q}(0;\boldsymbol{\mu})$.

While we define the right-hand side function via Galerkin projection in (2.12), we note that Petrov–Galerkin projections, with test and trial spaces that are different, are essential for stability when the right-hand side function $\boldsymbol{f}$ stems from non-symmetric PDE operators (Quarteroni and Rozza 2014, Hesthaven *et al.* 2016, Mehrmann and Unger 2023). Model reduction methods based on interpolation also typically derive test and trial spaces that are different from each other so that the reduced model is obtained with Petrov–Galerkin projection (Antoulas 2005, Baur *et al.* 2011, Antoulas *et al.* 2020). Using oblique projections in a Petrov–Galerkin setting can also improve the predictive capabilities of reduced models, as demonstrated by Otto, Padovan and Rowley (2022) and Otto, Macchio and Rowley (2023). Improving robustness and stability is also one of the motivations of least-squares Petrov–Galerkin methods that were introduced by Carlberg, Bou-Mosleh and Farhat (2011) and Carlberg, Barone and Antil (2017). Dahmen, Plesken and Welper (2014), Brunken, Smetana and Urban (2019) and Engwer, Ohlberger and Renelt (2024) explicitly target transport-dominated problems and develop reduced models that remain stable for these problems; however, their emphasis is on robustness and stability of linear model reduction rather than overcoming the Kolmogorov barrier of linear approximations. A different approach is to combine linear model reduction formulations with stabilizing terms (e.g. Pacciarini and Rozza 2014) or to develop online stabilization techniques (e.g. Balajewicz, Tezaur and Dowell 2016, Maday, Manzoni and Quarteroni 2016).

It is important to notice that the right-hand side function $\hat{\boldsymbol{f}}$ of the reduced model (2.11) is not necessarily linear in the reduced state $\hat{\boldsymbol{q}}(t,\boldsymbol{\mu})$, even though $\hat{\boldsymbol{q}}(t,\boldsymbol{\mu})$ is a linear approximation of the full-model state $\boldsymbol{q}(t,\boldsymbol{\mu})$ in the n-dimensional subspace spanned by the columns of $\boldsymbol{V}$. This stresses once more that linear model reduction methods are applicable to nonlinear full models, and the term ‘linear’ in linear model reduction refers solely to the linear ansatz for the reduced solution.

Remark 2.2. One generally distinguishes between intrusive and non-intrusive model reduction. Non-intrusive methods learn both the reduced representation, such as the reduced space, and the reduced dynamics primarily from snapshot data; see

e.g. Ghattas and Willcox (2021) and Kramer *et al.* (2024). In contrast, in this survey we focus on intrusive methods that derive the reduced dynamics from the governing equations via projection or more general residual minimization techniques, for example. We briefly revisit non-intrusive model reduction in Section 8.

2.4. Online computations, the lifting bottleneck and empirical interpolation

We now discuss the online phase of linear model reduction, which concerns performing numerical simulations efficiently with the reduced model for a given new parameter and initial condition. In particular, we discuss the lifting bottleneck, which describes the issue that applying a projection directly to derive the reduced right-hand side function can still require online computations that scale unfavourably with the full-space dimension N, which can result in little-to-no speedup. We then briefly review empirical interpolation as a strategy to overcome the lifting bottleneck.

2.4.1. Online efficiency

In the online phase, the reduced model is used to perform simulations – also referred to as model evaluations – for new parameters $\boldsymbol{\mu}$ and initial conditions by integrating the reduced dynamical system (2.11) in time. This process requires repeated evaluations of the reduced right-hand side function $\hat{\boldsymbol{f}}$ and, for implicit time-integration schemes, typically also evaluations of derivatives of $\hat{\boldsymbol{f}}$ within Newton iterations, for example. Consequently, the computational costs of evaluating $\hat{\boldsymbol{f}}$ are typically the dominating part of the online costs.

We speak of online-efficient reduced models if the computational costs of the online phase scale independently of the dimension of the full model N (Rozza *et al.* 2008). A classical setting in which online efficiency can often be achieved is when the full-model right-hand side depends linearly on the state and affinely on the parameter, that is,

$$\boldsymbol{f}(\boldsymbol{q};\boldsymbol{\mu}) = \sum_{i=1}^{\ell} \rho_i(\boldsymbol{\mu}) \boldsymbol{A}_i \boldsymbol{q}, \quad \rho_i : \mathcal{D} \to \mathbb{R}, \quad \boldsymbol{A}_i \in \mathbb{R}^{N \times N}.$$

In this case the parameter-independent reduced operators $\boldsymbol{V}^\top \boldsymbol{A}_i \boldsymbol{V}$ can be pre-computed offline, while the assembly and evaluation of $\hat{\boldsymbol{f}}$ for a given parameter only incur costs independent of the full-model dimension N (Rozza *et al.* 2008, Quarteroni and Rozza 2014, Hesthaven *et al.* 2016). If we need to evaluate the reduced model only at a certain time instant or a small time window, then online efficiency can be further improved using contour integral methods; see e.g. Guglielmi, López-Fernández and Manucci (2021) and Guglielmi and Manucci (2023). However, except for such special structures, evaluating the reduced right-hand side $\hat{\boldsymbol{f}}$ generally still involves operations in the high-dimensional full-model space, which fundamentally limits online efficiency.

2.4.2. *Lifting bottleneck*

To see why online efficiency is difficult to achieve in general, recall that the reduced right-hand side $\hat{\boldsymbol{f}}$ in (2.12) is obtained from the full-model right-hand side $\boldsymbol{f}$ via (Petrov–) Galerkin projection. A direct evaluation of $\hat{\boldsymbol{f}}$ at a reduced state $\hat{\boldsymbol{q}}(t, \boldsymbol{\mu}) \in \mathbb{R}^n$ at time t and parameter $\boldsymbol{\mu}$ proceeds as follows: the reduced state is first lifted to the high-dimensional space as $\boldsymbol{V}\hat{\boldsymbol{q}}(t, \boldsymbol{\mu}) \in \mathbb{R}^N$, then the full-model right-hand side function $\boldsymbol{f}$ is evaluated at this lifted state $\boldsymbol{V}\hat{\boldsymbol{q}}(t, \boldsymbol{\mu})$, which requires evaluating all N component functions, and finally, the output of $\boldsymbol{f}$ is projected onto the reduced space spanned by the columns of $\boldsymbol{V}$.

In special cases, such as the affine and linear setting discussed above, the terms involved in the evaluation of $\hat{\boldsymbol{f}}$ can be pre-computed in the offline phase and then reused online (Rozza *et al.* 2008). However, in more general situations, the lifting of the reduced state, the evaluation of $\boldsymbol{f}$ in $\mathbb{R}^N$ and the subsequent projection imply that computing the reduced right-hand side $\hat{\boldsymbol{f}}$ incurs computational costs that scale with the full-model dimension N. The need to repeatedly lift quantities to the high-dimensional space and perform computations there during online evaluation is commonly referred to as the lifting bottleneck.

2.4.3. *Empirical interpolation*

The empirical interpolation method was introduced by Barrault, Maday, Nguyen and Patera (2004) to circumvent the lifting bottleneck. Empirical interpolation replaces the projection in (2.12) with interpolation, so that only a few of the N component functions of the full-model right-hand side function need to be evaluated. Several other methods have been proposed to circumvent the lifting bottleneck with similar interpolation and regression strategies. These include missing point estimation (Astrid, Weiland, Willcox and Backx 2004, 2008), empirical operator learning (Haasdonk, Ohlberger and Rozza 2008, Drohmann, Haasdonk and Ohlberger 2012), Gauss–Newton approximated tensors (Carlberg *et al.* 2011) and the energy-conserving sampling and weighting method that focuses on preserving structure and stability of underlying full-model discretizations (Farhat, Avery, Chapman and Cortial 2014, Farhat, Chapman and Avery 2015).

Steps of empirical interpolation. We focus here on the empirical interpolation method by Barrault *et al.* (2004), and in particular on its discrete counterpart, the discrete empirical interpolation method introduced by Chaturantabut and Sorensen (2010). In the offline phase, pairwise distinct interpolation points are selected,

$$p_1, \ldots, p_n \in \{1, \ldots, N\}, \tag{2.13}$$

which correspond in the discrete setting to indices of the N component functions of the full-model right-hand side function $\boldsymbol{f}$. The interpolation points give rise to the interpolation points matrix

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{e}_{p_1} & \cdots & \boldsymbol{e}_{p_n} \end{bmatrix} \in \mathbb{R}^{N \times n},$$

so that when applying a vector $\boldsymbol{z} = [z_1, \ldots, z_N]^\top \in \mathbb{R}^N$ to $\boldsymbol{P}^\top$, the components corresponding to indices $p_1, \ldots, p_n$ are selected:

$$\boldsymbol{P}^\top \boldsymbol{z} = \begin{bmatrix} z_{p_1} & \cdots & z_{p_n} \end{bmatrix}^\top.$$

In implementations, the interpolation points matrix $\boldsymbol{P}$ is not assembled, and instead its application to a vector is mimicked via slicing operations (Chaturantabut and Sorensen 2010). The reduced right-hand side function is then obtained by interpolating the full-model right-hand side $\boldsymbol{f}$ at the interpolation points $p_1, \ldots, p_n$ in the basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$, that is,

$$\tilde{\boldsymbol{f}}(\hat{\boldsymbol{q}}; \boldsymbol{\mu}) = (\boldsymbol{P}^\top \boldsymbol{V})^{-1} \boldsymbol{P}^\top \boldsymbol{f}(\boldsymbol{V}\hat{\boldsymbol{q}}; \boldsymbol{\mu}). \tag{2.14}$$

A reduced model based on empirical interpolation is analogous to the model given in (2.11), except that the reduced right-hand side $\tilde{\boldsymbol{f}}$ given in (2.14) is used, rather than the right-hand side $\hat{\boldsymbol{f}}$ defined in (2.12) via projection. We remark that empirical interpolation can also be applied to individual terms of the right-hand side function $\boldsymbol{f}$. For example, Barrault *et al.* (2004) and Chaturantabut and Sorensen (2010) apply empirical interpolation only to nonlinear terms of $\boldsymbol{f}$ while projecting the linear terms. Moreover, one may employ a separate reduced space for the approximation of the nonlinear terms, i.e. use distinct reduced spaces for the state and for the empirical interpolation approximation of the nonlinear terms of $\boldsymbol{f}$.

In the online phase, using $\tilde{\boldsymbol{f}}$ given via empirical interpolation (2.14) as right-hand side function in the reduced model can lead to runtime speedups even when using $\hat{\boldsymbol{f}}$ obtained via the projection (2.12) does not. If the full-model right-hand side function $\boldsymbol{f}$ satisfies some structural properties, for example, each component function of $\boldsymbol{f}$ can be evaluated independently of all other component functions, then computing the term $\boldsymbol{P}^\top \boldsymbol{f}(\boldsymbol{V}\hat{\boldsymbol{q}}(t, \boldsymbol{\mu}); \boldsymbol{\mu})$ of (2.14) can require evaluating only n instead of all N component functions of $\boldsymbol{f}$. Thus, in this case, empirical interpolation circumvents the lifting bottleneck. We refer to discussions by Barrault *et al.* (2004) and Chaturantabut and Sorensen (2010) for more details.

Selection of interpolation points. The selection of the interpolation points (2.13) is critical for the accuracy of empirical interpolation approximations. A range of methods has been developed for selecting the interpolation points. Barrault *et al.* (2004) and Chaturantabut and Sorensen (2010) propose greedy methods that sequentially add points based on the interpolation residual at each step. The QDEIM algorithm introduced by Drmač and Gugercin (2016) selects interpolation points based on the pivoting elements of the QR factorization of the transposed basis matrix $\boldsymbol{V}^\top$, which can lead to sharper error bounds than the ones corresponding to the greedy selection of the interpolation points; this QR-based sampling approach is related to max-volume submatrix selection (Drmač and Gugercin 2016).

Oversampling and empirical regression. While standard empirical interpolation enforces interpolation at exactly n sampling points, a common generalization is to relax this requirement by using more sampling points than the reduced-space

dimension. Selecting $n_s > n$ points $p_1, \ldots, p_n, p_{n+1}, \ldots, p_{n_s}$ leads to empirical regression,

$$\tilde{\boldsymbol{f}}(\hat{\boldsymbol{q}}; \boldsymbol{\mu}) = (\boldsymbol{P}^\top \boldsymbol{V})^+ \boldsymbol{P}^\top \boldsymbol{f}(\boldsymbol{V}\hat{\boldsymbol{q}}; \boldsymbol{\mu}),$$

where $(\boldsymbol{P}^\top \boldsymbol{V})^+$ denotes the Moore–Penrose pseudo-inverse of $\boldsymbol{P}^\top \boldsymbol{V}$. Taking $n_s > n$ points is referred to as oversampling and was used in the context of model reduction first by methods motivated by the gappy POD method (Everson and Sirovich 1995), such as Astrid *et al.* (2004, 2008) and Carlberg *et al.* (2011). One situation where oversampling is critical has been identified by Argaud *et al.* (2018), who show that adding noise or perturbations can lead to instabilities in empirical interpolation when using $n = n_s$ points. Peherstorfer, Drmač and Gugercin (2020) propose random oversampling, i.e. $n_s > n$, and prove that a randomized selection of points can mitigate this instability.

It is critical to note that neither the greedy algorithm introduced by Barrault *et al.* (2004) and Chaturantabut and Sorensen (2010) nor the QDEIM algorithm by Drmač and Gugercin (2016) can be directly applied to selecting $n_s > n$ points. Instead, a range of oversampling strategies are available (Astrid *et al.* 2008, Carlberg *et al.* 2011). One line of work focuses on minimizing the norm of the sampling operator $\boldsymbol{P}^\top \boldsymbol{V}$ and interprets adding a point as a low-rank update to the operator $\boldsymbol{P}^\top \boldsymbol{V}$ to derive selection criteria (Zimmermann and Willcox 2016, Peherstorfer *et al.* 2020).

2.5. *Linear model reduction for diffusion- and transport-dominated problems*

Let us now consider a numerical example to demonstrate when linear model reduction becomes inefficient for transport-dominated problems. The following presentation follows Peherstorfer (2022). We distinguish between diffusion- and transport-dominated problems.

2.5.1. *Diffusion problem: Heat equation*

As an example of a diffusion-dominated problem, consider the heat equation

$$\partial_t q_{\mathcal{Q}}(t, x; \mu) - \mu \partial_x^2 q_{\mathcal{Q}}(t, x; \mu) = 1, \quad x \in \Omega, \tag{2.15}$$

over the one-dimensional spatial domain $\Omega = [0, 1]$ with homogeneous Dirichlet boundary conditions. The initial condition is the constant function that evaluates to zero in Ω. The parameter μ corresponds to the heat-conductivity coefficient and is set to $\mu = 1$ in this example. Figure 2.1(a) shows a numerical solution of the heat equation (2.15) over time and space. The numerical solution is computed with linear finite elements on $N = 1024$ nodes (including boundary points) in Ω and implicit Euler in time with $K = 400$ time steps of time-step size 10^{-3} over the time interval $\mathcal{T} = [0, 0.4]$. Let us now collect snapshots (2.7), as in Offline Step 1 of linear model reduction outlined in Section 2.3.1. Recall that $\mu = 1$ is fixed. We take snapshots at the $K + 1$ discrete time points to form the snapshot matrix $\boldsymbol{Q}_{\text{train}}$ as defined in (2.9) and plot the normalized singular values in Figure 2.1(b). The

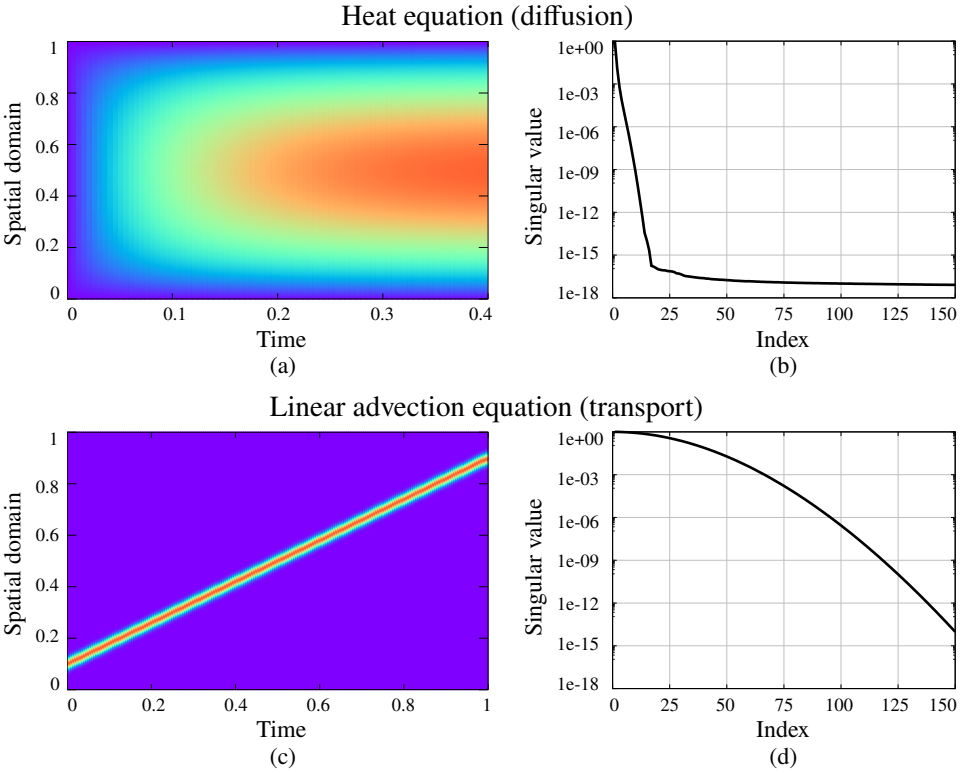

Figure 2.1. The snapshot matrix corresponding to the heat-equation example has rapidly decaying singular values, indicating that a low-dimensional subspace can capture the snapshot set efficiently. By contrast, the advection equation leads to a substantially slower decay, highlighting the limitations of linear reduced-space approximations for transport-dominated dynamics. First published in *Notices of the American Mathematical Society* **69** (2022), published by the American Mathematical Society. © 2022 American Mathematical Society.

normalized singular values are the ratios $\bar{\sigma}_i = \sigma_i/\sigma_1$ for $i = 1, \ldots, K+1$ so that the first normalized singular value is one.

The normalized singular values plotted in Figure 2.1(b) decay rapidly in the sense that the space spanned by the first $n = 20$ left-singular vectors approximates the snapshot data up to double precision in the relative Frobenius norm. The dimension of the snapshot vectors obtained from the full model is $N = 1024$ in this example (including the two boundary points), which is more than 50 times higher than the reduced dimension n. The fast decay of the singular values shows that the snapshots collected in this heat equation example can be well approximated with a low-dimensional space $\mathcal{V}$. While the singular values provide no guarantee about the approximation error of full solutions other than the ones in the snapshot matrix, the

singular values are still often used as a heuristic for how much reduction can be expected.

2.5.2. Transport problem: Linear advection equation

Let us now consider a canonical example of a transport problem, namely the linear advection equation,

$$\partial_t q_{\mathcal{Q}}(t,x;\mu) + \mu \partial_x q_{\mathcal{Q}}(t,x;\mu) = 0, \tag{2.16}$$

over the real line, where μ corresponds to the advection speed and is fixed to $\mu = 4/5$ in this example. The initial condition $q_{\mathcal{Q},0}\colon \mathbb{R} \to \mathbb{R}$ is a Gaussian probability density function with mean 10^{-1} and standard deviation 1.5×10^{-2}. The closed-form solution of the problem (2.16) is given by $(t,x) \mapsto q_{\mathcal{Q},0}(x - 4t/5)$ because we fixed $\mu = 4/5$. We evaluate the analytical solution at the $N = 1024$ equidistant points in $\Omega = [0,1]$ and take $K = 400$ equidistant time steps in $[0,1]$, leading to $K + 1$ snapshots in time. The snapshots are plotted in the time–space representation in Figure 2.1(c). The normalized singular values of the corresponding snapshot matrix are plotted in Figure 2.1(d).

The normalized singular values decay more slowly than in the case of the diffusion-dominated problem. In particular, the results appear to be consistent with an algebraic decay, in contrast to the rapid decay observed for the diffusion case. A space of dimension $n \geq 150$ is needed to represent the snapshot data up to double precision in the relative Frobenius norm, which yields a compression ratio N/n about one order of magnitude lower than for the diffusion problem. The slow decay is consistent with the geometric structure of the solution manifold for this transport problem: the dynamics only translate a localized feature (the Gaussian bump) through the domain. This means that the snapshots are shifted copies of the same function, and a vector space must use many basis functions to represent the same feature at different spatial locations. Thus the issue is not that the solution manifold is high-dimensional – in fact, in this example, it is one-dimensional because the parameter μ is fixed – but rather that it is inefficient to approximate this solution manifold by a low-dimensional vector space, as in linear model reduction.

In summary, this numerical illustration indicates that the efficiency of linear model reduction, as measured by the dimension of the reduced space required to reach a given accuracy, can be much lower for transport- than for diffusion-dominated problems.

3. Kolmogorov barrier

A central tool for understanding the performance limits of linear model reduction is the Kolmogorov n-width, which measures the best possible worst-case error achievable by approximations restricted to n-dimensional subspaces. In this section we briefly review this concept and discuss lower bounds on the n-width, which

show the fundamental limitations of linear model reduction methods for transport-dominated problems and motivate the notion of a Kolmogorov barrier. This section is intended to provide only a glimpse of the relevant approximation-theoretic concepts. For comprehensive treatments of the approximation-theoretic foundations of model reduction, we refer to Cohen and DeVore (2015, 2016), Benner *et al.* (2017) and Cohen, DeVore, Petrova and Wojtaszczyk (2022*b*).

3.1. Best linear approximations in Hilbert spaces: The Kolmogorov n-width

Let $\mathcal{Z}$ be a Hilbert space and let $\mathcal{N} \subseteq \mathcal{Z}$ be a subset. We might think of $\mathcal{Z}$ as the space over which the PDE problem (2.1) is defined or the finite-dimensional approximation space corresponding to the full model (2.4). Correspondingly, the subset $\mathcal{N}$ can be informally thought of as the set of all solutions to the PDE problem (2.1) or the full model (2.4), respectively. For a given n-dimensional subspace $\mathcal{Z}_n \subseteq \mathcal{Z}$, we define the deviation of $\mathcal{N}$ from $\mathcal{Z}_n$ as

$$d(\mathcal{N}, \mathcal{Z}_n) = \sup_{z \in \mathcal{N}} \inf_{z_n \in \mathcal{Z}_n} \|z - z_n\|, \tag{3.1}$$

which can be interpreted as the worst-case situation for the best approximation. Because $\mathcal{Z}$ is a Hilbert space, the infimum is obtained by the orthogonal projection of z onto $\mathcal{Z}_n$, which always exists as $\mathcal{Z}_n$ is of finite dimension n. Furthermore, the projection map is linear, which means that restricting best approximations to lie in a subspace of a Hilbert space leads to linear (subspace-based) approximations.

Let us now take the infimum of (3.1) over all n-dimensional subspaces of $\mathcal{Z}$ and so define the Kolmogorov n-width

$$d_n(\mathcal{N}, \mathcal{Z}) = \inf_{\substack{\mathcal{Z}_n \subseteq \mathcal{Z} \\ \dim(\mathcal{Z}_n) \leq n}} d(\mathcal{N}, \mathcal{Z}_n) = \inf_{\substack{\mathcal{Z}_n \subseteq \mathcal{Z} \\ \dim(\mathcal{Z}_n) \leq n}} \sup_{z \in \mathcal{N}} \inf_{z_n \in \mathcal{Z}_n} \|z - z_n\|, \tag{3.2}$$

an idea first proposed in Kolmogoroff (1936). Following Pinkus (1985), an n-dimensional subspace $\mathcal{Z}_n^\star$ of $\mathcal{Z}$ is called an optimal subspace for $d_n(\mathcal{N}, \mathcal{Z})$ if

$$d_n(\mathcal{N}, \mathcal{Z}) = d(\mathcal{N}, \mathcal{Z}_n^\star).$$

3.2. Fast and slow decay of Kolmogorov n-width and Kolmogorov barrier

Let us now apply the concept of the Kolmogorov n-width to the setting of model reduction. As mentioned above, we might think of $\mathcal{Z}$ as the space $\mathcal{Q}$ over which the PDE problem (2.1) is defined and of $\mathcal{N}$ as the set of all solutions. Analogously, we can think of $\mathcal{Z}$ as the finite-dimensional approximation space $\mathcal{Q}_N$ corresponding to the full model (2.4) and of the subset $\mathcal{N}$ as the set of all solutions (2.5) to the full model.

Under this setting, the Kolmogorov n-width (3.2) is the smallest worst-case error that can be achieved when approximating the solutions in an n-dimensional reduced space. Consequently, how fast or slow the Kolmogorov n-width decays with the dimension n of the reduced space has major implications on the efficiency of linear

model reduction. If it decays quickly, then a reduced low-dimensional space is often sufficient to meet a given error tolerance. Thus, in such a situation, linear model reduction can be efficient, at least from an approximation-theoretic perspective. In contrast, if the Kolmogorov n-width decays slowly, then the dimension of the reduced space to meet the error tolerance might be so high that linear model reduction becomes inefficient. In this case we speak of the Kolmogorov barrier, because it presents a hard limitation of linear model reduction.

Let us consider examples from the model reduction literature to clarify what fast and slow decay means in this context. Maday *et al.* (2002) consider elliptic coercive PDEs over a one-dimensional parameter μ and the corresponding set of solutions. Maday *et al.* (2002) show that in this situation there is an upper bound on the Kolmogorov n-width that decays exponentially fast in the dimension n of the reduced space. Thus, increasing the dimension of the reduced space in linear model reduction can achieve an exponential error reduction. Analogous results of a rapid, albeit not necessarily exponential, decay for more general elliptic problems are shown in Cohen, DeVore and Schwab (2011), Cohen and DeVore (2015, 2016) and Bachmayr and Cohen (2017). As a remark, we add that beyond existence results, Binev *et al.* (2011), Buffa *et al.* (2012) and Cohen *et al.* (2020) show that sequences of reduced spaces can be constructed with greedy methods, so that the same decay up to constants as that of the n-width is attained.

Broadly speaking, diffusion-dominated problems, such as the one discussed in Section 2.5, often fall into this category for which the Kolmogorov n-width decays quickly and thus for which linear model reduction is well suited.

3.3. Slow n-width decay for linear advection problems

Let us now consider examples of PDE problems for which lower bounds on the Kolmogorov n-width of solution manifolds have been shown.

Let us first consider the linear advection problem over the one-dimensional spatial domain $\Omega = [0, 1]$ and time domain $[0, 1]$ in $\mathcal{Q} = L^2([0, 1])$,

$$\partial_t q_{\mathcal{Q}}(t, x; \mu) + \mu \partial_x q_{\mathcal{Q}}(t, x; \mu) = 0, \tag{3.3}$$

with the conditions

$$q_{\mathcal{Q}}(0, x; \mu) = 0, \quad q_{\mathcal{Q}}(t, 0; \mu) = 1,$$

for all $\mu \in [0, 1]$. Let us consider the corresponding solution manifold for just $\mu = 1$,

$$\mathcal{M} = \{q_{\mathcal{Q}}(t, \cdot; \mu) \mid t \in [0, 1], \mu = 1\}, \tag{3.4}$$

which consists of the step functions

$$q_{\mathcal{Q}}(t, x; \mu) = \begin{cases} 1, & 0 \le x \le t\mu, \\ 0, & t\mu < x \le 1, \end{cases} \tag{3.5}$$

that can have a discontinuity anywhere in the spatial domain $[0, 1]$. Notice that we

use the set of solutions $\mathcal{M}$ not over a finite-dimensional (full-model) space $\mathcal{Q}_N$ here as in (2.5) but over $\mathcal{Q} = L^2([0, 1])$. Ohlberger and Rave (2016) show that the Kolmogorov n-width with respect to the L^2-norm is lower-bounded as

$$d_n(\mathcal{M}, \mathcal{Q}) \geq c\frac{1}{\sqrt{n}},$$

where n is the dimension of the reduced space and $c > 0$ is a constant.

We can interpret the bound as showing that there cannot exist a sequence of subspaces of L^2 that achieves a faster approximation error in the sense of the Kolmogorov n-width (3.2) than $1/\sqrt{n}$, which is a substantially slower error decay than the exponential error decay that can be achieved for the one-parameter diffusion-dominated problem discussed in Section 3.2. This result is also in agreement with the numerical experiments shown in Section 2.5, where the example based on the linear advection equation empirically exhibits a slower decay of the projection error given by the singular values, whereas the diffusion-dominated problem given by the instance of the heat equation leads to an exponential decay of the singular values. We stress once more that the decay of the singular values does not directly correspond to the decay of the Kolmogorov n-width. However, the singular values can provide an empirical indication under certain conditions; see Section 2.5.

Arbes *et al.* (2025) show that if the initial condition in the linear advection example is chosen to be smoother, then this smoothness is also reflected in the decay rate of the Kolmogorov n-width. Analogous results for the wave equation are shown in Greif and Urban (2019).

Overall, these approximation-theoretic results, as well as the empirical evidence discussed in Section 2.5 and the cited literature in Section 1.2, suggest that transport-dominated problems with wave-like phenomena and moving coherent, localized features lead to slowly decaying n-widths, which fundamentally limit linear model reduction techniques.

Remark 3.1. In this survey we focus on time-dependent problems; however, even for stationary problems, linear model reduction can become inefficient due to the Kolmogorov barrier. For example, Welper (2017), Taddei (2020) and Alireza Mirhoseini and Zahr (2023) consider problems in which the parameter $\boldsymbol{\mu}$ induces the transport of a coherent structure in the spatial domain.

3.4. Kolmogorov n-width and linear control systems

Throughout this survey we focus on a finite-dimensional parameter space. To complement the picture, we present here a particular example with an infinite-dimensional parameter space, given by linear, finite-dimensional, time-invariant control systems of the form

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{q}(t) &= \boldsymbol{A}\boldsymbol{q}(t) + \boldsymbol{B}\boldsymbol{u}(t), \\ \boldsymbol{y}(t) &= \boldsymbol{C}\boldsymbol{q}(t), \end{aligned} \tag{3.6}$$

with matrices $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, $\boldsymbol{B} \in \mathbb{R}^{N\times N_{\text{in}}}$ and $\boldsymbol{C} \in \mathbb{R}^{N_{\text{out}}\times N}$. The symbols $\boldsymbol{u}$ and $\boldsymbol{y}$ denote the input and output of the system, respectively, and the input $\boldsymbol{u} \in L^2(-\infty, 0; \mathbb{R}^{N_{\text{in}}})$ takes the role of the parameter. We assume that the control system (3.6) is asymptotically stable, that is, the eigenvalues of the matrix $\boldsymbol{A}$ are contained in the open left-half complex plane. In terms of model reduction, we focus on the approximation of the input-to-output operator

$$\mathcal{H}\colon L^2(-\infty, 0; \mathbb{R}^{N_{\text{in}}}) \to L^2(0, \infty; \mathbb{R}^{N_{\text{out}}}), \quad \boldsymbol{u} \mapsto \boldsymbol{y}, \tag{3.7}$$

which maps past inputs to future outputs. In the literature, the operator $\mathcal{H}$ is known as the Hankel operator. Introducing the impulse response $\boldsymbol{h}(t) = \boldsymbol{C}\exp(t\boldsymbol{A})\boldsymbol{B}$, the Hankel operator is given via the convolution integral

$$(\mathcal{H}\boldsymbol{u})(t) = \int_{-\infty}^{0} \boldsymbol{h}(t-s)\boldsymbol{u}(s)\,\mathrm{d}s.$$

It is well known that $\mathcal{H}$ is a finite-rank operator of rank at most N; see Antoulas (2005, Section 5.4) or Francis (1987, Section 5.1). Consequently, $\mathcal{H}$ is a compact operator and its singular values, commonly known as the Hankel singular values, can be computed as the square roots of the eigenvalues of the matrix product $\mathcal{P}_{\text{c}}\mathcal{P}_{\text{o}}$, where $\mathcal{P}_{\text{c}}, \mathcal{P}_{\text{o}} \in \mathbb{R}^{N\times N}$ are the unique solutions of the Lyapunov equations

$$\boldsymbol{A}\mathcal{P}_{\text{c}} + \mathcal{P}_{\text{c}}\boldsymbol{A}^{\top} + \boldsymbol{B}\boldsymbol{B}^{\top} = 0 \quad \text{and} \quad \boldsymbol{A}^{\top}\mathcal{P}_{\text{o}} + \mathcal{P}_{\text{o}}\boldsymbol{A} + \boldsymbol{C}^{\top}C = 0. \tag{3.8}$$

We sort the Hankel singular values in decreasing order and denote the ith Hankel singular value by $\sigma_i(\mathcal{H})$ for $i = 1, \ldots, N$. For $\mathcal{U} = L^2(-\infty, 0; \mathbb{R}^{N_{\text{in}}})$ and $\mathcal{Y} = L^2(0, \infty; \mathbb{R}^{N_{\text{out}}})$, Unger and Gugercin (2019) prove that the Kolmogorov n-widths coincide with the Hankel singular values in the sense that

$$d_n(\mathcal{H}(\{\boldsymbol{u} \in \mathcal{U} \mid \|\boldsymbol{u}\|_{\mathcal{U}} \le 1\}), \mathcal{Y}) = \sigma_{n+1}(\mathcal{H}). \tag{3.9}$$

We emphasize that in contrast to the previous results, (3.9) is not a lower or upper bound, but an exact identity that can be explicitly computed. While the computation of the n-widths, i.e. the computation of the Hankel singular values, is computationally well explored and feasible in the large-scale case (see Simoncini 2016 for a survey), the computation of an explicit reduced model of the form

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{q}}(t) &= \hat{\boldsymbol{A}}\hat{\boldsymbol{q}}(t) + \hat{\boldsymbol{B}}\boldsymbol{u}(t), \\ \hat{\boldsymbol{y}}(t) &= \hat{\boldsymbol{C}}\hat{\boldsymbol{q}}(t), \end{aligned}$$

that realizes this approximation quality is a non-trivial task. It is known in the literature as optimal Hankel norm approximation, and was first solved by Glover (1984), relying on results from Adamyan, Arov and Krein (1971). Extensions to descriptor systems are presented in Cao, Saltik and Weiland (2015) and Benner and Werner (2020).

While the spaces for the solution operator in (3.7) allow for a precise analytical treatment, in practical applications one is often interested in the solution operators

$$\mathcal{S}_1 \colon L^2(0,\infty;\mathbb{R}^{N_{\text{in}}}) \to L^2(0,\infty;\mathbb{R}^{N_{\text{out}}}), \quad \boldsymbol{u} \mapsto \boldsymbol{y},$$
$$\mathcal{S}_2 \colon L^2(0,\infty;\mathbb{R}^{N_{\text{in}}}) \to L^\infty(0,\infty;\mathbb{R}^{N_{\text{out}}}), \quad \boldsymbol{u} \mapsto \boldsymbol{y},$$

which correspond to approximations in the Hardy $\mathcal{H}_\infty$- and $\mathcal{H}_2$-norms. Nevertheless, studying the Hankel operator is of great interest, as the famous *a priori* error bound of balanced truncation is a bound on the $\mathcal{H}_\infty$-norm in terms of the truncated Hankel singular values.

We conclude this short excursion to linear control systems with four remarks. First, the analysis in Unger and Gugercin (2019) demonstrates that the Kolmogorov n-widths for control systems can be interpreted as the dual concept to the method of active subspaces developed by Constantine (2015). Second, if the matrices $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ have a parameter dependence with a parameter $\boldsymbol{\mu}$ in a compact parameter set $\mathcal{D}$, then Unger and Gugercin (2019, Theorem 3) prove that the Kolmogorov n-widths are lower-bounded by the maximum of the Hankel singular values over the parameter set. Consequently, one can go beyond the Kolmogorov n-width by allowing parameter-dependent projection matrices, as demonstrated in Wittmuess *et al.* (2016), Gosea, Gugercin and Unger (2021) and Hund, Mitchell, Mlinarić and Saak (2022). Third, an extension to bilinear control systems is presented in Zuyev, Feng and Benner (2024). Fourth, the Kolmogorov n-widths critically depend on the space $\mathcal{Z}$ and the associated norm. In view of transport-dominated problems and associated methods, the standard L_p-norms may paint an incomplete picture, and additional metrics such as a Wasserstein distance may provide different insights (Ehrlacher, Lombardi, Mula and Vialard 2020); see also Section 5.2.2.

4. The elements of nonlinear model reduction

Nonlinear model reduction aims to overcome the Kolmogorov barrier by seeking nonlinear instead of linear approximations of full-model solutions.

4.1. Overcoming the Kolmogorov barrier with nonlinear parametrizations

We now introduce what we mean by a parametrization and then discuss that nonlinear parametrizations can lead to nonlinear approximations that overcome the Kolmogorov barrier.

4.1.1. Parametrizations

A parametrization is a map with signature

$$\hat{q} \colon \mathbb{R}^n \times \Omega \to \mathbb{R}, \quad (\boldsymbol{\theta}, \boldsymbol{x}) \mapsto \hat{q}(\boldsymbol{\theta}, \boldsymbol{x}), \tag{4.1}$$

so that for an n-dimensional weight vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_n]^\top \in \mathbb{R}^n$, the function $\hat{q}(\boldsymbol{\theta}, \cdot) \colon \Omega \to \mathbb{R}$ is an element $\hat{q}(\boldsymbol{\theta}, \cdot) \in \mathcal{Z}$ of a function space of interest $\mathcal{Z}$. For

example, $\mathcal{Z}$ could be the space $\mathcal{Q}$ over which we defined the PDE problem (2.1) or the space $\mathcal{Q}_N$ over which the full-model solutions (2.2) are defined. More precisely, one could first define a map $\hat{Q}\colon \mathbb{R}^n \to \mathcal{Z}$ to obtain $\hat{Q}(\boldsymbol{\theta}) \in \mathcal{Z}$, which can then, whenever elements of $\mathcal{Z}$ admit pointwise evaluation, be evaluated at points in the spatial domain Ω over which $\mathcal{Z}$ is defined as $\hat{Q}(\boldsymbol{\theta})(\boldsymbol{x})$. It will be convenient for us to work directly with (4.1) and write $\boldsymbol{\theta} \mapsto \hat{q}(\boldsymbol{\theta}, \cdot)$ if we mean $\hat{Q}$. At this point, we do not specify how to obtain a weight vector $\boldsymbol{\theta}$ for approximating an element of a function space of interest. A map that produces a weight vector for a given function is sometimes referred to as an encoder map (and correspondingly $\boldsymbol{\theta} \mapsto \hat{q}(\boldsymbol{\theta}, \cdot)$ is then referred to as a decoder map). In model reduction, however, we typically do not have access to the functions that we would like to encode in the online phase because these are the full-model solutions. Instead, the weight vectors have to be computed via reduced dynamics in the online phase; see the forthcoming Section 4.4.

We call a parametrization (4.1) linear (or affine) if it depends linearly (or affinely) on the weight vector $\boldsymbol{\theta}$, that is, if the map $\boldsymbol{\theta} \mapsto \hat{q}(\boldsymbol{\theta}, \cdot)$ is linear (or affine). It is important to note that linearity refers to the dependence on $\boldsymbol{\theta}$ and not $\boldsymbol{x}$, that is, the map $\hat{q}(\boldsymbol{\theta}, \cdot)\colon \Omega \to \mathbb{R}, \boldsymbol{x} \mapsto \hat{q}(\boldsymbol{\theta}, \boldsymbol{x})$ can still be nonlinear in the spatial coordinate $\boldsymbol{x}$ for all $\boldsymbol{\theta} \in \mathbb{R}^n$. Furthermore, the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ can depend nonlinearly on quantities such as time t and parameter $\boldsymbol{\mu}$.

Remark 4.1. The vector $\boldsymbol{\theta}$ is often called the parameter vector; however, we follow the convention in model reduction of referring to $\boldsymbol{\mu}$ as the parameter, and have thus opted to call $\boldsymbol{\theta}$ a weight vector, in preparation for nonlinear parametrizations such as neural networks. In the context of model reduction, the weight vector $\boldsymbol{\theta}$ is also referred to as the reduced or latent state because, as we will see in later sections, it can represent the reduced solution.

4.1.2. Linear parametrizations and linear model reduction

Recall the reduced ansatz in linear model reduction given by the linear combination (2.6). This ansatz can equivalently be written as a function of the coefficients (or reduced state (2.10)) as

$$\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \boldsymbol{x}) = \sum_{i=1}^{n} \theta_i(t, \boldsymbol{\mu}) \varphi_i(\boldsymbol{x}), \tag{4.2}$$

which makes the linear dependence on the coefficients explicit. Notice that we identify in (4.2) the coefficients of the linear combination (2.6) with n weights $\theta_1(t, \boldsymbol{\mu}), \ldots, \theta_n(t, \boldsymbol{\mu})$, which are collected into the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ and depend on time t and parameter $\boldsymbol{\mu}$. From this perspective, we can interpret the reduced ansatz (2.6) as a linear parametrization (4.2) of the reduced space $\mathcal{V}$, which is spanned by the reduced basis functions $\varphi_1, \ldots, \varphi_n$, with the n-dimensional weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$. In other words, linear model reduction represents reduced solutions with linear parametrizations.

The set of all reduced solutions that can be represented by the linear parametrization (4.2) is the reduced space $\mathcal{V}$, because for every $\boldsymbol{\theta} \in \mathbb{R}^n$ we have $\hat{q}(\boldsymbol{\theta}, \cdot) \in \mathcal{V}$, and conversely every element in $\mathcal{V}$ can be written as $\hat{q}(\boldsymbol{\theta}, \cdot)$ for some $\boldsymbol{\theta} \in \mathbb{R}^n$. The degrees of freedom of a reduced solution are the weights in the vector $\boldsymbol{\theta}$, which can depend on time t and parameter $\boldsymbol{\mu}$, but the reduced space $\mathcal{V}$ given by the basis $\varphi_1, \ldots, \varphi_n$ remains fixed and independent of time and parameter.

4.1.3. Nonlinear parametrizations

Let us now consider nonlinear parametrizations, i.e. parametrizations that are nonlinear in the weight vector $\boldsymbol{\theta}$. Given a nonlinear parametrization, we can consider the set of all functions that can be reached by varying the weight $\boldsymbol{\theta}$ as

$$\widehat{\mathcal{M}} = \{\hat{q}(\boldsymbol{\theta}, \cdot) \mid \boldsymbol{\theta} \in \mathbb{R}^n\}, \tag{4.3}$$

which is often referred to as trial set or trial manifold. While linear parametrizations lead to sets $\widehat{\mathcal{M}}$ with a vector space structure, nonlinear parametrizations correspond to more general trial sets that are not necessarily vector spaces and may therefore represent target solutions more efficiently with respect to the number of degrees of freedom n. Broadly speaking, the nonlinear dependence on the weight vector $\boldsymbol{\theta}$ means that nonlinear parametrizations may circumvent the Kolmogorov barrier because the induced set of functions $\widehat{\mathcal{M}}$ does not necessarily have a vector space structure any longer.

We can also interpret the nonlinear dependence on the weight vector $\boldsymbol{\theta}$ as allowing the representation itself to change. To illustrate this point, consider the parametrization

$$\hat{q}(\boldsymbol{\theta}, \boldsymbol{x}) = \sum_{i=1}^{r} \beta_i \psi_i(\alpha_i, \boldsymbol{x}). \tag{4.4}$$

Here, the weight vector $\boldsymbol{\theta} = [\boldsymbol{\alpha}^\top, \boldsymbol{\beta}^\top]^\top \in \mathbb{R}^n$ with $n = 2r$ consists of r weights $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_r]^\top \in \mathbb{R}^r$ that enter nonlinearly through the representation functions $\psi_1, \ldots, \psi_r \colon \mathbb{R} \times \Omega \to \mathbb{R}$, and r weights $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_r]^\top \in \mathbb{R}^r$ that enter linearly. Unlike linear parametrizations, where the representation is fixed by the basis functions $\varphi_1, \ldots, \varphi_n$, the weights $\alpha_1, \ldots, \alpha_r$ in (4.4) directly influence the functions $\psi_1(\alpha_1, \cdot), \ldots, \psi_r(\alpha_r, \cdot)$. As a result, the representation adapts to the specific function being approximated. For a full-model solution $q(t, \cdot; \boldsymbol{\mu}) \in \mathcal{M}$, the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ may therefore change not only the coefficients of a fixed expansion, but also the shape of the underlying representation itself. This is fundamentally different from linear model reduction, where the representation corresponds to the basis functions $\varphi_1, \ldots, \varphi_n$ that span a vector space, which is fixed in advance, and the weights in the linear parametrization only determine the coefficients of the associated linear combination.

Remark 4.2. There is an analogue to linear versus nonlinear parametrizations in machine learning. Linear parametrizations typically correspond to kernel

approximations (Schölkopf and Smola 2001). Once a kernel is picked, the learning problem reduces to finding a linear combination of the kernels, which means that the representation is fixed and the coefficients (weights) of approximations in the kernel space enter linearly. In contrast, nonlinear parametrizations are given, for example, by neural networks with at least one hidden layer and nonlinear activation functions (LeCun, Bengio and Hinton 2015, Goodfellow, Bengio and Courville 2016). The inner layers correspond to the representation and are adapted with the weights. In other words, the network adapts the basis functions themselves, which is called representation learning (Rumelhart, Hinton and Williams 1986, Bengio, Courville and Vincent 2013).

Remark 4.3. In addition to parametrizations in which the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ evolves in time, one can also consider global-in-time parametrizations, in which time is treated as another input and the weight vectors do not depend on time; this is analogous in spirit to space–time discretizations. We do not pursue global-in-time parametrizations here and instead focus on sequential- or local-in-time parametrizations, where the time dependence enters via the weight vector, which aligns with the widely used model reduction paradigm in which reduced states are advanced by reduced dynamical systems. We refer the reader to Zhang, Chen, Vanden-Eijnden and Peherstorfer (2024) for a discussion of global-in-time versus local-in-time (nonlinear) parametrizations.

4.1.4. Example: Nonlinear parametrizations and linear advection

Let us revisit the linear advection equation problem of Section 3.3. The corresponding solution manifold consists of the step functions given in (3.5). We view the step functions (3.5) as the restriction to $[0, 1]$ of a translated reference profile,

$$q_{\mathcal{Q}}(t, x; \mu) = q_0(x - t\mu), \tag{4.5}$$

where $q_0 : \mathbb{R} \to \mathbb{R}$ is defined as

$$q_0(x) = \begin{cases} 1, & x \leq 0, \\ 0, & x > 0. \end{cases}$$

Representing the solution to the linear advection problem as (4.5) reveals a key difficulty: the solution can be written as a composition of the translation $(t, x, \mu) \mapsto x - t\mu$ and the initial function q_0. A linear parametrization (4.2) can efficiently represent superpositions of features because it is a linear combination of basis functions. However, representing a translation of a feature via superposition means that the feature must be deactivated (low weight in the superposition) at its old spatial location and activated (high weight in the superposition) at its new spatial location – a process that can require a prohibitively large number of basis functions to describe a smooth motion of the feature.

With a nonlinear parametrization such as the one defined in (4.4), translation can be described more efficiently by composing a translation with the superposition of

features. To be concrete, let us derive a nonlinear parametrization of the form (4.4) that can exactly represent the step function (4.5). One instance is

$$\hat{q}(\boldsymbol{\theta}(t,\mu),x) = \beta_1(t,\mu)\psi_1(\alpha_1(t,\mu),x), \tag{4.6}$$

where $\beta_1(t,\mu) = 1$ and $\alpha_1(t,\mu) = t\mu$, yielding the weight vector $\boldsymbol{\theta}(t,\mu) = [\alpha_1(t,\mu),\beta_1(t,\mu)]^\top$. The function ψ_1 is given as

$$\psi_1(\alpha,x) = q_0(x-\alpha).$$

Thus the step functions (4.5), which include all functions on the solution manifold (3.4) of the linear advection problem in Section 3.3, can be exactly represented by the nonlinear parametrization (4.4) with only $n = 2$ weights. In contrast, the error of using a linear parametrization (4.2) cannot decay faster than with a rate $1/\sqrt{n}$ in L^2, as discussed in Section 3.3.

Even though this example is synthetic, it already shows that with a suitable choice of a nonlinear parametrization, phenomena such as transport and superposition of features can be composed, instead of having to approximate all phenomena with only a superposition of features. This additional flexibility can lead to more efficient parametrizations in terms of the number of weights.

4.2. *Extensions of Kolmogorov n-width to nonlinear parametrizations*

There are several generalizations of the Kolmogorov n-width concepts to nonlinear parametrizations. One widely used concept builds on an autoencoder-like structure. An encoder maps an element of the set of functions of interest $\mathcal{N} \subseteq \mathcal{Z}$ to an n-dimensional vector, i.e. the weight $\boldsymbol{\theta}$ in our terminology. A decoder maps the n-dimensional vector back onto a function in $\mathcal{Z}$. One is then interested in the worst-case approximation error of the composition of the encoder and decoder and its decay with the dimension n; see e.g. DeVore, Kyriazis, Leviatan and Tikhomirov (1993) and DeVore (1998). We revisit autoencoders from a numerical perspective in Section 7.

The existence of an encoder–decoder pair that achieves a low approximation error is insufficient for meaningful numerical computations. The regularity of the encoder and decoder map is essential, because it excludes pathological parametrizations and so relates to numerical stability. This motivates the stable manifold widths considered in Cohen *et al.* (2022*b*), which ask for at least Lipschitz bounds on the encoder and decoder.

Cohen, Farhat, Maday and Somacal (2024) further restrict the encoder to consist of n linear functionals of the solution function that is to be encoded. The decoder can remain nonlinear. The expressivity of such nonlinear approximations is quantified by the so-called sensing numbers. In particular, Cohen *et al.* (2024) discuss examples of step functions that are transported over time for which the error of such approximations decays faster than the error achieved with linear approximations. Indeed, it is shown in Cohen *et al.* (2022*b*, 2024) that the sensing numbers and the

stable manifold width attain the same decay rates. Numerical experiments related to sensing numbers are presented in Glas and Unger (2025).

4.3. *Nonlinear parametrizations with offline and online weights*

There is a wide range of nonlinear parametrizations used in numerical analysis, machine learning and computational science and engineering, which also motivates nonlinear parametrizations for model reduction. We provide only a brief overview in this section and revisit them in detail in later sections of this survey.

4.3.1. *Offline and online weights*

We now consider parametrizations that can depend on an offline weight vector $\boldsymbol{\theta}_{\text{off}} \in \mathbb{R}^P$,

$$\hat{q}\colon \mathbb{R}^n \times \Omega \times \mathbb{R}^P \to \mathbb{R}, \quad (\boldsymbol{\theta}, \boldsymbol{x}; \boldsymbol{\theta}_{\text{off}}) \mapsto \hat{q}(\boldsymbol{\theta}, \boldsymbol{x}; \boldsymbol{\theta}_{\text{off}}), \tag{4.7}$$

so that for any fixed $\boldsymbol{\theta}_{\text{off}}$ we obtain a parametrization $\hat{q}(\cdot, \cdot; \boldsymbol{\theta}_{\text{off}})\colon \mathbb{R}^n \times \Omega \to \mathbb{R}$ in agreement with the definition in (4.1). When the dependence on the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ is clear from the context, we suppress it in the notation and simply write $\hat{q}(\boldsymbol{\theta}, \boldsymbol{x})$ as in (4.1).

In the context of model reduction, the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ is fixed in the offline phase and thus independent of time t and parameter $\boldsymbol{\mu}$. In contrast, the weight vector $\boldsymbol{\theta}$ plays the role of a reduced (latent) state, analogous to the coefficient or state vector (2.10) in linear model reduction, and thus can change in the online phase and depend on time t and parameter $\boldsymbol{\mu}$. We sometimes refer to $\boldsymbol{\theta}$ as the online weight vector to clearly distinguish it from the offline weight vector $\boldsymbol{\theta}_{\text{off}}$.

We broadly distinguish between *a priori* versus trained choices of the offline weights. In the trained case, the vector $\boldsymbol{\theta}_{\text{off}}$ (or parts of it) is fitted to snapshot data in the offline phase, analogous to how a reduced basis spanning the reduced space $\mathcal{V}$ is constructed from snapshots in linear model reduction. In contrast, in the *a priori* case, the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ is either empty or prescribed without training, for instance by choosing an analytic transformation with a given shift. The online weight vector $\boldsymbol{\theta}$ still has to be determined in the online phase.

In model reduction, we are often particularly interested in trained nonlinear parametrizations, where the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ is learned offline to provide an expressive parametrization so that only a low-dimensional online weight vector $\boldsymbol{\theta}$ needs to be computed online.

4.3.2. *Examples of nonlinear parametrizations with offline–online weights*

A common class of nonlinear parametrizations consists of parametrizations that are polynomial in the online weights. A quadratic example is

$$\hat{q}(\boldsymbol{\theta}, \boldsymbol{x}; \boldsymbol{\theta}_{\text{off}}) = \sum_{i=1}^{n} \theta_i \varphi_i(\boldsymbol{x}; \boldsymbol{\theta}_{\text{off}}) + \sum_{1 \le i \le j \le n} \theta_i \theta_j \psi_{ij}(\boldsymbol{x}; \boldsymbol{\theta}_{\text{off}}) \tag{4.8}$$

with functions $\varphi_i(\cdot;\boldsymbol{\theta}_{\text{off}})\colon \Omega \to \mathbb{R}$ and $\psi_{ij}(\cdot;\boldsymbol{\theta}_{\text{off}})\colon \Omega \to \mathbb{R}$ for $i, j = 1, \ldots, n$. These functions (or their coefficients in a discretization) are fitted to snapshot data by fitting the offline weight vector $\boldsymbol{\theta}_{\text{off}}$; we will revisit such polynomial approximations in Section 7.

Composing a linear expansion with a nonlinear transformation yields another important class. Let $\mathrm{T}(\cdot;\boldsymbol{\theta}_{\text{off}})\colon \mathbb{R}^r \times \Omega \to \Omega$ be a transformation that depends on the offline weight vector. Additionally, let $\varphi_1(\cdot;\boldsymbol{\theta}_{\text{off}}), \ldots, \varphi_r(\cdot;\boldsymbol{\theta}_{\text{off}})$ be a set of basis functions. With a decomposition of the online weight vector as $\boldsymbol{\theta} = [\boldsymbol{\alpha};\boldsymbol{\beta}]$, with $\boldsymbol{\alpha} \in \mathbb{R}^r$ and $\boldsymbol{\beta} \in \mathbb{R}^r$, one obtains

$$\hat{q}(\boldsymbol{\theta}, \boldsymbol{x};\boldsymbol{\theta}_{\text{off}}) = \sum_{i=1}^{r} \beta_i \varphi_i(\mathrm{T}(\boldsymbol{\alpha}, \boldsymbol{x};\boldsymbol{\theta}_{\text{off}});\boldsymbol{\theta}_{\text{off}}), \tag{4.9}$$

where we assume $n = 2r$ for ease of exposition. For example, $\mathrm{T}(\alpha, x) = x - \alpha$ could model translation given by a scalar shift α as in Section 4.1.4. Here the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ can be trained to find the basis $\varphi_1, \ldots, \varphi_r$ and fix parameters of the transformation T, while the online weight vector $\boldsymbol{\theta}$ selects the shift and coefficients. We will revisit such nonlinear parametrizations in Section 5.

A class of nonlinear parametrizations that we have briefly touched on in (4.4) are parametrizations that adapt the representation. Let the online weight vector $\boldsymbol{\theta} = [\boldsymbol{\alpha};\boldsymbol{\beta}] \in \mathbb{R}^n$ be decomposed into the feature vector $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{n_1}]^\top \in \mathbb{R}^{n_1}$ and the coefficient vector $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_{n_2}]^\top \in \mathbb{R}^{n_2}$ with $n = n_1 + n_2$. Furthermore, let $\psi_1(\cdot,\cdot;\boldsymbol{\theta}_{\text{off}}), \ldots, \psi_{n_2}(\cdot,\cdot;\boldsymbol{\theta}_{\text{off}})\colon \mathbb{R}^{n_1} \times \Omega \to \mathbb{R}$ be representation functions that depend on the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ and are trained on snapshot data offline. Then the nonlinear parametrization

$$\hat{q}(\boldsymbol{\theta}, \boldsymbol{x};\boldsymbol{\theta}_{\text{off}}) = \sum_{i=1}^{n_2} \beta_i \psi_i(\boldsymbol{\alpha}, \boldsymbol{x};\boldsymbol{\theta}_{\text{off}}) \tag{4.10}$$

is linear in the coefficients $\boldsymbol{\beta}$ for a fixed $\boldsymbol{\alpha}$ but nonlinear in the online weight vector $\boldsymbol{\theta}$ through the feature vector $\boldsymbol{\alpha}$. In particular, if $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ varies with time (and with parameters), then the representation

$$\psi_1(\boldsymbol{\alpha}(t, \boldsymbol{\mu}), \cdot;\boldsymbol{\theta}_{\text{off}}), \ldots, \psi_{n_2}(\boldsymbol{\alpha}(t, \boldsymbol{\mu}), \cdot;\boldsymbol{\theta}_{\text{off}})\colon \Omega \to \mathbb{R}$$

can vary with time as well. We will discuss such nonlinear parametrizations in Section 6.

Another common class is given by neural networks. As an example, let us consider a fully connected feedforward neural network with a linear output layer,

$$\hat{q}(\boldsymbol{\theta}, \boldsymbol{x};\boldsymbol{\theta}_{\text{off}}) = \mathcal{C}_L(\sigma(\mathcal{C}_{L-1}(\ldots \sigma(\mathcal{C}_1(\boldsymbol{x}))\ldots))), \tag{4.11}$$

where σ denotes an activation function that is applied component-wise and $\mathcal{C}_\ell(\boldsymbol{y}) = \boldsymbol{W}_\ell \boldsymbol{y} + \boldsymbol{b}_\ell$ denotes the ℓth layer with matrix $\boldsymbol{W}_\ell$ and bias $\boldsymbol{b}_\ell$ of suitable dimensions. One can then collect some of the weight matrices $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$ and biases $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_L$ into the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ and others into the online weight vector $\boldsymbol{\theta}$. One

approach is to include all weight matrices and biases of the first ℓ layers into the offline weight vector, and those of the last $L-\ell$ layers into the online weight vector $\boldsymbol{\theta}$. But there are various other variants that we will discuss in more detail in Section 7.

4.4. The elements of nonlinear model reduction

We identify three elements that are essential for nonlinear model reduction – the choice of the parametrization, the construction of reduced dynamics for the online weights, and the numerical solvers for efficient online evaluations – which we will revisit throughout the remainder of the survey.

4.4.1. Overview of elements of nonlinear model reduction

Up to this point, our discussion has focused primarily on approximation-theoretic aspects of nonlinear model reduction, in particular on how nonlinear parametrizations can overcome the limitations inherent to linear model reduction and the associated Kolmogorov barrier. While expressive nonlinear parametrizations are a crucial ingredient, they represent only one component of a successful nonlinear model reduction approach.

Beyond the choice of a nonlinear parametrization, one must construct reduced dynamics that are compatible with this parametrization, that is, evolution equations for the online weights whose solutions yield accurate reduced solutions over time and across parameters and can be solved robustly, analogous to suitable formulations and discretizations for full models. Moreover, the ultimate purpose of model reduction is to achieve lower computational costs compared to full-model simulations at a prescribed accuracy, and not only to reduce the number of degrees of freedom from N to n. Thus, achieving $n \ll N$ alone is insufficient unless the reduced model can be evaluated efficiently in the online phase. Efficient online computations typically require both suitable reduced dynamics and efficient numerical methods for time integration and nonlinear solvers.

Guided by these considerations, we identify and distinguish three elements of nonlinear model reduction:

(i) nonlinear parametrizations that provide expressive approximations and circumvent the Kolmogorov barrier,

(ii) reduced dynamics that are compatible with the chosen parametrization and govern the evolution of the online weights in an accurate and robust manner, and

(iii) efficient online evaluation strategies and numerical solvers that render reduced-model simulations computationally advantageous compared to full-model simulations.

These elements mirror standard concepts of numerically solving PDEs: one chooses an approximation class (e.g. a finite element space), a formulation that

defines a problem (e.g. Galerkin projection, least-squares, mixed methods) and then a numerical method for the efficient solution.

Remark 4.4. Error estimation and certification often form an additional, distinct element in model reduction. While rigorous and efficient *a posteriori* error estimation is mature for linear model reduction (Grepl and Patera 2005, Veroy and Patera 2005, Haasdonk and Ohlberger 2008, 2011, Rozza *et al.* 2008), it is in its early stages for nonlinear model reduction. For example, we refer to Klein and Ohlberger (2026) and Black, Schulze and Unger (2020). However, we note that nonlinear reduced models can be combined with the full models in multi-fidelity methods to obtain accuracy guarantees for approximations of outer-loop results; see Peherstorfer *et al.* (2018) for a survey of multi-fidelity methods.

4.4.2. Element 1: Nonlinear parametrizations

The first element is the nonlinear parametrization that defines the ansatz of the reduced solutions used to approximate the full-model solutions. Typically, one has to specify a suitable parametrization architecture $\hat{q}$ such as a polynomial ansatz (4.8) or a neural network (4.11) and then one determines the offline weights $\boldsymbol{\theta}_{\text{off}}$ in the offline phase by training on snapshot data.

Training on snapshot data to fit the offline weights $\boldsymbol{\theta}_{\text{off}}$ aims to make the parametrization capture the dominant features of the full-model solutions so that the dimension n of the online weight vector $\boldsymbol{\theta} \in \mathbb{R}^n$ can be kept low. In addition, the parametrization should satisfy suitable regularity properties, for example, small changes in the online weight vector $\boldsymbol{\theta}$ lead to controlled changes in $\hat{q}(\boldsymbol{\theta}, \cdot; \boldsymbol{\theta}_{\text{off}})$, which is important for robust online computations; see also Section 4.2.

More generally, the parametrization has to be chosen with the intended reduced dynamics and online evaluations in mind, because its structure can strongly affect the costs of evaluating $\hat{q}$ and of assembling quantities needed for evolving the reduced dynamics online.

4.4.3. Element 2: Reduced dynamics

The second element of nonlinear model reduction is to impose reduced dynamics on the weights of the nonlinear parametrization, that is, to derive equations that determine how the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ evolves in time and depends on the parameters so that $\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)$ approximates the full-model solution in some suitable sense.

In the setting considered throughout this survey, $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ is propagated over a time interval $t \in [0, T]$ for parameters $\boldsymbol{\mu} \in \mathcal{D}$ by enforcing the governing equations on the nonlinear parametrization in a reduced sense. The analogue in linear model reduction is the derivation of the reduced model for the reduced coordinates; however, for nonlinear parametrizations, standard techniques such as Galerkin projection cannot be applied verbatim because the classical projection framework critically relies on the vector space structure underlying linear parametrizations.

A key aspect of nonlinear model reduction is therefore to develop suitable reduced-dynamics formulations that are compatible with nonlinear parametrizations and support reliable online computations. In particular, the reduced dynamics critically rely on the map $\boldsymbol{\theta} \mapsto \hat{q}(\boldsymbol{\theta}, \cdot; \boldsymbol{\theta}_{\text{off}})$ and so their robustness depends on the regularity and conditioning of this map. In practice, it is desirable that the resulting methods can handle regions where the map becomes poorly conditioned.

4.4.4. Element 3: Efficient online evaluation strategies and solvers

The third element concerns the efficient online solution of the reduced dynamics, i.e. the numerical methods to advance the online weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ in time (or to solve the corresponding reduced algebraic problem in the steady case).

Depending on the time discretization, this involves time integrators and, for implicit schemes, solving nonlinear systems at each time step, typically via iterative methods that can require repeated solves with Jacobian (or Jacobian-like) systems induced by the parametrization $\hat{q}$. For many reduced dynamics that are compatible with nonlinear parametrizations (e.g. instantaneous residual minimization), each time step can be interpreted as an optimization problem to find an update to the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$. Consequently, online computations in nonlinear model reduction are often closer in spirit to optimization than to classical reduced linear system solves. Moreover, in many variational formulations, the reduced dynamics are obtained by requiring the residual to be orthogonal to test directions, which leads to inner products and norms involving the nonlinear parametrization $\hat{q}$ and often also its derivatives with respect to $\boldsymbol{\theta}$. In the online phase, these quantities must be computed by numerical quadrature or some other numerical means. Standard techniques can be applied in linear (and affine) settings to pre-compute local matrices in the offline phase, and then rapidly assemble a matrix online by superposition. However, typically these are no longer directly applicable with nonlinear parametrizations, because the quantities depend nonlinearly on $\boldsymbol{\theta}$ through $\hat{q}$ and therefore cannot generally be assembled from pre-computed terms alone. This is closely related in spirit to the lifting bottleneck in linear model reduction. Thus, even more so than in linear model reduction, a reduction in the number of degrees of freedom from N to n does not by itself imply a reduction in online costs.

5. Transformation-based methods

The class of nonlinear model reduction methods surveyed in this section is based on transformations. The central idea is to identify a suitable coordinate system in which the solution manifold becomes amenable to linear reduction. Rather than directly approximating the solution in the original variables, one introduces one or more transformations that capture dominant nonlinear features such as transport, rotation or deformation. A common feature of transformation-based methods is that the transformation is either motivated by physical insight about the underlying

problem or, when learned from data, admits a meaningful physical interpretation *a posteriori*.

5.1. Model reduction with transformation-based methods

The motivational example of the one-dimensional advection equation discussed in Section 3.3, and the representation of the solution as a translation of a template in Section 4.1.4, suggests the introduction of one or more explicit transformation maps T, typically taking the form of coordinate or transport maps. Such transformations may be prescribed *a priori*, as in the case of a known spatial shift, or inferred from data. Once suitable transformations are identified, there are two principal ways in which they can be incorporated into a model reduction approach.

(i) The transformation may be applied directly to the solution, so that linear model reduction techniques are applied to the transformed full-model solutions $q \circ \mathrm{T}$ rather than to q itself.

(ii) Alternatively, the transformation may act on the reduced basis functions φ_i, yielding an approximation based on transformed modes, as outlined in (4.9).

Depending on the properties of the transformation, the two approaches may be equivalent in some cases. In general, however, they lead to distinct reduced models. This distinction becomes particularly relevant when no diffeomorphism exists that transforms the governing equations into a form that admits an efficient linear reduced representation. An example of such a situation arises in the modelling of pulsed detonation combustors (Schulze *et al.* 2019).

Moreover, transformations need not be restricted to mappings of the spatial or temporal domain. More general transformation operators $\mathscr{T}$ acting directly on the reduced basis functions can be considered, leading to transformed modes of the form $\mathscr{T}_i\varphi_i$. In all these variants, the combination of linear model reduction with transformations yields reduced models with a clear physical interpretation, as the solution is decomposed into distinct features whose evolution is described through explicit transformations tailored to the application at hand.

5.2. Literature review of transformation-based methods

We survey the literature on nonlinear model reduction based on transformations.

5.2.1. Transformations via group actions and shifts

A classical and widely explored idea is the exploitation of known symmetries of the underlying system, such as translational or rotational invariance. In this setting, the dynamics associated with a group action are separated from the remaining evolution of the solution field, which is then approximated using linear model reduction techniques (Kirby and Armbruster 1992, Rowley and Marsden 2000, Rowley, Kevrekidis, Marsden and Lust 2003, Mowlavi and Sapsis 2018). Closely related is the method of freezing introduced by Beyn and Thümmler (2004), which

forms the conceptual basis of several reduction techniques. For instance, Ohlberger and Rave (2013) apply this idea in the context of model reduction to the Burgers' equation and show that once the dominant transport is factored out, the remaining solution evolution exhibits low-dimensional structure and can be efficiently reduced using linear techniques.

A limitation of many symmetry-based approaches is that they typically account for a single global group action, such as a uniform translation. As a consequence, capturing multiple simultaneously transported structures, e.g. several waves travelling at different speeds, poses significant challenges without further extensions. One such extension is the shifted POD introduced in Reiss, Schulze, Sesterhenn and Mehrmann (2018), which represents the solution as a superposition of low-rank components, each associated with its own shift. This allows features moving at different velocities to be approximated using a small number of modes, while the corresponding shifts are inferred directly from snapshot data rather than prescribed *a priori*. Extensions, variants and generalizations of the shifted POD are presented for instance in Black *et al.* (2020), Reiss (2021), Papapicco *et al.* (2022), Burela, Krah and Reiss (2025) and Krah *et al.* (2025). In a similar vein, Rim, Moe and LeVeque (2018) aim to reverse transport effects by shifting solution snapshots back to a reference template. An iterative procedure is used to extract multiple transported structures sequentially, and linear model reduction is then applied to this aligned data. Cagniart, Maday and Stamm (2019) introduce a data-driven calibration approach that trains a transformation to make the transformed solution snapshots linearly reducible. In the online phase, linear model reduction is applied to the transformed solution, and both the reduced state and the transformation are advanced, driven by the residual. Greif, Junk and Urban (2022) start with linear reduced approximations and then add ridge terms that are learned from data and that can account for transported features. A non-intrusive approach that also uses shifts is introduced in Issan and Kramer (2023). Composing linear approximations with global transport dynamics that evolve along characteristic curves of conservation laws is proposed in Rim, Peherstorfer and Mandli (2023). In particular, the transport dynamics based on the characteristics can be computed efficiently in the online phase. A recent generalization of this work is presented by Rim and Welper (2026).

5.2.2. Optimal transport and formulations in metric spaces

Another line of research on transformation-based model reduction is inspired by optimal transport theory. Iollo and Lombardi (2014) compute transport maps between snapshots and use these maps to advect the reduced basis functions in time. A different perspective is pursued by Ehrlacher *et al.* (2020) and Battisti *et al.* (2023), who formulate model reduction in metric spaces, in particular in the Wasserstein space. This means that solutions are represented as measures and reduced in Wasserstein geometry via barycentres, for example, which can be loosely interpreted as a transformation-induced nonlinear representation that is well suited for solutions with moving features. Extensions of these ideas to higher-dimensional settings

via sparse Wasserstein barycentres have been proposed in Do, Feydy and Mula (2025). Other approaches introduce concepts of optimal transport into deep learning methods (Khamlich, Pichi and Rozza 2025) and build a bridge to registration-based methods (Blickhan 2024). Closely related is work relying on the Radon cumulative distribution transform (Kolouri, Park and Rohde 2016), studied in the context of model reduction by, for example, Ren, Wolf and Mao (2021) and Long *et al.* (2025); see Rim (2018) for computational aspects regarding the Radon transform.

5.2.3. Registration-based methods

Registration-based approaches compute transformations directly from data, often in an equation-agnostic and purely data-driven way inspired by image registration, with the goal of aligning dominant features across parameter values or time instances to make them linearly reducible. Such registration concepts have been proposed for model reduction in various forms. In parametric settings where the parameter $\boldsymbol{\mu}$ induces non-smooth behaviour in the solution manifold, Welper (2017, 2020) proposes first learning suitable transformations that align moving features, and then performing interpolation in the transformed space, where the dependence on $\boldsymbol{\mu}$ is smoother. We note that Amsallem and Farhat (2008) already use nonlinear mappings to perform Grassmann interpolation between local reduced bases in the context of model reduction.

Taddei (2020) introduces an equation-independent variational registration procedure, which has been combined with efficient online computations in Ferrero, Taddei and Zhang (2022). Sarna and Benner (2021) pursue a purely data-driven approach that learns transformations and combines them with a regression-based method to compute reduced states online. Several works refine how transformations are constructed and deployed online: Grundel and Sarna (2024) combine transformations with adaptive reduced meshes to efficiently compute reduced approximations, Sarna, Giesselmann and Benner (2024) compute transformations via monotonic feature matching of discontinuities, while Torlo (2025) uses moving meshes. Registration can also be carried out in a space–time setting by learning a mapping on the combined space–time domain as in Taddei and Zhang (2021) and Kleikamp, Ohlberger and Rave (2022). Other work on registration-based methods includes that of Nonino, Ballarin, Rozza and Maday (2023), who explicitly target fluid–structure interaction problems, as well as Iollo, Labatut, Mounoud and Taddei (2026) on the foundations of registration on bounded domains.

5.2.4. Lagrangian formulations and other transformations

Transformation-based model reduction can also be formulated in a Lagrangian framework, where the dynamics are described in coordinates that move with the flow. The Lagrangian basis method introduced by Mojgani and Balajewicz (2017) is an early example of this perspective. A data-driven approach that extends dynamic mode decomposition to Lagrangian settings is presented by Lu and Tartakovsky

(2020, 2021). Finally, more problem-specific transformations have been proposed that explicitly identify shock curves or moving interfaces: Taddei, Perotto and Quarteroni (2015) explicitly track the shock curve to exploit piecewise smooth components, while Razavi and Yano (2025) and Taddei, Tifouti and Barral (2025) compute registration maps that align shocks. Alireza Mirhoseini and Zahr (2023) also align dominant features but do so implicitly via an online optimization problem.

5.3. *Shifted proper orthogonal decomposition (shifted POD)*

We now discuss the transformation-based shifted POD method in more detail. This was introduced in Reiss *et al.* (2018) and further developed in, for example, Schulze *et al.* (2019), Black *et al.* (2020), Reiss (2021), Papapicco *et al.* (2022), Schulze (2023), Burela *et al.* (2025) and Krah *et al.* (2025). Despite its name, the method is not tied to a shift transformation and thus we present it with general transformations. We specifically discuss the elements of nonlinear model reduction for shifted POD and conclude with a short discussion of the design of the transformation.

5.3.1. *Shifted POD: Motivational example*

To motivate shifted POD, we again discuss our prototypical transport-dominated problem, namely the one-dimensional advection equation; see Section 3.3. In particular, we consider the following instance of the advection equation PDE problem: set the time domain to $\mathcal{T} = (0, 1)$ and the spatial domain to $\Omega = (0, 1)$, and consider

$$\begin{aligned} \partial_t q_{\mathcal{Q}}(t, \boldsymbol{x}; \boldsymbol{\mu}) &= -\boldsymbol{\mu} \partial_{\boldsymbol{x}} q_{\mathcal{Q}}(t, \boldsymbol{x}; \boldsymbol{\mu}), \quad && (t, \boldsymbol{x}) \in \mathcal{T} \times \Omega, \\ q_{\mathcal{Q}}(t, 0; \boldsymbol{\mu}) &= q_{\mathcal{Q}}(t, 1; \boldsymbol{\mu}), && t \in \mathcal{T}, \\ q_{\mathcal{Q}}(0, \boldsymbol{x}; \boldsymbol{\mu}) &= q_{\mathcal{Q},0}(\boldsymbol{x}), && \boldsymbol{x} \in \Omega, \end{aligned} \tag{5.1}$$

with periodic boundary conditions. Although we have already seen that the Kolmogorov n-widths decay at best only polynomially for this problem class, we explicitly compute the best linear approximation of the solution. Assuming that the solution of (5.1) is available for parameter samples $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_M\} \subseteq \mathcal{D}$, the best linear basis $\{\varphi_1, \ldots, \varphi_n\}$ can be computed by solving the optimization problem

$$\begin{aligned} &\min \frac{1}{2} \sum_{\kappa=1}^{M} \int_{\mathcal{T}} \left\| q_{\mathcal{Q}}(t, \cdot; \boldsymbol{\mu}_\kappa) - \sum_{j=1}^{n} \langle q_{\mathcal{Q}}(t, \cdot; \boldsymbol{\mu}_\kappa), \varphi_j \rangle_{\mathcal{P}} \, \varphi_j \right\|_{\mathcal{P}}^2 \mathrm{d}t, \\ &\text{such that } \varphi_i \in \mathcal{Q} \text{ and } \langle \varphi_i, \varphi_j \rangle_{\mathcal{P}} = \delta_{ij} \text{ for } i, j = 1, \ldots, n, \end{aligned} \tag{5.2}$$

where δ_{ij} denotes the Kronecker delta and $\mathcal{P}$ denotes a Hilbert space, e.g. the pivot space. For the specific example of the advection equation (5.1) we choose the Hilbert spaces $\mathcal{Q} = H^1_{\text{per}}(\Omega)$ and $\mathcal{P} = L^2(\Omega)$. The optimization problem (5.2) is known as the POD minimization problem; see Sirovich (1987) and Gubisch and

Volkwein (2017). The optimization problem (5.2) has a closed-form solution, as the basis functions are given as the n leading eigenfunctions of the non-negative, self-adjoint, compact operator

$$\mathcal{R}\colon \mathscr{P} \to \mathscr{P}, \quad \mathcal{R}\varphi = \sum_{\kappa=1}^{M} \int_{\mathcal{T}} \langle q_{\mathcal{Q}}(t,\cdot;\boldsymbol{\mu}_\kappa), \varphi\rangle_{\mathscr{P}}\, q_{\mathcal{Q}}(t,\cdot;\boldsymbol{\mu}_\kappa)\,\mathrm{d}t$$

(see e.g. Gubisch and Volkwein 2017). For the particular choice of the advection equation as in (5.1), one can show (see Holmes *et al.* 1996, Black *et al.* 2020) that the optimal basis functions are given by Fourier modes, and thus are not tailored to the concrete solution, which only depends on the initial function $q_{\mathcal{Q},0}$. In fact, this observation serves as an explanation for the oscillatory behaviour of POD-based linear reduced models for transport-dominated problems.

5.3.2. *Shifted POD: Transformed modes*

The POD optimization problem (5.2) leads to the best linear approximation of the given data. The idea from Reiss *et al.* (2018) is to introduce multiple shift operators $\mathcal{T}_i$ to account for different waves inherent to the solution, and so to mitigate the issue with transport-dominated problems as in the motivational example. The shift operators are parametrized by an additional parameter γ_i, called the path variable, which amounts to the shift amount and, in multiple dimensions, the shift direction for each mode. To simplify the presentation, we restrict ourselves to one-dimensional path variables here, i.e. $\gamma_i \in \mathbb{R}$ for $i = 1, \ldots, n$, and refer to Black *et al.* (2020) for the vector-valued setting. One then seeks a reduced approximation of the form

$$\hat{q}(t,\boldsymbol{x},\boldsymbol{\mu}) = \sum_{i=1}^{n} \hat{q}_i(t,\boldsymbol{\mu})[\mathcal{T}_i(\gamma_i(t,\boldsymbol{\mu}))\varphi_i](\boldsymbol{x}), \tag{5.3}$$

where $\mathcal{T}_i$ is a shift operator acting on the basis function φ_i and the resulting function $[\mathcal{T}_i(\gamma_i(t,\boldsymbol{\mu}))\varphi_i]$ is the transformed mode that is then evaluated at the spatial coordinate. If the path variables $\gamma_1(t,\boldsymbol{\mu}), \ldots, \gamma_n(t,\boldsymbol{\mu})$ are not fixed *a priori*, then (5.3) is a nonlinear parametrization, which constitutes the first element of nonlinear model reduction as outlined in Section 4.4. In the language of Section 4.3, the online weights are therefore given by the coefficients $\hat{q}_1(t,\boldsymbol{\mu}), \ldots, \hat{q}_n(t,\boldsymbol{\mu})$ as well as the path variables $\gamma_1(t,\boldsymbol{\mu}), \ldots, \gamma_n(t,\boldsymbol{\mu})$. We emphasize that the method is not limited to $\mathcal{T}_i$ being a shift operator. In fact, any parameter family of operators can be used, which is why we refer to $\mathcal{T}_i$ as a transformation operator. Typical requirements include that $\gamma_i \mapsto \mathcal{T}_i(\gamma_i)\varphi_i$ is continuous, the transformation operators are $\mathcal{Q}$-invariant and that $\mathcal{T}_i$ is uniformly bounded.

To generalize the POD optimization problem (5.2) to the decomposition (5.3) with transformed modes, it is important to observe that in general there is no guarantee that $\{\mathcal{T}_i(\gamma_i)\varphi_i\}_{i=1}^n$ forms a set of orthonormal basis functions. Consequently, inspired by Reiss *et al.* (2018), Schulze *et al.* (2019) and Black *et al.* (2020) pose the transformed

POD optimization problem as

$$\min \frac{1}{2} \sum_{\kappa=1}^{M} \int_{\mathcal{T}} \left\| q_{\mathcal{Q}}(t, \cdot; \boldsymbol{\mu}_\kappa) - \sum_{i=1}^{n} \hat{q}_i(t, \boldsymbol{\mu}_\kappa) \mathcal{T}_i(\gamma_i(t, \boldsymbol{\mu}_\kappa)) \varphi_i \right\|_{\mathscr{P}}^2 \mathrm{d}t, \quad \text{such that } \varphi_i \in \mathcal{Q} \text{ and } \|\varphi_i\|_{\mathscr{P}} = 1 \text{ for } i = 1, \ldots, n, \tag{5.4}$$

which has to be minimized over the coefficient functions $\hat{q}_i$, the path variables γ_i and the modes φ_i. Using a suitable functional analytic setting including assumptions on the transformation operators $\mathcal{T}_i$ and appropriate spaces and bounds for the coefficient functions $\hat{q}_i$ and the path variables γ_i, one can show that the shifted POD optimization problem (5.4) is solvable; see Black *et al.* (2020) and Black, Schulze and Unger (2022). In comparison with the POD optimization problem (5.2), which can be obtained from (5.4) by setting $\mathcal{T}_i$ to be the identity, the analysis of the properties of (5.4) is more delicate; see Black *et al.* (2022). A notable exception is given if all transformation operators and their path variables are forced to be identical, that is, the approximation (5.3) takes the form

$$\hat{q}(t, \boldsymbol{x}, \boldsymbol{\mu}) = \sum_{i=1}^{n} \hat{q}_i(t, \boldsymbol{\mu}) [\mathcal{T}(\gamma(t, \boldsymbol{\mu})) \varphi_i](\boldsymbol{x}). \tag{5.5}$$

If we additionally assume that $\mathcal{T}$ is an isometry, then the corresponding minimization problem can be solved by first transforming snapshot data accordingly and then applying standard POD to transformed snapshots. For further details we refer to Black *et al.* (2020); see also Reiss *et al.* (2018), Cagniart *et al.* (2019) and Schulze *et al.* (2019) for related transformations and their computations. The specific case where the transformation operators are given by ridge functions is discussed in Greif *et al.* (2022).

5.3.3. Shifted POD: Computational aspects for the optimization problem

In general no closed-form solution to (5.4) is available and an approximate minimizer has to be computed numerically. Besides a spatial and temporal discretization, this also requires numerically solving the optimization problem, for instance by employing a gradient-based method as in Black *et al.* (2022). As discussed in Schulze (2023), even for a moderate number n this becomes computationally expensive, as the degrees of freedom scale with the number of temporal and spatial degrees of freedom. Instead of solving the full optimization problem, one can also use a variable projection approach, as discussed in detail in Schulze (2023). A good initialization is often important in practice. This can for instance be obtained by initializing the path variables so that they suitably align the snapshots; see also the techniques discussed in Cagniart *et al.* (2019), Mendible *et al.* (2020) and Reiss (2021). Given an initialization for the path variables, the modes can be initialized by applying the transformation operators with the path variables and the modes are then the leading singular vectors of the transformed snapshots, as was done in the case of a wildland fire simulation in Black *et al.* (2021*a*), for example.

5.3.4. Shifted POD: Construction of the reduced model

The second element of nonlinear model reduction constitutes the reduced dynamics. We start our discussion by assuming that the path variables γ_i are fixed, which, assuming a suitable parametrization of the path variables with weights independent of time and $\boldsymbol{\mu}$, means that we can consider them as offline weights; see Section 4.3. In this case, the transformed modes ansatz (5.3) can be used in a classical Galerkin projection framework similarly as in (2.12). In more detail, recall the PDE problem specified in (2.1) and the reduced state $\hat{\boldsymbol{q}}(t,\boldsymbol{\mu}) = [\hat{q}_1(t,\boldsymbol{\mu}),\dots,\hat{q}_n(t,\boldsymbol{\mu})]^\top \in \mathbb{R}^n$. We now introduce the reduced path vector as

$$\hat{\boldsymbol{\gamma}}(t,\boldsymbol{\mu}) = [\hat{\gamma}_1(t,\boldsymbol{\mu}),\dots,\hat{\gamma}_n(t,\boldsymbol{\mu})]^\top \in \mathbb{R}^n,$$

which, at the moment, we assume to be determined by the offline weights alone and thus can be considered as being given online. The associated Galerkin reduced model is

$$\boldsymbol{M}_1(\hat{\boldsymbol{\gamma}}(t,\boldsymbol{\mu}))\dot{\hat{\boldsymbol{q}}}(t,\boldsymbol{\mu}) + \boldsymbol{M}_s(\hat{\boldsymbol{\gamma}}(t,\boldsymbol{\mu}))\boldsymbol{D}(\hat{\boldsymbol{q}}(t,\boldsymbol{\mu}))\dot{\hat{\boldsymbol{\gamma}}}(t,\boldsymbol{\mu}) = \hat{\boldsymbol{f}}_1(\hat{\boldsymbol{q}}(t,\boldsymbol{\mu}),\hat{\boldsymbol{\gamma}}(t,\boldsymbol{\mu});\boldsymbol{\mu}) \quad (5.6)$$

with

$$\begin{aligned}
\boldsymbol{M}_1(\hat{\boldsymbol{\gamma}}) &= \left[\langle \mathscr{T}_i(\hat{\gamma}_i)\varphi_i, \mathscr{T}_j(\hat{\gamma}_j)\varphi_j\rangle_{\mathscr{P}}\right]_{i,j=1}^n \in \mathbb{R}^{n\times n},\\
\boldsymbol{M}_s(\hat{\boldsymbol{\gamma}}) &= \left[\langle \mathscr{T}_i(\hat{\gamma}_i)\varphi_i, \mathscr{T}_j'(\hat{\gamma}_j)\varphi_j\rangle_{\mathscr{P}}\right]_{i,j=1}^n \in \mathbb{R}^{n\times n},\\
\boldsymbol{D}(\hat{\boldsymbol{q}}) &= \operatorname{diag}(\hat{q}_1,\dots,\hat{q}_n) \in \mathbb{R}^{n\times n},\\
\hat{\boldsymbol{f}}_1(\hat{\boldsymbol{q}},\hat{\boldsymbol{\gamma}};\boldsymbol{\mu}) &= \left[\left\langle \mathscr{T}_i(\hat{\gamma}_i)\varphi_i, f\left(\cdot, \sum\nolimits_{k=1}^n \hat{q}_k \mathscr{T}_k(\hat{\gamma}_k)\varphi_k;\boldsymbol{\mu}\right)\right\rangle_{\mathscr{P}}\right]_{i=1}^n \in \mathbb{R}^n,
\end{aligned}$$

where $\mathscr{T}_i'$ denotes the derivative of $\mathscr{T}_i$.

If the reduced path variables are unknown online, i.e. if they constitute online weights, then (5.6) constitutes an underdetermined differential–algebraic equation; see Mehrmann and Unger (2023) for a recent survey. To render the underdetermined equation (5.6) well-posed, we have to add additional equations, e.g. of the form

$$\boldsymbol{\psi}(\hat{\boldsymbol{\gamma}},\dot{\hat{\boldsymbol{\gamma}}},\hat{\boldsymbol{q}},\dot{\hat{\boldsymbol{q}}};\boldsymbol{\mu}) = 0, \quad (5.7)$$

which are called *phase conditions* or *reconstruction equations* in the literature; see e.g. Rowley and Marsden (2000), Rowley *et al.* (2003), Beyn and Thümmler (2004) and Ohlberger and Rave (2013). *A priori*, it is unclear what the optimal choice for the phase condition (5.7) is, and several instances are reported in the literature, for instance by minimizing the temporal change of the reduced coefficients. We refer to Schulze (2023) for an overview. Black *et al.* (2020) propose obtaining the phase condition via a Petrov–Galerkin projection, where the test space is chosen as

$$\operatorname{span}\{\hat{q}_1\mathscr{T}_1'(\hat{\gamma}_1)\varphi_1,\dots,\hat{q}_n\mathscr{T}_n'(\hat{\gamma}_n)\varphi_n\}.$$

With this particular choice, the phase condition reads

$$\boldsymbol{\psi}(\hat{\boldsymbol{\gamma}},\dot{\hat{\boldsymbol{\gamma}}},\hat{\boldsymbol{q}},\dot{\hat{\boldsymbol{q}}};\boldsymbol{\mu}) = \boldsymbol{D}(\hat{\boldsymbol{q}})^\top\left(\boldsymbol{M}_s(\hat{\boldsymbol{\gamma}})^\top\dot{\hat{\boldsymbol{q}}} + \boldsymbol{M}_2(\hat{\boldsymbol{\gamma}})\boldsymbol{D}(\hat{\boldsymbol{q}})\dot{\hat{\boldsymbol{\gamma}}} - \hat{\boldsymbol{f}}_2(\hat{\boldsymbol{q}},\hat{\boldsymbol{\gamma}};\boldsymbol{\mu})\right) \quad (5.8)$$

with

$$\boldsymbol{M}_2(\hat{\boldsymbol{\gamma}}) = \left[\langle \mathcal{T}_i'(\hat{\gamma}_i)\varphi_i, \mathcal{T}_j'(\hat{\gamma}_j)\varphi_j\rangle_{\mathscr{P}}\right]_{i,j=1}^{n} \in \mathbb{R}^{n\times n},$$
$$\hat{\boldsymbol{f}}_2(\hat{\boldsymbol{q}}, \hat{\boldsymbol{\gamma}}; \boldsymbol{\mu}) = \left[\left\langle \mathcal{T}_i'(\hat{\gamma}_i)\varphi_i, f\left(\cdot, \sum_{k=1}^{n} \hat{q}_k \mathcal{T}_k(\hat{\gamma}_k)\varphi_k; \boldsymbol{\mu}\right)\right\rangle_{\mathscr{P}}\right]_{i=1}^{n} \in \mathbb{R}^n.$$

The coupled reduced model for the reduced state $\hat{\boldsymbol{q}}$ and the reduced path variable $\hat{\boldsymbol{\gamma}}$ is thus given by (ignoring the arguments)

$$\begin{bmatrix} \boldsymbol{I}_n & 0 \\ 0 & \boldsymbol{D}(\hat{\boldsymbol{q}})^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{M}_1(\hat{\boldsymbol{\gamma}}) & \boldsymbol{M}_s(\hat{\boldsymbol{\gamma}}) \\ \boldsymbol{M}_s(\hat{\boldsymbol{\gamma}})^\top & \boldsymbol{M}_2(\hat{\boldsymbol{\gamma}}) \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_n & 0 \\ 0 & \boldsymbol{D}(\hat{\boldsymbol{q}}) \end{bmatrix} \begin{bmatrix} \dot{\hat{\boldsymbol{q}}} \\ \dot{\hat{\boldsymbol{\gamma}}} \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{f}}_1(\hat{\boldsymbol{q}}, \hat{\boldsymbol{\gamma}}; \boldsymbol{\mu}) \\ \boldsymbol{D}(\hat{\boldsymbol{q}})^\top \hat{\boldsymbol{f}}_2(\hat{\boldsymbol{q}}, \hat{\boldsymbol{\gamma}}; \boldsymbol{\mu}) \end{bmatrix}. \quad (5.9)$$

It can be shown that this reduced model minimizes the residual in a certain sense (see Black *et al.* 2020), which was termed *continuous optimality* in Carlberg *et al.* (2017) and Lee and Carlberg (2020), and already used by Miller and Miller (1981) in the context of moving finite element methods. We emphasize that this is a particular instance of the Dirac–Frenkel variational principle, which will be discussed in more general settings in Section 7.

In general, there is no guarantee that the matrix on the left-hand side of (5.9) is non-singular, in which case a regularization might be required; see also the forthcoming Section 7.5. Indeed, whenever a coefficient $\hat{q}_i$ of the reduced state attains the value zero, then one cannot solve for the corresponding path variable $\hat{\gamma}_i$. To mitigate this issue it is favourable to enforce the same path variable and same transformation operator for multiple reduced coefficients. In the case of a single transformation operator and a single path variable, this amounts to the representation (5.5). The general case is presented in detail in Schulze (2023). In the case that the right-hand side of the PDE is linear, one can use semigroup theory (e.g. Pazy 1983) to construct an *a posteriori* error estimator (Black *et al.* 2020). We emphasize that the resulting error estimator is quite generic and can also be applied to certify the prediction error of a physics-informed neural network (Raissi, Perdikaris and Karniadakis 2019); see Hillebrecht and Unger (2025*a*,*b*).

5.3.5. *Shifted POD: Offline–online decomposition*

We now turn to the third element of nonlinear model reduction (see Section 4.4), which in this case is the efficient evaluation of the reduced model (5.9). Due to the nonlinear ansatz, all matrices and the right-hand sides formally depend on the full-model dimension, and might not be easily pre-computable. A notable exception is given if a single transformation operator with a single path variable is used as in (5.5), and the transformation operator is unitary, which is the case for a one-dimensional shift operator with periodic boundary conditions. In this case most of the inner products are independent of the full-model dimension and can be pre-computed and efficiently evaluated; see Schulze (2023, Theorem 4.3.1). In very specific cases, this is even true for the general ansatz (5.3), as demonstrated by Black, Schulze and Unger (2021*b*) for the one-dimensional wave equation. In

general, however, this is not the case, and the costs of the numerical evaluation of (5.9) scale with the dimension of the full model.

To mitigate this issue, different strategies are proposed in Black *et al.* (2021*a*), differentiating between the inner products for the matrices on the left-hand side of (5.9) and the evaluation of the inner products corresponding to the potentially nonlinear full-model right-hand side. As the inner-product matrices on the left-hand side of (5.9) only depend on $\hat{\boldsymbol{\gamma}}$, Black *et al.* (2021*a*) propose sampling these matrices for typical values of $\hat{\boldsymbol{\gamma}}$ and then assembling the matrices via interpolation of the sampled matrices. A similar strategy can also be employed if the full-model right-hand side f is linear, and we refer to Cagniart *et al.* (2019) and Schulze (2023) for details. Exploiting properties of the transformation operator and using the method of active subspaces presented in Constantine (2015) might be used to lower the dimension of the space to be sampled. If f is nonlinear, then we can apply an extension of empirical interpolation as discussed in Section 2.4.3 to account for the transformation operators. Corresponding ideas are presented in, for instance, Black *et al.* (2021*a*), Rim *et al.* (2023) and Grundel and Sarna (2024). An additional opportunity for more speedup is obtained by applying time-integrators with adaptive step-sizes, as the ansatz (5.3) often allows for large time steps compared to linear reduced models.

An alternative approach to the numerical integration of the reduced model (5.9) is given by learning the reduced coefficients and reduced paths via machine learning. In the case of transformed modes, this is pursued in Papapicco *et al.* (2022) and Burela *et al.* (2025).

5.3.6. Shifted POD: Transformation operators

The question that remains to be answered is the design of the transformation operators $\mathcal{T}_i$. In a one-dimensional setting with periodic boundary conditions, a simple shift operator might be sufficient. Nevertheless, the situation changes for non-periodic boundary conditions or more involved wave propagation in two or higher-dimensional computational domains. The case of non-periodic boundary conditions is discussed in Schulze (2023, Section 3.3), for instance by constant extrapolation, virtually extending the computational domain, or by zero padding. In general, it is desirable to construct the transformation operators based on expert knowledge of the application at hand, for instance by studying properties of the semigroup underlying the respective problem. In particular, the semigroup approach suggests the construction of transformation operators that are equivariant with respect to the full-model right-hand side f. We emphasize that incorporating such expert knowledge can be the key to having high-fidelity approximations even far outside the training data. If no such expert knowledge is available, then one can resort to learning the transformation operators, as for instance demonstrated in Krah, Büchholz, Häringer and Reiss (2023).

6. Online adaptive model reduction

This section surveys model reduction methods that adapt the reduced space over time during the online phase. The underlying motivation is that solutions to transport-dominated problems can often be approximated well in reduced spaces locally in time, even when approximations in a single, global reduced space are inefficient.

6.1. Model reduction with online basis updates

We consider online adaptive model reduction methods, which represent reduced solutions as linear combinations of basis functions that evolve with time t (and possibly the parameter $\boldsymbol{\mu}$). Correspondingly, the reduced space spanned by the basis functions evolves over time.

One common way to realize time-dependent basis functions is through a nonlinear parametrization in which the basis depends on time-varying features that are contained in the online weights $\boldsymbol{\theta}(t, \boldsymbol{\mu})$, as in (4.10). The online phase then includes an adaptation step for the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ that evolves these features (and thus the basis) over time. In the semi-discrete setting, the time-dependent reduced space $\mathcal{V}(t)$ is represented by the time-dependent basis matrix $\boldsymbol{V}(t) \in \mathbb{R}^{N \times n}$, possibly also depending on the parameter $\boldsymbol{\mu}$, which is then updated online. We remark that some of the transformation-based methods can also be interpreted as time-dependent basis methods where, for example, shifts evolve the basis over time.

The principle underlying time-dependent basis methods that can overcome the Kolmogorov barrier is local-in-time approximations: over short time windows, the full-model solutions can be well approximated in a low-dimensional vector space, but the space needs to be adapted over time as the solutions evolve. Unlike in linear model reduction with a single reduced space that is fixed over time, an online adaptive reduced space can approximate solutions with moving coherent structures and transport without requiring a high-dimensional online weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$.

A consequence of the online adaptation is that the offline–online decomposition of linear model reduction is abandoned or at least weakened. In particular, the basis matrix $\boldsymbol{V}(t)$ (or equivalently the corresponding basis functions) is adapted while the reduced model is simulated online; these basis updates can incur costs that scale with the dimension N of the full-model solution space, thereby limiting speedups. A central challenge is therefore to design basis-update mechanisms that limit the N-dependence, e.g. via sparse basis updates. At the same time, this weakening of the offline–online decomposition enables the reduced space to adapt online and capture dynamics and solution features that were not present in the training data used during the offline phase. This can be of particular importance when the parameter $\boldsymbol{\mu}$ is high-dimensional and the corresponding parameter domain $\mathcal{D}$ cannot be sampled exhaustively, or when the full model is so expensive to simulate that not many training data trajectories can be generated. We focus on online adaptive model reduction to overcome the Kolmogorov barrier rather than other advantages, such as

avoiding a comprehensive offline training phase, for which we refer to a discussion in Peherstorfer and Willcox (2015*b*) and Remark 6.1.

Remark 6.1. There is a line of work on adapting reduced spaces over outer-loop iterations, such as optimization, as in, for example, Arian, Fahl and Sachs (2000), Kunisch and Volkwein (2008), Amsallem, Zahr and Washabaugh (2015), Zahr and Farhat (2015) and Qian, Grepl, Veroy and Willcox (2017). The purpose of progressively building reduced spaces is often that the parameter domain cannot be exhaustively sampled, and so reduced models are built during outer-loop iterations. Instead, in this survey we focus on adaptation over time to mitigate the Kolmogorov barrier.

6.2. *Overview of online adaptive model reduction and time-dependent basis methods*

Let us set online adaptive model reduction in the broader context of time-dependent basis methods, i.e. methods where the approximation space evolves with time rather than staying fixed as in linear model reduction.

6.2.1. *Localized model reduction*

Localized model reduction methods rely on multiple low-dimensional spaces, with a selection or weighting mechanism to use them online. For example, a common approach is to pre-compute multiple reduced spaces offline and then select one online with an indicator function based on, for example, the current reduced solution, parameter or time. Selecting the reduced space based on online information results in a nonlinear parametrization. The key is that the decomposition into local reduced spaces is performed so that each local reduced space corresponds to a local regime that can be well approximated by a low-dimensional reduced space, even if a single global reduced space of the same dimension would lead to inaccurate approximations.

One way to find a decomposition into local regimes is by tailoring the local reduced spaces to regions in the parameter domain $\mathcal{D}$ (Eftang, Patera and Rønquist 2010, Eftang, Knezevic and Patera 2011, Haasdonk, Dihlmann and Ohlberger 2011, Eftang and Stamm 2012, Wieland 2015, Bonito *et al.* 2021). Another way is to decompose the time interval and use different local reduced spaces for each time window (Dihlmann, Drohmann and Haasdonk 2011, Drohmann, Haasdonk and Ohlberger 2011, Parish and Carlberg 2021, Shimizu and Parish 2021, Copeland, Cheung, Huynh and Choi 2022), which can be interpreted as an approximation with a switched or hybrid system. The reduced spaces can also be localized directly based on the state space (in the semi-discrete setting) (Amsallem, Zahr and Farhat 2012, Washabaugh, Amsallem, Zahr and Farhat 2012, Peherstorfer, Butnaru, Willcox and Bungartz 2014, Amsallem *et al.* 2015, Amsallem and Haasdonk 2016, Hess *et al.* 2019, Grimberg, Farhat, Tezaur and Bou-Mosleh 2021, Geelen and Willcox 2022). One approach is via clustering: a clustering method is applied to the snapshots in

the offline phase so that snapshots that are similar are grouped together into clusters and a local reduced space is constructed for each cluster of snapshots. Localized model reduction can also be viewed as yielding nonlinear parametrizations that are induced by a union of multiple reduced spaces, as in Cohen, Dahmen, Mula and Nichols (2022*a*). Motivated by domain decomposition and multiscale methods, Ohlberger and Schindler (2015, 2017), Buhr and Smetana (2018) and Smetana and Taddei (2023) localize over the spatial domain and compute local reduced spaces for each subdomain, which is specifically designed to target problems with a multiscale structure. A central development of these works is rigorous error control, which is then leveraged for adaptive online enrichment (Ohlberger and Schindler 2017). If the domain can be partitioned in such a way that almost the same localized features can be expected in each subdomain, then Van Gastelen, Edeling and Sanderse (2026) propose using a single local basis. Localization can also be used to support local modifications in the online phase, so that updates and recomputation remain local and lightweight (Phuong Huynh, Knezevic and Patera 2013, Smetana and Patera 2016, Buhr, Engwer, Ohlberger and Rave 2017).

Dictionary approaches push the idea of selecting a local reduced space further by pre-computing a large set of candidate basis functions offline and then selecting a few of the candidate basis functions online to span a reduced space (Kaulmann and Haasdonk 2013, Abgrall, Amsallem and Crisovan 2016, Daniel, Casenave, Akkari and Ryckelynck 2020, Balabanov and Nouy 2021, Herkert, Buchfink and Haasdonk 2024, Nouy and Pasco 2024).

6.2.2. *Dynamic low-rank approximations and related methods*

Koch and Lubich (2007) introduce dynamic low-rank approximations, which parametrize the solution via a low-rank matrix and derive reduced dynamics for how the factors of the low-rank matrix representation evolve over time. The key principle underlying the reduced dynamics of dynamic low-rank approximations is the Dirac–Frenkel variational principle (Dirac 1930, Frenkel 1934), which projects the full dynamics onto the tangent space of the manifold of matrices of a fixed rank; we will revisit the Dirac–Frenkel variational principle for general nonlinear parametrizations in Section 7.

A challenge of dynamic low-rank approximations is that the reduced dynamics of the low-rank factors can become poorly conditioned due to numerically small singular values of the low-rank matrix representations. Thus, applying standard integrators can suffer from strong step-size restrictions. Correspondingly, tailored time integration schemes for dynamic low-rank approximations have been developed. There are projector-splitting integrators introduced by Lubich and Oseledets (2014) and Kieri, Lubich and Walach (2016), which are robust to the small singular values in the low-rank matrix representations. An alternative is offered by integrators called basis-update and Galerkin (BUG), introduced by Ceruti and Lubich (2022) and Ceruti, Einkemmer, Kusch and Lubich (2024*a*), which can be computationally more efficient. In general, efficient, stable and robust time integration of the

reduced dynamics corresponding to dynamic low-rank approximations remains an active research direction; see e.g. Ceruti, Kusch and Lubich (2022), Hauck and Schnake (2023), Hochbruck, Neher and Schrammer (2023), Ceruti, Kusch and Lubich (2024*b*), Bachmayr, Eisenmann and Uschmajew (2025*b*), Lam, Ceruti and Kressner (2025), Nobile and Trindade (2025*b*) and Carrel (2026).

Another active research direction is structure preservation. For example, for Hamiltonian wave equations, a symplectic dynamic low-rank approximation approach was developed by Musharbash, Nobile and Vidličková (2020). For kinetic problems, in particular for Vlasov-type equations, there is a line of work on enforcing conservation laws in dynamic low-rank approximations (Einkemmer and Lubich 2019, Einkemmer and Joseph 2021, Einkemmer, Hu and Wang 2021, Einkemmer, Ostermann and Scalone 2023, Coughlin, Hu and Shumlak 2024, Hesthaven, Pagliantini and Ripamonti 2024).

Dynamic low-rank approximations have also been extended to tensor decompositions by Koch and Lubich (2010), Lubich, Rohwedder, Schneider and Vandereycken (2013), Arnold and Jahnke (2014) and Lubich, Oseledets and Vandereycken (2015). Additionally, there are methods to evolve bases that specifically represent only parts of the full-model dynamics such as instability directions (Babaee and Sapsis 2016).

Dynamic low-rank approximations are related to the dynamic orthogonal decomposition for stochastic (partial) differential equations, which was introduced by Sapsis and Lermusiaux (2009). There is a line of work on the error analysis for dynamic orthogonal approximations (Musharbash, Nobile and Zhou 2015, Musharbash and Nobile 2018, Kazashi, Nobile and Zoccolan 2025) and for time integration (Feppon and Lermusiaux 2018, Charous and Lermusiaux 2023, 2024). Furthermore, there are generalizations to Petrov–Galerkin formulations for the random PDE setting (Nobile and Trindade 2025*a*,*b*).

We can provide only a very brief overview of dynamic low-rank approximation methods and their related methods. We refer to the dedicated survey by Einkemmer *et al.* (2025) for more details.

6.2.3. Time-adaptive basis in model reduction

Let us now focus on time-adaptive basis methods developed explicitly for model reduction. For a survey dedicated to model reduction of time-dependent problems, see Hesthaven *et al.* (2022*b*).

One line of work on online adaptive reduced spaces derives the basis evolution by mimicking the transport dynamics. For example, Iollo and Lombardi (2014) build on ideas from optimal transport to evolve the basis functions spanning the reduced space. Analogously, Gerbeau and Lombardi (2014) evolve the reduced basis based on approximated Lax pairs. Related concepts are followed by Rim *et al.* (2023) for transporting a reduced space along characteristics. A different approach is presented in Black *et al.* (2020), which evolves the modes of the reduced basis via instantaneous residual minimization in time; see Section 5 for a detailed discussion.

Rapún and Vega (2010) update a reduced model by switching back to the full model occasionally to generate new snapshots. The method introduced in Carlberg (2015) and Etter and Carlberg (2020) aims to mimic adaptive mesh refinement in finite elements by splitting reduced-basis vectors to enrich the reduced space. The focus is on failsafe refinement in the sense that the full-model space is recovered in the limit.

Another line of work performs data-driven online basis updates by incorporating newly available online information, such as full-model residuals and incomplete state fields, to derive low-rank updates to the reduced spaces. For example, for linear problems, dynamic data-driven reduced models introduced by Peherstorfer and Willcox (2015*a*) adapt reduced spaces with an incremental SVD approach and the reduced-model operators with corresponding low-rank updates, without having to rebuild the reduced model from scratch.

For nonlinear problems, Peherstorfer and Willcox (2015*b*) introduce the online adaptive discrete empirical interpolation method (ADEIM), which builds on the empirical interpolation method (Barrault *et al.* 2004, Chaturantabut and Sorensen 2010) and computes low-rank updates to the reduced space from sparse residual samples. The basis update is formulated as a low-rank correction inferred from a small (possibly random) sketch via a sampling operator. Relying on sparse residual samples helps to reduce online costs. The sparsity of the updates also helps to mitigate the lifting bottleneck when reducing nonlinear systems (see Section 2.4.2). A central component of ADEIM is how to choose and adapt the sampling points over time. Peherstorfer (2020) couples online basis updates with adaptive sampling, which is shown to be critical for transport-dominated problems. Sparse sampling raises the question not only of where to sample but also when to sample: sampling the residual information corresponding to the current state can lag behind. Lookahead strategies use short-term forecasting rules to evaluate the residual and forecast states that anticipate upcoming dynamics, which improves robustness of the online updates and reduces instabilities; see Singh, Uy and Peherstorfer (2023) and Zucatti and Zahr (2024).

There are multiple ways to compute the low-rank updates from the sparse samples. The basis updates proposed in the original ADEIM work (Peherstorfer and Willcox 2015*b*) do not guarantee that the adapted basis is orthogonal. Zimmermann, Peherstorfer and Willcox (2018) build on subspace tracking methods from signal processing (Balzano, Nowak and Recht 2010) to derive an efficient rank-one update rule for ADEIM that preserves orthonormality of the adapted basis over time. Huang and Duraisamy (2023) and Mohaghegh and Huang (2026) introduce a related online adaptation strategy specifically for least-squares formulations, which are key for reducing chaotic and multiscale problems.

There are also time-dependent basis methods in model reduction that do not update the basis via online sparse residual information. For example, Ramezanian, Nouri and Babaee (2021), Naderi and Babaee (2023) and Aitzhan, Nouri, Givi and Babaee (2025) derive and integrate evolution equations for a low-rank representation

directly from the governing equations. In this sense, these methods share similarities with dynamic low-rank approximations.

There are also other online adaptive reduced basis methods inspired by dynamic low-rank approximations and that are structure-preserving; see e.g. Hesthaven and Pagliantini (2021), Pagliantini (2021) and Hesthaven, Pagliantini and Ripamonti (2022*a*). In particular, Pagliantini and Vismara (2025, 2026) show how to preserve structure even when the basis is adapted via sparse sampling-type reduction strategies such as empirical interpolation.

Another approach to online adaptive reduced-space methods is to pre-compute a time-dependent reduced space in the offline phase from training trajectories and then use it online without further adaptation or enrichment of the reduced space (Billaud-Friess and Nouy 2017).

6.3. Adaptive discrete empirical interpolation method (ADEIM)

We now discuss online adaptive model reduction using ADEIM as an example of how to realize online updates. Reduced models based on ADEIM adapt the reduced space online with low-rank updates from sparse residual information. The presentation loosely follows Peherstorfer and Willcox (2015*b*) and Peherstorfer (2020). We drop the dependence of the state on $\boldsymbol{\mu}$ for ease of exposition in this section.

6.3.1. Time-discrete full models

Let us consider a fully discrete full model of the form

$$\boldsymbol{q}_{k-1} = \boldsymbol{f}^{(\Delta)}(\boldsymbol{q}_k; \boldsymbol{\mu}), \quad k = 1, \ldots, K, \tag{6.1}$$

which can be obtained from the semi-discrete full model (2.4) with, for example, an implicit time-integration scheme. In particular, $\boldsymbol{q}_k$ is an approximation to $\boldsymbol{q}(t_k)$ for some $t_k \in [0, T]$. The form (6.1) generally requires a nonlinear solve to obtain $\boldsymbol{q}_k$ from $\boldsymbol{q}_{k-1}$. Note that the right-hand side function $\boldsymbol{f}$ of the semi-discrete full model (2.4) and the right-hand side function $\boldsymbol{f}^{(\Delta)}$ of the fully discrete full model (6.1) are different.

Following the steps in Section 2.3, a discrete reduced model can be obtained from the discrete full model as

$$\hat{\boldsymbol{q}}_{k-1} = \hat{\boldsymbol{f}}^{(\Delta)}(\hat{\boldsymbol{q}}_k; \boldsymbol{\mu}), \quad k = 1, \ldots, K, \tag{6.2}$$

where the right-hand side function $\hat{\boldsymbol{f}}^{(\Delta)}$ is obtained from the full-model right-hand side function $\boldsymbol{f}^{(\Delta)}$ via empirical interpolation; see Section 2.4. In particular, the map $\hat{\boldsymbol{f}}^{(\Delta)}$ depends on the basis matrix $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n] \in \mathbb{R}^{N \times n}$ and the interpolation points matrix $\boldsymbol{P}$ corresponding to the pairwise distinct interpolation points $p_1, \ldots, p_n \in \{1, \ldots, N\}$.

In this section on ADEIM and in the context of online adaptive reduced modelling, we refer to the reduced model (6.2) as a static reduced model because the basis matrix $\boldsymbol{V}$ and the interpolation points $\boldsymbol{P}$ are fixed over all time steps $k = 1, \ldots, K$.

6.3.2. ADEIM: Basis and points adaptation

We now discuss ADEIM's update mechanism for the basis and interpolation points.

Basis updates and nonlinear parametrizations. Now let the basis matrix $\boldsymbol{V}_k$ depend on the time step $k = 1, \ldots, K$. ADEIM adapts the basis matrix $\boldsymbol{V}_{k-1}$ to $\boldsymbol{V}_k$ as

$$\boldsymbol{V}_k = \boldsymbol{V}_{k-1} + \boldsymbol{\alpha}_{k-1}\boldsymbol{\beta}_{k-1}^\top, \tag{6.3}$$

where $\boldsymbol{\alpha}_{k-1} \in \mathbb{R}^{N\times r}$ and $\boldsymbol{\beta}_{k-1} \in \mathbb{R}^{n\times r}$ provide a rank $r \in \mathbb{N}$ update. Typically, the rank is small, $r \ll n$, or even just a rank-one update, $r = 1$. In addition to the basis matrix, ADEIM adapts the interpolation points so that the matrix $\boldsymbol{P}_{k-1}$ is adapted to $\boldsymbol{P}_k$.

Adapting the basis matrix $\boldsymbol{V}_k$ (and the interpolation points matrix $\boldsymbol{P}_k$) over time can be interpreted as evolving the representation of the reduced state. In the terminology of nonlinear parametrizations, this means that the online weights at time step k comprise the reduced state (e.g. $\hat{\boldsymbol{q}}_k$) as well as the low-rank update variables (e.g. $\boldsymbol{\alpha}_{k-1}$ and $\boldsymbol{\beta}_{k-1}$). More generally, the online weights may also include variables associated with the interpolation point update, as well as additional auxiliary variables required by the update, such as sampling points and window data. The offline weights consist of quantities fixed in the offline phase, such as the initial reduced basis matrix and initial interpolation points matrix. From this viewpoint, ADEIM can be interpreted as realizing a fully discrete instance of the time-adaptive nonlinear parametrization discussed earlier in (4.10): the reduced solution is still represented in a low-dimensional form, but the representation is adapted online through time-dependent basis updates.

Basis updates. Let us first consider the basis adaptation. For ease of exposition, we adapt the basis matrix at every time step $k = 1, \ldots, K$; however, in practice, the basis is often updated only at selected adaptation steps (e.g. every zth time step), to reduce online costs. At time step k, the basis matrix $\boldsymbol{V}_{k-1}$ from the previous time step $k-1$ is given, and ADEIM adapts it to the basis matrix $\boldsymbol{V}_k$.

The update is constructed such that the space given by the updated basis matrix $\boldsymbol{V}_k$ can approximate well the vectors in a sliding window

$$\boldsymbol{F}_{k-1} = [\mathfrak{f}_{k-w}, \ldots, \mathfrak{f}_{k-1}] \tag{6.4}$$

of length $w \in \mathbb{N}$. One might want to use the full-model solutions as the vectors of the sliding window $\boldsymbol{F}_{k-1}$, but they are unavailable. Instead, ADEIM uses surrogates $\mathfrak{f}_{k-w}, \ldots, \mathfrak{f}_{k-1}$ that capture some of the information from the full-model solutions but can be computed without the expensive cost of time-stepping the full model. We will discuss how to obtain the vectors of the sliding window in Section 6.3.4. For now, let us assume the sliding window $\boldsymbol{F}_{k-1}$ is given.

Let us formally define an objective J_{nl} for computing the basis update as the norm of the residual when approximating the columns of $\boldsymbol{F}_{k-1}$ in the updated space spanned by the adapted basis matrix $\boldsymbol{V}_{k-1} + \boldsymbol{\alpha}\boldsymbol{\beta}^\top$ and with the interpolation points

matrix $\boldsymbol{P}$, that is,

$$J_{\mathrm{nl}}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{P}) = \|(\boldsymbol{V}_{k-1} + \boldsymbol{\alpha}\boldsymbol{\beta}^\top)(\boldsymbol{P}^\top(\boldsymbol{V}_{k-1} + \boldsymbol{\alpha}\boldsymbol{\beta}^\top))^{-1}\boldsymbol{P}^\top\boldsymbol{F}_{k-1} - \boldsymbol{F}_{k-1}\|_F. \tag{6.5}$$

A low-rank update and interpolation points matrix that minimize J_{nl} minimize the error of approximating the vectors in the sliding window in the adapted space (6.3). Note that an admissible interpolation points matrix needs to satisfy additional conditions, which are not present in the objective (6.5). The objective J_{nl} is nonlinear in the update $\boldsymbol{\alpha}\boldsymbol{\beta}^\top$ and the interpolation points matrix $\boldsymbol{P}$. Solving a nonlinear optimization problem with objective J_{nl} can be computationally expensive in the online phase.

ADEIM simplifies the objective J_{nl} to the objective

$$J(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \|(\boldsymbol{V}_{k-1} + \boldsymbol{\alpha}\boldsymbol{\beta}^\top)\boldsymbol{C}_{k-1} - \boldsymbol{F}_{k-1}\|_F, \tag{6.6}$$

where the coefficient matrix from time step $k-1$,

$$\boldsymbol{C}_{k-1} = \left(\boldsymbol{P}_{k-1}^\top\boldsymbol{V}_{k-1}\right)^{-1}\boldsymbol{P}_{k-1}^\top\boldsymbol{F}_{k-1}, \tag{6.7}$$

is used instead of the coefficients that depend on the adapted quantities as in (6.5). Correspondingly, the objective J is independent of the adapted interpolation points matrix. To reduce the costs of the online update further, the residual is only minimized at sparse sampling points,

$$\min_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^{N\times r}\\ \boldsymbol{\beta}\in\mathbb{R}^{n\times r}}} \left\|\boldsymbol{S}_{k-1}^\top\left((\boldsymbol{V}_{k-1} + \boldsymbol{\alpha}\boldsymbol{\beta}^\top)\boldsymbol{C}_{k-1} - \boldsymbol{F}_{k-1}\right)\right\|_F, \tag{6.8}$$

where

$$\boldsymbol{S}_{k-1} = \left[\boldsymbol{e}_{s_1^{(k-1)}}, \ldots, \boldsymbol{e}_{s_{n_s}^{(k-1)}}\right]$$

is the sampling matrix corresponding to the pairwise distinct sampling points $s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}$. Typically, $n \leq n_s \ll N$. Notice that the problem (6.8) specifies $\boldsymbol{\alpha}_{k-1}$ only at the n_s sampling points $s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}$.

Peherstorfer and Willcox (2015*b*) build on an eigenvalue decomposition to compute the update that solves (6.8); however, again with the aim of reducing the costs of the online update, Peherstorfer (2020, Algorithm 3) shows that an approximation can be efficiently computed from a thin SVD of the $n_s \times w$ matrix $\boldsymbol{S}_{k-1}^\top\boldsymbol{R}_{k-1}$, which is the residual matrix $\boldsymbol{R}_{k-1} = \boldsymbol{V}_{k-1}\boldsymbol{C}_{k-1} - \boldsymbol{F}_{k-1}$ restricted to the components corresponding to the sampling points $s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}$. Let $\boldsymbol{U}_{1:r} \in \mathbb{R}^{n_s\times r}$ and $\boldsymbol{W}_{1:r} \in \mathbb{R}^{w\times r}$ contain the first r left- and right-singular vectors of $\boldsymbol{S}_{k-1}^\top\boldsymbol{R}_{k-1}$ corresponding to singular values $\sigma_1, \ldots, \sigma_r$, respectively. Collect the first r singular values $\sigma_1, \ldots, \sigma_r$ on the diagonal of $\boldsymbol{\Sigma}_{1:r}$. Then the ADEIM update is

$$\boldsymbol{\alpha}_{k-1} = -\boldsymbol{S}_{k-1}\boldsymbol{U}_{1:r}\boldsymbol{\Sigma}_{1:r} \in \mathbb{R}^{N\times r}, \quad \boldsymbol{\beta}_{k-1} = (\boldsymbol{C}_{k-1}^+)^\top\boldsymbol{W}_{1:r} \in \mathbb{R}^{n\times r}, \tag{6.9}$$

which changes the basis matrix $\boldsymbol{V}_{k-1}$ only at the sampling points $s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}$

when applied as in (6.3). This is consistent with the objective (6.8) that defines the update $\boldsymbol{\alpha}_{k-1}$ only at the sampling points $s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}$ corresponding to $\boldsymbol{S}_{k-1}$. Notice, however, that subsequent orthonormalization of $\boldsymbol{V}_k$ generally alters all rows. The costs of computing the SVD for solving the problem scale as $O(n_s w^2)$, in the common case that $w \leq n_s$.

Updating the interpolation points. After the basis matrix has been updated from $\boldsymbol{V}_{k-1}$ to $\boldsymbol{V}_k$, the interpolation points matrix $\boldsymbol{P}_{k-1}$ is updated to $\boldsymbol{P}_k$. Peherstorfer and Willcox (2015*b*) propose an algorithm that computes a rank-one update to $\boldsymbol{P}_{k-1}$, which corresponds to an update of at most one point per basis update. In contrast, Peherstorfer (2020) re-computes the interpolation points using the QDEIM algorithm (Drmač and Gugercin 2016) from scratch. The costs of computing the points from scratch scale as $O(Nn^2)$ and thus linearly in the full-model dimension N.

6.3.3. ADEIM: Forward step of reduced model

At time step k, after the basis matrix has been updated to $\boldsymbol{V}_k$ and the interpolation points matrix to $\boldsymbol{P}_k$, the corresponding reduced model right-hand side function $\hat{\boldsymbol{f}}_k^{(\Delta)}$ is derived from the full-model right-hand side function $\boldsymbol{f}^{(\Delta)}$ via empirical interpolation. Notice that the reduced right-hand side function $\hat{\boldsymbol{f}}_k^{(\Delta)}$ depends on k because the basis matrix and the interpolation points depend on k. The reduced model is then integrated forward one step from $k-1$ to k to compute the reduced solution $\hat{\boldsymbol{q}}_k$ at time step k, which means solving the generally nonlinear system of equations $\hat{\hat{\boldsymbol{q}}}_{k-1} = \hat{\boldsymbol{f}}_k^{(\Delta)}(\hat{\boldsymbol{q}}_k; \boldsymbol{\mu})$, where $\hat{\hat{\boldsymbol{q}}}_{k-1}$ is $\boldsymbol{V}_{k-1}\hat{\boldsymbol{q}}_{k-1}$ projected onto the adapted basis $\boldsymbol{V}_k$.

6.3.4. ADEIM: Sliding window

Recall that the columns of the sliding window (see (6.4)) serve as surrogates for unavailable full-model solutions. ADEIM leverages that if the full-model solution $\boldsymbol{q}_k$ at time step k is plugged into the fully discrete full model (6.1), then $\boldsymbol{q}_{k-1}$ is obtained.

In ADEIM, at time step k, the sliding window is updated from $\boldsymbol{F}_{k-1}$ to $\boldsymbol{F}_k$. After computing the reduced solution $\hat{\boldsymbol{q}}_k$ with the adapted basis $\boldsymbol{V}_k$ (see Section 6.3.3), it is used to derive

$$\mathfrak{f}_k = \boldsymbol{f}^{(\Delta)}(\boldsymbol{V}_k\hat{\boldsymbol{q}}_k; \boldsymbol{\mu}), \tag{6.10}$$

which is an approximation of the full-model solution $\boldsymbol{q}_{k-1}$ at the previous time step $k-1$. The window is then updated from $\boldsymbol{F}_{k-1}$ to $\boldsymbol{F}_k$ by dropping the oldest column and adding $\mathfrak{f}_k$. The window $\boldsymbol{F}_k$ is then used to drive the basis update from $\boldsymbol{V}_k$ to $\boldsymbol{V}_{k+1}$ at the next time step $k+1$.

The surrogate $\mathfrak{f}_k$ generally has components outside the reduced space spanned by the columns of $\boldsymbol{V}_k$, whereas the lifted reduced solution $\boldsymbol{V}_k\hat{\boldsymbol{q}}_k$ lies in the range of $\boldsymbol{V}_k$ by construction. Therefore, using surrogates of the form (6.10) in the sliding window can reveal missing directions in the current basis and lead to meaningful

basis updates (6.9). In this sense, $\mathfrak{f}_k$ can be viewed as residual-type information: it probes the full-model right-hand side at the current reduced solution and captures components that the current basis cannot represent.

Now recall that ADEIM minimizes the residual (6.8) only at the $n_s < N$ sampling points corresponding to $\boldsymbol{S}_{k-1}$ and not at all N components. This means that the full-model right-hand side function needs to be evaluated only at the components corresponding to the sampling points, which we denote as

$$\boldsymbol{S}_{k-1}^\top \mathfrak{f}_k = \boldsymbol{S}_{k-1}^\top \boldsymbol{f}^{(\Delta)}(\boldsymbol{V}_k \hat{\boldsymbol{q}}_k; \boldsymbol{\mu}). \tag{6.11}$$

The components that are not indexed by $\boldsymbol{S}_{k-1}$ can be approximated with empirical regression as

$$\check{\boldsymbol{S}}_{k-1}^\top \mathfrak{f}_k = \check{\boldsymbol{S}}_{k-1}^\top \boldsymbol{V}_k \left(\boldsymbol{S}_{k-1}^\top \boldsymbol{V}_k\right)^+ \boldsymbol{S}_{k-1}^\top \boldsymbol{f}^{(\Delta)}(\boldsymbol{V}_k \hat{\boldsymbol{q}}_k; \boldsymbol{\mu}), \tag{6.12}$$

where $\check{\boldsymbol{S}}_{k-1}$ corresponds to the sampling points matrix associated with the complement

$$\{1, \ldots, N\} \setminus \left\{s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}\right\}.$$

Note that we approximate the unobserved components by a least-squares regression using the pseudo-inverse $\left(\boldsymbol{S}_{k-1}^\top \boldsymbol{V}_k\right)^+$ rather than via interpolation; see Section 2.4.3 for oversampling in empirical interpolation. With this procedure, ADEIM evaluates the right-hand side function $\boldsymbol{f}^{(\Delta)}$ of the full model at only the n_s sampling points.

Computing $\mathfrak{f}_k$ requires only evaluating the time-discrete full-model right-hand side function $\boldsymbol{f}^{(\Delta)}$ at the lifted reduced solution $\boldsymbol{V}_k \hat{\boldsymbol{q}}_k$, rather than solving the nonlinear systems corresponding to integrating the full model (6.1) for one time step. Consequently, the cost of obtaining a vector for the sliding window is comparable to a full-model residual evaluation. Note, however, that if the full model is discretized with an explicit scheme, then evaluating $\boldsymbol{f}^{(\Delta)}$ and advancing the full model by one step are typically of comparable cost.

The procedure described here for filling the sliding window is backward-looking: the reduced solution $\hat{\boldsymbol{q}}_k$ at time step k is used to obtain the surrogate approximating the full-model solution at time step $k-1$. Lookahead strategies have been developed that use inexpensive forecasts to generate forward-looking surrogates, which avoid the one-step lag and so can improve stability and robustness (Singh *et al.* 2023, Zucatti and Zahr 2024).

6.3.5. ADEIM: Sampling

The ADEIM basis updates are driven by the sliding window, which is obtained from sparse evaluations of the full-model right-hand side function as given in (6.11)–(6.12). The indices of the components at which the full-model right-hand side function is evaluated are given by the sampling points denoted as $s_1^{(k-1)}, \ldots, s_{n_s}^{(k-1)}$, which are collected into the sampling points matrix $\boldsymbol{S}_{k-1}$. Peherstorfer and Willcox (2015*b*) primarily focus on uniformly distributed sampling points. However, for transport-dominated problems, which can exhibit local features such as wave

fronts, uniform sampling is inefficient, as shown in Peherstorfer (2020). Instead, Peherstorfer (2020) proposes an adaptive sampling strategy.

Let us measure the discrepancy between two spaces spanned by the basis matrices $\bar{\boldsymbol{V}}$ and $\boldsymbol{V}_k$, respectively, via

$$d(\bar{\boldsymbol{V}}, \boldsymbol{V}_k) = \|\bar{\boldsymbol{V}} - \boldsymbol{V}_k \boldsymbol{V}_k^\top \bar{\boldsymbol{V}}\|_F^2, \tag{6.13}$$

where it is assumed for ease of exposition that $\boldsymbol{V}_k$ and $\bar{\boldsymbol{V}}$ are orthonormal. Let us consider the situation that we want to adapt the basis matrix $\boldsymbol{V}_k$ to $\boldsymbol{V}_{k+1}$ so that $\boldsymbol{V}_{k+1}$ spans a space that is close to the space spanned by $\bar{\boldsymbol{V}}$ in the sense of (6.13). Notice that we now consider the adaptation from time step k to $k+1$ instead of $k-1$ to k. The reason is that the sampling points adaptation is performed in the algorithm after the basis has been updated at time step k, and hence is in preparation for adapting the space at the subsequent time step $k+1$; see the algorithm given in Section 6.3.6.

When adapting from $\boldsymbol{V}_k$ to $\boldsymbol{V}_{k+1}$, the sliding window is $\boldsymbol{F}_k$. We assume for now that it is given as $\boldsymbol{F}_k = \bar{\boldsymbol{V}}\bar{\boldsymbol{F}}_k$ with $\bar{\boldsymbol{F}}_k$ full-rank and $w \geq n$. This means that the column span of the sliding window $\boldsymbol{F}_k$ is the space spanned by $\bar{\boldsymbol{V}}$. Consider the coefficient matrix $\boldsymbol{C}_k$ defined by (6.7), replacing k with $k-1$, and the residual matrix $\boldsymbol{R}_k = \boldsymbol{V}_k \boldsymbol{C}_k - \boldsymbol{F}_k$. Now let $\boldsymbol{S}$ be a valid sampling points matrix. Peherstorfer (2020) shows that the ADEIM rank-r update $\boldsymbol{V}_{k+1} = \boldsymbol{V}_k + \boldsymbol{\alpha}_k \boldsymbol{\beta}_k^\top$ via (6.8) with $\boldsymbol{F}_k$ and sampling points matrix $\boldsymbol{S}$ leads to the bound

$$d(\bar{\boldsymbol{V}}, \boldsymbol{V}_{k+1}) \leq \frac{1}{\sigma_{\min}^2(\boldsymbol{F}_k)} \left(\|\check{\boldsymbol{S}}^\top \boldsymbol{R}_k\|_F^2 + \sum_{i=r+1}^{\bar{r}} \sigma_i^2 \right), \tag{6.14}$$

where $\bar{r}$ is the rank of $\boldsymbol{S}^\top \boldsymbol{R}_k$, $\sigma_1 \geq \cdots \geq \sigma_{\bar{r}} > 0$ are the singular values of $\boldsymbol{S}^\top \boldsymbol{R}_k$, and $\sigma_{\min}(\boldsymbol{F}_k)$ is the smallest non-zero singular value of $\boldsymbol{F}_k$. Recall that $\check{\boldsymbol{S}}$ is the complementary sampling points matrix to $\boldsymbol{S}$ in the sense that it corresponds to the sampling points $\{1, \ldots, N\} \setminus \{s_1, \ldots, s_{n_s}\}$. The bound (6.14) motivates introducing the objective

$$b(\boldsymbol{S}) = \|\check{\boldsymbol{S}}^\top \boldsymbol{R}_k\|_F^2 + \sum_{i=r+1}^{\bar{r}} \sigma_i^2. \tag{6.15}$$

Selecting sampling points $\{s_1, \ldots, s_{n_s}\} \subseteq \{1, \ldots, N\}$ so that the corresponding matrix $\boldsymbol{S}$ minimizes the objective b incurs a combinatorial subset-selection problem in the full-model dimension N, which makes this direct approach intractable. Instead, Peherstorfer (2020) suggests selecting sampling points that minimize only the norm $\|\check{\boldsymbol{S}}^\top \boldsymbol{R}_k\|_F^2$ of the sparse residual $\check{\boldsymbol{S}}^\top \boldsymbol{R}_k$ corresponding to the complementary sampling points $\{1, \ldots, N\} \setminus \{s_1, \ldots, s_{n_s}\}$, which is one term in the objective b. Because the equality

$$\|\boldsymbol{R}_k\|_F^2 = \|\boldsymbol{S}^\top \boldsymbol{R}_k\|_F^2 + \|\check{\boldsymbol{S}}^\top \boldsymbol{R}_k\|_F^2$$

holds for any sampling points matrix $\boldsymbol{S}$ and its complementary sampling points matrix $\check{\boldsymbol{S}}$, finding a set of sampling points so that its complement minimizes $\|\check{\boldsymbol{S}}^\top \boldsymbol{R}_k\|_F^2$ is

equivalent to finding a set of sampling points that maximizes $\|\boldsymbol{S}^\top \boldsymbol{R}_k\|_F^2$. This can be achieved as follows. First, compute the Euclidean norm of all N rows of $\boldsymbol{R}_k$, then sort them in descending order, and then select the n_s indices corresponding to the n_s largest row norms. The selected indices are the new sampling points $s_1^{(k)}, \ldots, s_{n_s}^{(k)}$, with sampling points matrix $\boldsymbol{S}_k$.

Cortinovis, Kressner, Massei and Peherstorfer (2020) show that the approach based on maximizing the residual norm leads to an objective value (6.15) that is at most a factor two worse than the objective value obtained with the optimal sampling points selection that minimizes b. The costs of selecting the points via the residual norm are dominated by the costs of evaluating the full-model right-hand side function at all N components and then sorting the norms of the N rows. There is a range of other sampling strategies for ADEIM and related online adaptive model reduction techniques (Wentland, Duraisamy and Huang 2023, Zucatti and Zahr 2024, Mohaghegh and Huang 2026). For example, Wentland *et al.* (2023) focus on computational fluid-dynamics applications and propose physics-motivated strategies.

6.3.6. Online algorithm for ADEIM reduced models

There are several variants of ADEIM in the literature (Peherstorfer and Willcox 2015*b*, Peherstorfer 2020, Singh *et al.* 2023), but they share the same core steps:

(1) low-rank basis adaptation (Section 6.3.2),

(2) interpolation points update (Section 6.3.2),

(3) reduced time integration with empirical interpolation (Section 6.3.3),

(4) sparse full-model queries to populate the sliding window (Section 6.3.4), and

(5) sampling points adaptation (Section 6.3.5).

We summarize a generic online ADEIM procedure below. For ease of exposition we assume that the basis is adapted at every time step; in practice, steps (5) and/or (1)–(2) can be executed only intermittently (e.g. every zth step) to reduce online costs.

We follow the online ADEIM procedure given in Peherstorfer (2020, Algorithm 1). For the first $w_{\text{init}} \in \mathbb{N}$ time steps, the full model is integrated in time to generate the snapshots $\boldsymbol{Q}_{1:w_{\text{init}}} \in \mathbb{R}^{N \times w_{\text{init}}}$, from which the initial basis matrix $\boldsymbol{V}_{w_{\text{init}}}$, interpolation points matrix $\boldsymbol{P}_{w_{\text{init}}}$ and reduced right-hand side function $\hat{\boldsymbol{f}}^{(\Delta)}_{w_{\text{init}}}$ are constructed with empirical interpolation. Furthermore, the sliding window $\boldsymbol{F}_{w_{\text{init}}} = \boldsymbol{Q}_{w_{\text{init}}-w+1:w_{\text{init}}}$ is initialized with the snapshots. Then the following steps are iterated for $k = w_{\text{init}} + 1, \ldots, K$.

- *Step 1 (basis update).* Adapt ADEIM basis from $\boldsymbol{V}_{k-1}$ to $\boldsymbol{V}_k$ using the update (6.9) based on the sliding window $\boldsymbol{F}_{k-1}$ and sampling points matrix $\boldsymbol{S}_{k-1}$ (Section 6.3.2).

- *Step 2 (interpolation points update).* Compute the interpolation points matrix $\boldsymbol{P}_k$, for example by applying QDEIM to the adapted basis matrix $\boldsymbol{V}_k$ (Section 6.3.2).
- *Step 3 (forward step).* Solve the reduced model given by the right-hand side function $\hat{\boldsymbol{f}}_k^{(\Delta)}$ with adapted basis and interpolation points for $\hat{\boldsymbol{q}}_k$ and store the lifted reduced solution $\boldsymbol{V}_k\hat{\boldsymbol{q}}_k$ by extending $\boldsymbol{Q}_{1:k-1}$ to $\boldsymbol{Q}_{1:k}$. Note that $\hat{\boldsymbol{f}}_k^{(\Delta)}$ now depends on the step k because the underlying basis $\boldsymbol{V}_k$ and interpolation points matrix $\boldsymbol{P}_k$ depend on k (Section 6.3.3).
- *Step 4 (sliding window update) and optional Step 5 (sampling points update).* If the sampling points should be adapted at this time step k, then evaluate the full-model right-hand side function $\boldsymbol{f}^{(\Delta)}(\boldsymbol{V}_k\hat{\boldsymbol{q}}_k)$ at all N components and append it to the sliding window to obtain $\boldsymbol{F}_k$. Drop the oldest column from $\boldsymbol{F}_k$. Compute the coefficient matrix $\boldsymbol{C}_k$ and the residual matrix $\boldsymbol{R}_k = \boldsymbol{V}_k\boldsymbol{C}_k - \boldsymbol{F}_k$ and select new sampling points $\boldsymbol{S}_k$ based on row-wise residual norms (Section 6.3.5). Otherwise, if the sampling points should not be updated at this time step k, then evaluate $\boldsymbol{f}^{(\Delta)}(\boldsymbol{V}_k\hat{\boldsymbol{q}}_k)$ only at the sampling points corresponding to $\boldsymbol{S}_{k-1}$ and update the sliding window $\boldsymbol{F}_k$ with (6.11)–(6.12). Then set $\boldsymbol{S}_k = \boldsymbol{S}_{k-1}$.

The computational cost of an online time step with an ADEIM reduced model is typically dominated by evaluating the full-model right-hand side function used to update the sliding window. At time steps where the sampling points are adapted, the full-model right-hand side function is evaluated at all N components, which results in online costs that scale at least linearly in N, but this is typically still cheaper than solving the nonlinear system (6.1) corresponding to the full model at this time step. At all other time steps, the full-model right-hand side function is evaluated only at $n_s \ll N$ sampling points, so the costs of these sparse full-model queries scale with n_s rather than N.

7. Instantaneous residual minimization methods

We now consider generic parametrizations (4.1) with a time- and parameter-dependent weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ that enters nonlinearly, possibly in combination with an offline weight vector as in (4.7). Generic means here that we do not restrict the parametrization to have a specific structural form for how the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ enters. This stands in contrast to the transformation-based methods discussed in Section 5 and the online adaptive basis methods in Section 6, where the weight vector enters in a specific, prescribed way that is leveraged algorithmically for efficient computation. To derive reduced dynamics for such generic nonlinear parametrizations, we review instantaneous residual minimization methods. The central idea is to determine, at each time t, the time derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ of the weight vector by minimizing the residual norm.

7.1. Overview of this section

We present the Dirac–Frenkel variational principle as a core building block for instantaneous residual minimization with generic nonlinear parametrizations in Section 7.2, and review numerical methods such as Neural Galerkin schemes that build upon this principle. The derivation and practical implementation of methods based on the Dirac–Frenkel variational principle involve several numerical considerations. First, the residual norm has to be approximated numerically. We review static and adaptive collocation approaches in the context of Neural Galerkin schemes in Section 7.3. Second, the evolution equations for the weight vector induced by the residual minimization need to be discretized in time. We briefly consider time-stepping choices and their implications in Section 7.4. Third, instantaneous residual minimization may be insufficient to fully determine the weight vector, leading to singular or poorly conditioned systems. We therefore discuss regularization strategies for such cases in Section 7.5. Fourth, the residual minimization leads to a system of algebraic equations given by the first-order optimality conditions, which must be solved numerically at each time step. We outline solvers based on randomized sketching to reduce costs, and we discuss how randomization can also improve conditioning in Section 7.5.

We further relate the Dirac–Frenkel variational principle to optimize-then-discretize schemes and discuss discretize-then-optimize schemes for instantaneous residual minimization as an alternative to the Dirac–Frenkel variational principle in Section 7.6. In Section 7.7 we consider instantaneous residual minimization in the context of nonlinear parametrizations that have been trained on snapshot data, including autoencoder-based approaches and neural representations.

7.2. The Dirac–Frenkel variational principle

A common way to formulate instantaneous residual minimization is via the Dirac–Frenkel variational principle. Its origin can be traced back to Dirac (1930) and Frenkel (1934). More modern presentations can be found in McLachlan (1964), Kramer and Saraceno (1981), Broeckhove, Lathouwers, Kesteloot and Van Leuven (1988) and Lubich (2008). A short history of the Dirac–Frenkel variational principle is reported in Lasser and Lubich (2020, Section 3.8).

7.2.1. The Dirac–Frenkel variational principle

A nonlinear parametrization $\hat{q}(\boldsymbol{\theta}, \cdot)\colon \Omega \to \mathbb{R}$ with weight vector $\boldsymbol{\theta} \in \mathbb{R}^n$ (which could depend on time t and parameter $\boldsymbol{\mu}$) induces the set $\widehat{\mathcal{M}}$ defined in (4.3), which contains all functions that can be represented by varying the weight vector in $\mathbb{R}^n$. Note that we do not explicitly denote the possible dependence of $\hat{q}$ on an offline weight vector for now. Recall that $\mathcal{M}$ denotes the solution manifold defined in (2.5), whereas we denote the set (4.3) as $\widehat{\mathcal{M}}$. Even though we do not restrict the parametrizations to have a specific structural form, we do need to make suitable regularity assumptions, for example, $\hat{q}$ is differentiable in the weight vector $\boldsymbol{\theta}$ and

all functions $\hat{q}(\boldsymbol{\theta}, \cdot) \in \widehat{\mathcal{M}}$ are also in the space $\mathcal{Q}$ over which the PDE problem is formulated; see Section 2.1.

Let us now interpret $\widehat{\mathcal{M}}$ as a trial manifold for solving the PDE problem (2.1) and plug the nonlinear parametrization $\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)$, with time- and parameter-dependent weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$, into the PDE (2.1) to obtain the residual function $r \colon \mathbb{R}^n \times \mathbb{R}^n \times \Omega \to \mathbb{R}$ for a fixed $\boldsymbol{\mu}$ as

$$r(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}), \cdot) = \nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)^\top \dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}) - f(\cdot, \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot); \boldsymbol{\mu}). \tag{7.1}$$

We used the chain rule

$$\partial_t \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot) = \sum_{i=1}^{n} \partial_{\theta_i} \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot) \dot{\theta}_i(t, \boldsymbol{\mu}) = \nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)^\top \dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$$

to make the dependence of the time derivative $\partial_t \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)$ on the time derivative of the weight vector $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ more explicit.

The Dirac–Frenkel variational principle seeks a time derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}) \in \mathbb{R}^n$ of the weight vector such that the residual is orthogonal to all admissible variations of $\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)$ at time t. If $\widehat{\mathcal{M}}$ has a suitable manifold structure induced by the regularity of the parametrization, then this is equivalent to

$$\langle v, r(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}), \cdot) \rangle = 0 \quad \text{for all } v \in \mathcal{T}_{\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)} \widehat{\mathcal{M}}, \tag{7.2}$$

where $\mathcal{T}_{\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)} \widehat{\mathcal{M}}$ denotes the tangent space of $\widehat{\mathcal{M}}$ at $\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)$. Notice that conditions (7.2) do not necessarily determine a unique time derivative if the dimension of the space $\mathcal{T}_{\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)} \widehat{\mathcal{M}}$ is less than the number n of weights – a fact that we will revisit in Section 7.5 and that we have already encountered in Section 5.3.4.

The choice of the inner product $\langle \cdot, \cdot \rangle$ of the orthogonality conditions (7.2) is critical. Out of convenience, a common choice is the L^2 inner product; however, the L^2 inner product may be inappropriate for some problems in the sense that the orthogonality conditions on the residual do not control the error in the norm of interest. At the same time, numerically evaluating the orthogonality conditions formulated with other inner products can be challenging. We refer to Bachmayr, Dahmen and Oster (2025*a*) for an in-depth discussion.

The orthogonality conditions (7.2) are instantaneous in the sense that they act locally in time: at each time t, a derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ is chosen that sets the residual orthogonal to all elements in the tangent space $\mathcal{T}_{\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)} \widehat{\mathcal{M}}$, which defines an evolution equation for the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$. The instantaneous conditions on the residual stand in contrast to global-in-time approaches, such as tensor-based space–time approaches (Nouy 2010) and physics-informed neural networks (Raissi *et al.* 2019), which determine a global approximation by minimizing a residual-based objective over the space–time domain. A detailed comparison between instantaneous (or sequential-in-time) and global-in-time methods in the context of nonlinear parametrizations can be found in Zhang *et al.* (2024); for a discussion

in the context of linear model reduction we refer to Glas, Mayerhofer and Urban (2017).

Remark 7.1. While the Dirac–Frenkel variational principle provides a natural and widely used mechanism for deriving reduced dynamics, it represents only one particular choice within a broader geometric setting. Buchfink, Glas, Haasdonk and Unger (2024*b*) developed a differential–geometric framework for nonlinear model reduction that adopts a more general viewpoint. In this framework, reduced models are not required to arise from the Dirac–Frenkel variational principle; instead, more general reduction maps defined on the tangent bundle of the solution manifold are admitted, which can serve as a blueprint for structure-preserving model reduction using nonlinear parametrizations.

7.2.2. Dirac–Frenkel and instantaneous residual minimization

Given the residual function (7.1), instantaneous residual minimization means minimizing the residual norm at the current time t,

$$\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu}) \in \operatorname*{arg\,min}_{\boldsymbol{\eta}\in\mathbb{R}^n} \|r(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\boldsymbol{\eta},\cdot)\|^2, \tag{7.3}$$

where $\|\cdot\|$ is the norm induced by the inner product $\langle\cdot,\cdot\rangle$. Setting the first-order optimality condition of (7.3) to zero, that is,

$$\nabla_{\boldsymbol{\eta}}\|r(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\boldsymbol{\eta},\cdot)\|^2 = 0,$$

leads to the system of equations

$$\langle \partial_{\theta_i}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot), r(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\boldsymbol{\eta},\cdot)\rangle = 0, \quad i = 1,\dots,n. \tag{7.4}$$

Now notice that if $\widehat{\mathcal{M}}$ is equipped with a suitable manifold structure, then the tangent space at $\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)$ is spanned by the partial derivatives of $\hat{q}$ in all directions of the weight vector components,

$$\mathcal{T}_{\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)}\widehat{\mathcal{M}} = \operatorname{span}\{\partial_{\theta_1}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot),\dots,\partial_{\theta_n}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)\}. \tag{7.5}$$

Thus conditions (7.4) derived from the instantaneous residual norm minimization (7.3) are equivalent to the conditions imposed by the Dirac–Frenkel variational principle (7.2).

If the parametrization is linear, i.e. of the form (4.2), then (7.4) is equivalent to

$$\langle \varphi_i, \partial_t\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot) - f(\cdot,\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot);\boldsymbol{\mu})\rangle = 0, \quad i = 1,\dots,n,$$

which, provided that the basis functions φ_i are orthonormal, corresponds (in coordinates) exactly to the Galerkin reduced model (2.12). Consequently, the Dirac–Frenkel variational principle can be interpreted as a generalization of the Galerkin orthogonality condition.

Remark 7.2. The Dirac–Frenkel variational principle, and more generally instantaneous residual minimization, provides a unifying framework for deriving

reduced dynamics on nonlinear trial manifolds. In particular, several methods discussed in the sections on transformation-based reduction (Section 5) and online adaptive bases (Section 6) can be interpreted as instances of instantaneous residual minimization. Nevertheless, when the nonlinear parametrization has additional exploitable structure, the reduced dynamics can often be written in a more specific form and solved more efficiently (e.g. matrix factorizations in dynamic low-rank approximations or shift parameters in transformation-based model reduction). We therefore treat such more structured nonlinear parametrizations separately in this survey.

7.2.3. Literature overview of instantaneous residual minimization with nonlinear parametrizations

There is a vast literature on the Dirac–Frenkel variational principle and its applications in science and engineering. In this section we focus on works that develop schemes based on the Dirac–Frenkel variational principle for parametrizations that are not necessarily trained offline on snapshot data, i.e. approaches that are closer in spirit to PDE solvers. From the perspective of nonlinear model reduction, however, methods that apply the Dirac–Frenkel principle to trained nonlinear parametrizations are of particular interest; these are reviewed separately in Section 7.7.

The computational chemistry community has developed methods based on the Dirac–Frenkel variational principle for numerically solving the Schrödinger equation, for example (Heller 1976, Meyer, Manthe and Cederbaum 1990, Beck, Jäckle, Worth and Meyer 2000, Lubich 2005, 2008). The Dirac–Frenkel variational principle is also used in dynamic low-rank approximations and dynamic orthogonal decompositions with matrix factorizations as parametrizations, which we survey in the context of time-dependent basis methods in Section 6.2.

The moving finite element method of Miller and Miller (1981) parametrizes solutions by finite element coefficients together with time-dependent nodal positions, and their evolution equations are obtained with the Dirac–Frenkel variational principle. But Dirac–Frenkel can be applied to more generic nonlinear parametrizations. For example, Anderson and Farazmand (2022) use Dirac–Frenkel dynamics with morphing-shape parametrizations. Neural-network parametrizations are of particular interest due to their expressivity, as studied in the context of Dirac–Frenkel schemes in Lee and Carlberg (2020) and Du and Zaki (2021). In a similar vein, the Neural Galerkin schemes introduced in Bruna, Peherstorfer and Vanden-Eijnden (2024) formulate Dirac–Frenkel evolution equations for the weights of neural-network parametrizations, which have led to follow-up works on high-dimensional problems (Wen, Vanden-Eijnden and Peherstorfer 2024), nonlinear model reduction (Berman and Peherstorfer 2024, Weder, Schwerdtner and Peherstorfer 2025) and fast and structure-preserving online solvers (Berman and Peherstorfer 2023, Schwerdtner, Schulze, Berman and Peherstorfer 2024). Related ideas have also been used for filtering in Aghili *et al.* (2025). Various aspects and challenges of using the Dirac–Frenkel variational principle with neural networks have been addressed: Kast

and Hesthaven (2024) propose using a Laplace–Beltrami positional embedding to handle geometrically complex domains, while Finzi, Potapczynski, Choptuik and Wilson (2023) introduced a re-training step to improve numerical stability; Zhao *et al.* (2024) have developed a variational Monte Carlo-type sampling estimator to model Dirac–Frenkel parameter dynamics for high-dimensional PDE problems. An important variant of the Dirac–Frenkel variational principle when applied to generic nonlinear parametrization such as neural networks is its regularized version, which is systematically studied in Feischl, Lasser, Lubich and Nick (2024) and Lubich and Nick (2025), and which we revisit in Section 7.5.

7.3. Neural Galerkin schemes

We now discuss Neural Galerkin schemes that instantiate the Dirac–Frenkel variational principle using neural-network parametrizations (Berman, Schwerdtner and Peherstorfer 2024, Bruna *et al.* 2024). We focus on one particular numerical aspect, which is central to Neural Galerkin schemes as well as other schemes based on the Dirac–Frenkel variational principle: the orthogonality conditions given in (7.2) involve inner products that must be evaluated numerically. Neural Galerkin schemes approximate these inner products, or equivalently the norm in the residual minimization formulation, via quadrature or collocation (sampling) over the spatial domain.

7.3.1. Neural Galerkin: An instantiation of the Dirac–Frenkel variational principle

Motivated by neural-network parametrizations, Bruna *et al.* (2024) introduce Neural Galerkin schemes that impose the Dirac–Frenkel dynamics on the neural-network weights $\boldsymbol{\theta}(t, \boldsymbol{\mu})$. This means that the neural-network weights depend on time and are evolved following the Dirac–Frenkel variational principle. We remark that the earlier work of Du and Zaki (2021) also uses the Dirac–Frenkel variational principle in the context of neural-network parametrizations.

From a numerical perspective, Neural Galerkin schemes directly consider instantaneous residual norm minimization (7.3), so that the time derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ is a solution of the time-dependent least-squares problem

$$\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}) \in \arg\min_{\boldsymbol{\eta} \in \mathbb{R}^n} \| \nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot)^\top \boldsymbol{\eta} - f(\cdot, \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot); \boldsymbol{\mu}) \|^2, \tag{7.6}$$

which is the same optimization problem as (7.3) except that the residual function (7.1) is written out explicitly. The formulation (7.6) highlights that this is a least-squares problem that is linear in the unknown $\boldsymbol{\eta}$, even though a nonlinear parametrization $\hat{q}$ is used to represent the PDE solution. Furthermore, the formulation (7.6) makes explicit that the time derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ is the coefficient vector of the best approximation (projection) of the right-hand side function onto the space (7.5) spanned by the component functions of the gradient, as visualized in Figure 7.1.

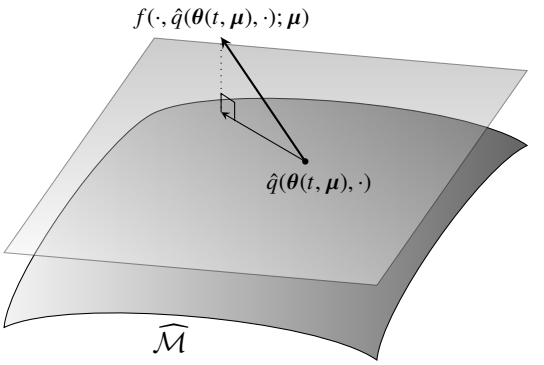

Figure 7.1. The Dirac–Frenkel variational principle evolves the weight vector $\boldsymbol{\theta}(t,\boldsymbol{\mu})$ by instantaneous residual minimization: it chooses the time derivative $\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})$ as the coordinates of the orthogonal projection of the right-hand side function $f(\cdot, \hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot);\boldsymbol{\mu})$ onto the tangent space of the trial manifold $\widehat{\mathcal{M}}$ at the current solution $\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)$.

The normal equations of the least-squares problem (7.6) lead to a dynamical system for $\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})$,

$$\boldsymbol{E}(\boldsymbol{\theta}(t,\boldsymbol{\mu}))\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu}) = \boldsymbol{G}(\boldsymbol{\theta}(t,\boldsymbol{\mu});\boldsymbol{\mu}), \tag{7.7}$$

with matrices

$$\boldsymbol{E}_{ij}(\boldsymbol{\theta}(t,\boldsymbol{\mu})) = \langle \partial_{\theta_i}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot), \partial_{\theta_j}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)\rangle, \quad i,j = 1,\dots,n, \tag{7.8}$$

$$\boldsymbol{G}_i(\boldsymbol{\theta}(t,\boldsymbol{\mu});\boldsymbol{\mu}) = \langle \partial_{\theta_i}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot), f(\cdot,\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot);\boldsymbol{\mu})\rangle, \quad i = 1,\dots,n. \tag{7.9}$$

The dynamical system (7.7) can be integrated in time to obtain a weight trajectory $\boldsymbol{\theta}(t,\boldsymbol{\mu})$ so that at each point in time the orthogonality conditions (7.4) are satisfied. Alternatively, the least-squares problem (7.6) can be integrated in time, which typically leads to better conditioned intermediate numerical problems at each time step than solving the normal equations (Berman *et al.* 2024). We remark once more that (7.7) can be underdetermined; see the forthcoming Section 7.5.

7.3.2. Estimating inner products with sampling points

In this section we discuss collocation strategies for numerically estimating the norm in the least-squares problem (7.6) in the context of Neural Galerkin schemes.

The need for numerically estimating inner products. Consider the least-squares formulation (7.6) corresponding to the Neural Galerkin schemes. The objective of the least-squares problem is formulated via the norm $\|\cdot\|$, which is induced by the inner product $\langle\cdot,\cdot\rangle$ that underlies the residual orthogonality conditions (7.2) given by the Dirac–Frenkel variational principle. Thus the objective has to be numerically evaluated to solve the least-squares problem for $\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})$.

As Berman *et al.* (2024) point out, there is an analogue in numerical analysis: in finite element methods, assembling Galerkin systems requires evaluation of inner products of basis functions to compute entries of mass and stiffness matrices. For linear parametrizations, such as linear combinations of local basis functions that are centred at fixed grid points, these inner products can often be pre-computed and reused efficiently (Ern and Guermond 2004). In contrast, for nonlinear parametrizations, this pre-computation is no longer possible because the test space changes over time with the weights of the parametrization, and so the quantities in the inner products change over time and cannot be directly re-used. Furthermore, because of the nonlinearity of the parametrizations, the superposition principle is lost, and so the residual norm cannot be obtained from separate components.

There are situations when the Dirac–Frenkel variational principle is applied, where there is a closed form of the residual so that it can be evaluated exactly without numerical computations. For example, in some instances, nonlinear parametrizations based on Gaussian wave packets can be chosen such that they lead to closed-form residual norms for variants of the Schrödinger equations; see e.g. Lubich (2008) and Lasser and Lubich (2020). In other cases symbolic computations can be used, as by Anderson and Farazmand (2024). However, in many settings, especially when the nonlinear parametrizations are trained on snapshot data in an offline phase as in model reduction, there is no closed-form solution, so the residual norm must be computed numerically.

Estimating inner products via static collocation points. For the numerical evaluation of the norm in (7.6), let us consider collocation points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \Omega$ that have been sampled from a measure ν supported on Ω. For example, if ν is the uniform measure over Ω, then the points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ are sampled uniformly in Ω. We refer to the collocation points as static because they are fixed over time, and additionally independent of the parameter $\boldsymbol{\mu}$.

The collocation points are used to estimate the norm in (7.6), which leads to the optimization problem with the empirical objective

$$\min_{\boldsymbol{\eta} \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} |\nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \boldsymbol{x}_i)^\top \boldsymbol{\eta} - f(\boldsymbol{x}, \hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot); \boldsymbol{\mu})|_{\boldsymbol{x}=\boldsymbol{x}_i}|^2. \tag{7.10}$$

If the points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ are sampled from ν, then the empirical objective in (7.10) is a Monte Carlo estimator. Estimating the objective (7.6) with a Monte Carlo estimator (7.10) is analogous to situations found in machine learning, where population loss functions are estimated with empirical loss functions (Vapnik 1991).

Numerical quadrature offers an alternative to Monte Carlo estimation. The points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ are selected based on some quadrature rule that accounts for the measure ν. Using a quadrature rule leads to estimators analogous to (7.10), except that there can be weights other than $1/m$. Numerical quadrature for evaluating inner products in the context of the Dirac–Frenkel variational principle is considered in, for example, Du and Zaki (2021) and Schwerdtner *et al.* (2024).

$$\boldsymbol{\theta}(t_0) \longrightarrow \boldsymbol{\theta}(t_1) \longrightarrow \boldsymbol{\theta}(t_2) \longrightarrow \cdots$$

$$\{\boldsymbol{x}_i(t_0)\}_{i=1}^m \longrightarrow \{\boldsymbol{x}_i(t_1)\}_{i=1}^m \longrightarrow \{\boldsymbol{x}_i(t_2)\}_{i=1}^m \longrightarrow \cdots$$

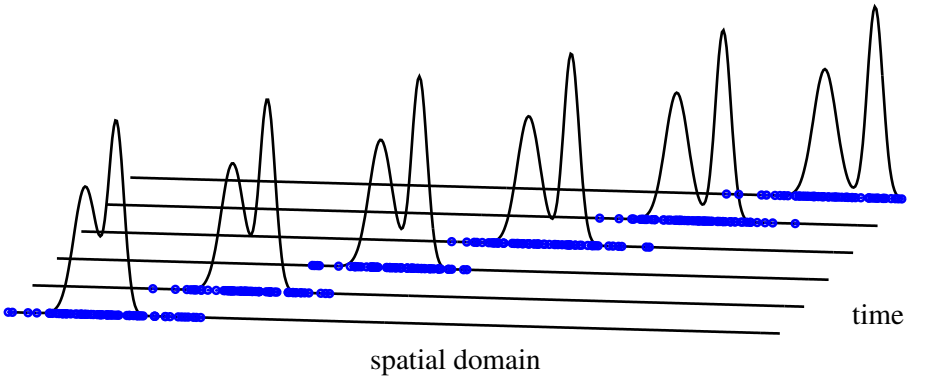

Figure 7.2. Neural Galerkin schemes evolve the weight vector as prescribed by the Dirac–Frenkel variational principle, and can simultaneously adapt sampling points corresponding to a potential informed by the residual. This adaptation concentrates the sample points in regions that dominate the residual and so yield more efficient residual-norm estimates than static sampling.

7.3.3. Adaptive sampling points

Neural Galerkin schemes support evolving the sampling points together with the weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$; see Figure 7.2.

Limitations of static collocation points for transport-dominated problems. Returning to the motivating example in Figure 2.1 from Section 2.5, it is common that transport-dominated problems exhibit moving wave fronts or other local, coherent features – sharp solution gradients, bumps – that travel through the spatial domain.

When using a set of static collocation points, which means that they are fixed for all times t, then these points must be chosen so that the moving local features are adequately resolved at all times. However, ensuring uniform accuracy at all times becomes difficult because the local feature may traverse any and all regions of the spatial domain, so a fine resolution across the whole spatial domain is needed, which often requires a large number of sampling points (Bruna *et al.* 2024, Wen *et al.* 2024). In the linear transport example shown in Figure 2.1, a uniform static sampling must include points in all parts of the domain at all times to track the narrow Gaussian bump as it moves. This is the case even though most of these points, at most time steps, will lie in regions where the solution is close to zero, and so contribute little useful information while still requiring gradient and right-hand side evaluations to empirically estimate the objective of (7.6), as in (7.10). Consequently, relying

on static sampling points can require large numbers of sampling points to resolve moving local features.

Remark 7.3. A similar sampling challenge can be found in physics-informed machine learning methods that aim to minimize the residual norm over spatial (and time) domains. If solution features are local, then a large number of collocation points, and thus residual-norm evaluations, are needed, which can become a computational bottleneck (Nabian, Gladstone and Meidani 2021, Rotskoff, Mitchell and Vanden-Eijnden 2022, Wu *et al.* 2023).

Estimating inner products with adaptive collocation points. Estimating inner products with time-adaptive collocation points offers one method to keep the number of collocation points low even if solution features are local. Wen *et al.* (2024) have formulated Neural Galerkin schemes via time-dependent measures $\nu_{\boldsymbol{\theta}(t,\boldsymbol{\mu}),\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})}$ that are allowed to change with the weights $\boldsymbol{\theta}(t,\boldsymbol{\mu})$ and the derivative $\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})$. Adaptive sampling in the context of nonlinear model reduction is also addressed by Bon, Caris and Mula (2025), who propose a sampling criterion that directly targets conditioning of the reduced dynamics, and Black *et al.* (2021*a*), who update interpolation points over time for the efficient estimation of nonlinear terms. Also relevant is the work of Gruhlke, Nouy and Trunschke (2024), who introduce sampling for natural gradient descent, which corresponds to analogue dynamics as given by Neural Galerkin schemes (Zhang *et al.* 2024). We emphasize that in numerical analysis, the adaptive choice of sampling points in the context of PDE approximations is used for instance in moving finite element methods (Miller and Miller 1981).

Following Wen *et al.* (2024), we formulate the orthogonality conditions via a time-dependent measure, which leads to the analogous least-squares problem of (7.6) except that the L^2-norm $\|\cdot\|_{\nu_t}$ now depends on ν_t, that is,

$$\min_{\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})\in\mathbb{R}^n} \|\nabla_{\boldsymbol{\theta}}\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)^{\top}\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu}) - f(\cdot,\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot);\boldsymbol{\mu})\|_{\nu_t}^2,$$

where we denote $\nu_{\boldsymbol{\theta}(t,\boldsymbol{\mu}),\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})}$ as ν_t for brevity. Wen *et al.* (2024) show that if the residual can be set to zero with respect to a static measure ν, then the optimum is invariant to changes in the measure under standard conditions, such as the measures being fully supported on the spatial domain Ω. It is further argued in Wen *et al.* (2024) that the residual being close to zero is a reasonable assumption because we are typically interested in cases where the parametrization is so rich that the residual can be kept small. We note that alternatively to formulating the objective in a time-dependent norm, one can derive an importance-sampling estimator of the objective with a time-dependent biasing measure while keeping the base measure fixed; we refer to Bruna *et al.* (2024), who discuss such an importance-sampling approach.

One major question is how to propagate the measures ν_t forward in time so that samples from ν_t lead to accurate estimators of the residual norm. Wen *et al.* (2024)

propose letting the residual itself drive the evolution of ν_t by defining ν_t as a Gibbs measure

$$\nu_t \propto \exp\left(-U_{\boldsymbol{\theta}(t,\boldsymbol{\mu}),\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})}\right)$$

with the potential $U_{\boldsymbol{\theta},\dot{\boldsymbol{\theta}}}$ taken as the squared residual, that is,

$$U_{\boldsymbol{\theta}(t,\boldsymbol{\mu}),\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})}(\boldsymbol{x}) = |r(\boldsymbol{\theta}(t,\boldsymbol{\mu}), \dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu}), \boldsymbol{x})|^2. \tag{7.11}$$

High-probability regions of the measure ν_t correspond to spatial locations with large absolute residual. Other potentials could be used, for example by directly sampling from the solution function if it is a probability density function (Bruna *et al.* 2024) or by building on other sampling criteria such as the stability-based criteria introduced in Bon *et al.* (2025).

A formal way to evolve the time-dependent measure is given via the Fokker–Planck equation driven by the potential (7.11),

$$\partial_t \nu_t = \gamma \nabla \cdot \left(\nabla \nu_t + \nu_t \nabla U_{\boldsymbol{\theta}(t,\boldsymbol{\mu}),\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})}\right), \tag{7.12}$$

which can be coupled to the weight dynamics,

$$\begin{aligned} \boldsymbol{E}_t(\boldsymbol{\theta}(t,\boldsymbol{\mu}))\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu}) &= \boldsymbol{G}_t(\boldsymbol{\theta}(t,\boldsymbol{\mu});\boldsymbol{\mu}), \\ \partial_t \nu_t &= \gamma \nabla \cdot \left(\nabla \nu_t + \nu_t \nabla U_{\boldsymbol{\theta}(t,\boldsymbol{\mu}),\dot{\boldsymbol{\theta}}(t,\boldsymbol{\mu})}\right), \end{aligned} \tag{7.13}$$

where the matrices $\boldsymbol{E}_t$ and $\boldsymbol{G}_t$ given in (7.8) and (7.9), respectively, now also depend on time because they are formulated with respect to the inner product associated with ν_t. The parameter γ controls the time scale on which the measure ν_t adapts relative to the weights $\boldsymbol{\theta}(t,\boldsymbol{\mu})$.

Crucially, it is unnecessary to solve the Fokker–Planck equation (7.12) explicitly because only samples are needed from the density corresponding to ν_t. Thus the measure ν_t can be approximated via the empirical measure

$$\hat{\nu}_t = \frac{1}{m}\sum_{i=1}^{m} \delta_{\boldsymbol{x}_i(t)}$$

given by time-dependent samples $\boldsymbol{x}_1(t), \ldots, \boldsymbol{x}_m(t)$. Because the potential (7.11) is given via the residual, one can evaluate the gradient of the residual to obtain the score $\nabla \log \nu_t$ of ν_t and therefore advance the samples $\boldsymbol{x}_1(t), \ldots, \boldsymbol{x}_m(t)$ over time with particle methods rather than solving the Fokker–Planck equation (7.12). Wen *et al.* (2024) propose using Langevin samplers and Stein variational gradient descent, both of which propagate samples via the score $\nabla \log \nu_t$. In this way, the weight trajectory $\boldsymbol{\theta}(t,\boldsymbol{\mu})$ and the set of samples $\{\boldsymbol{x}_i(t)\}_{i=1}^m$ are advanced together in time, approximating the coupled dynamics (7.13), as illustrated in Figure 7.2.

7.4. Semi- and fully discrete instantaneous residual minimization

We now discuss semi-discrete and fully discrete variants of the least-squares problem resulting from the Dirac–Frenkel variational principle, where we follow the formulation and terminology used by Neural Galerkin schemes.

Semi-discrete least-squares problem. Given m sampling points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \Omega$, which can be obtained statically as in Section 7.3.2 or adaptively as in Section 7.3.3, consider the so-called batch gradient

$$\boldsymbol{J}(\boldsymbol{\theta}) = \begin{bmatrix} \nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}, \boldsymbol{x}_1)^\top \\ \vdots \\ \nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}, \boldsymbol{x}_m)^\top \end{bmatrix} \in \mathbb{R}^{m \times n}, \tag{7.14}$$

which has as rows the transpose of the gradient $\nabla_{\boldsymbol{\theta}} \hat{q}(\boldsymbol{\theta}, \cdot)$ of the nonlinear parametrization $\hat{q}$ evaluated at the m sampling points. The name batch gradient stems from the consideration of approximating the objective as in (7.10), where the $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ are the current batch of samples, which can in principle change over the time steps (and optimization iterations in the time-discrete setting; see below). Analogously, we define the batch right-hand side function as

$$\boldsymbol{F}(\boldsymbol{\theta}; \boldsymbol{\mu}) = \begin{bmatrix} f(\boldsymbol{x}, \hat{q}(\boldsymbol{\theta}, \cdot); \boldsymbol{\mu})|_{\boldsymbol{x}=\boldsymbol{x}_1} \\ \vdots \\ f(\boldsymbol{x}, \hat{q}(\boldsymbol{\theta}, \cdot); \boldsymbol{\mu})|_{\boldsymbol{x}=\boldsymbol{x}_m} \end{bmatrix} \in \mathbb{R}^m. \tag{7.15}$$

Using the batch gradient and the batch right-hand side function, we can write the semi-discrete instantaneous residual minimization problem as

$$\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}) \in \arg\min_{\boldsymbol{\eta} \in \mathbb{R}^n} \|\boldsymbol{J}(\boldsymbol{\theta}(t, \boldsymbol{\mu}))\boldsymbol{\eta} - \boldsymbol{F}(\boldsymbol{\theta}(t, \boldsymbol{\mu}); \boldsymbol{\mu})\|_2^2, \tag{7.16}$$

which is a matrix representation of the empirical objective (7.10) on the sample points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ up to a constant scaling factor. Note that the norm $\|\cdot\|_2$ in (7.16) denotes the Euclidean vector norm, which can be readily evaluated. It is important to notice that the least-squares problem (7.16) is linear in the unknown, a property that one often wants to preserve when discretizing in time.

Time discretization. Consider the discrete time steps $0 = t_0 < t_1 < \cdots < t_K = T$ with constant time-step size $\tau > 0$. An explicit time discretization of the least-squares problem (7.6) with Runge–Kutta methods leads to a sequence of least-squares problems that are also linear in the unknown. For example, using the explicit Euler method leads to linear least-squares problems at every time step $k = 1, \ldots, K$, namely

$$\min_{\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}) \in \mathbb{R}^n} \|\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}))\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}) - \boldsymbol{F}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}); \boldsymbol{\mu})\|_2^2, \tag{7.17}$$

with the update

$$\boldsymbol{\theta}_k(\boldsymbol{\mu}) = \boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}) + \tau\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}). \tag{7.18}$$

Thus, if discretized with an explicit scheme, only a linear least-squares problem has to be solved at each time step.

In contrast, implicit time discretization leads to nonlinear optimization problems at each time step. Using the implicit Euler method for demonstration purposes, we

obtain

$$\min_{\delta\boldsymbol{\theta}_k(\boldsymbol{\mu})\in\mathbb{R}^n} \|\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}) + \tau\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}))\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}) - \boldsymbol{F}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}) + \tau\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}); \boldsymbol{\mu})\|_2^2, \quad (7.19)$$

where the update $\delta\boldsymbol{\theta}_k(\boldsymbol{\mu})$ is applied as in (7.18). In the case of an implicit time discretization, the time-discrete least-squares problem becomes nonlinear in the update $\delta\boldsymbol{\theta}_k(\boldsymbol{\mu})$ because both the batch gradient and the batch right-hand side function are evaluated at the next iterate $\boldsymbol{\theta}_k(\boldsymbol{\mu})$. Thus, numerically solving (7.19) amounts to solving a nonlinear least-squares problem in the update $\delta\boldsymbol{\theta}_k(\boldsymbol{\mu})$, and therefore inherits the typical difficulties of fitting a nonlinear parametrization such as a neural network. In particular, the fitting has to be done at each time step. Consequently there is a strong interest in explicit time integration, which preserves the linearity of the least-squares problem in discrete time for many Dirac–Frenkel-based methods.

7.5. Regularization and randomized time integration of Dirac–Frenkel dynamics

As mentioned before, the least-squares problems arising from the orthogonality conditions in the Dirac–Frenkel variational principle can become poorly conditioned or even rank-deficient. A particular example was presented in the context of transformation-based methods in Section 5.3.4. In general, the ill-conditioning motivates the use of regularization. We consider explicit regularization via Tikhonov and truncated SVD. We also briefly discuss randomized time integration via right sketching, which acts as a form of regularization while additionally reducing the number of unknowns solved for, at each time step, by restricting the update to a low-dimensional random subspace.

7.5.1. Poor conditioning of least-squares problem and tangent space collapse

Let us consider the time-discrete least-squares problem given in (7.17), which is obtained by applying the explicit Euler method to the time-continuous problem given in (7.16).

A major challenge for schemes building on Dirac–Frenkel is numerically solving the least-squares problem. Besides computational costs, the least-squares problem can be challenging to solve numerically because of poor conditioning. The conditioning of the least-squares problem (7.17) is dominated by the condition number of the batch gradient $\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}))$ at the weight vector $\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu})$. For nonlinear parametrizations, the batch gradient can become poorly conditioned or even (numerically) rank-deficient, in which case the least-squares problem is underdetermined, and the weight update is not unique. Furthermore, rank-deficiency is closely related to the accuracy in Dirac–Frenkel methods: error bounds show that the error is controlled by the projection error incurred when projecting the right-hand side onto the column space of the batch gradient (equivalently, the component functions of the gradient in continuous time or the tangent space). When the batch gradient is rank-deficient and the corresponding space onto which the right-hand

side is projected cannot represent it well, then the residual and approximation error can be large, leading to low accuracy (Lubich 2005).

But rank-deficiency has broader implications in the context of the Dirac–Frenkel variational principle. The component functions of the gradient, which span the tangent space under suitable regularity assumptions and which correspond to the columns of the batch gradient $\boldsymbol{J}$ after discretization, also span the test space against which the residual is set orthogonal with conditions (7.4). Thus rank-deficiency also means that the Dirac–Frenkel conditions enforce orthogonality only with respect to a low-dimensional test space so that residual components outside the low-dimensional test space are uncontrolled and can grow, leading to a deterioration of accuracy despite an accurate intermediate approximation at a given time. In the context of the Dirac–Frenkel variational principle, this degeneracy phenomenon is known as the matrix singularity problem (Sawada, Heather, Jackson and Metiu 1985, Kay 1989, Rowan *et al.* 2020). Zhang *et al.* (2024) refer to it as tangent space collapse because the space spanned by the component functions of the gradient becomes low-dimensional and they span the tangent space under suitable conditions.

Regularization is a natural remedy and important in practice, but a major difficulty is that one does not solve a single, isolated least-squares problem here but instead one solves a sequence of least-squares problems whose conditioning and solutions depend on the previous time steps. As a result, any bias introduced by the regularization can accumulate over time, and judicious techniques and a systematic and rigorous analysis are necessary to control and interpret the effect of the bias across the whole time trajectory (Feischl *et al.* 2024). Furthermore, regularization must be used with caution because even commonly used regularizers can lead to a selection of updates (e.g. minimal-norm updates) that preserve parameter redundancies (e.g. identical or redundant components remain identical) and so prevent recovery from a degeneracy (Zhang *et al.* 2024).

7.5.2. Regularized Dirac–Frenkel variational principle

A systematic and rigorous study of Tikhonov regularization of Dirac–Frenkel least-squares problems has been conducted by Feischl *et al.* (2024). To mitigate ill-conditioning in the residual minimization problem, Feischl *et al.* (2024) apply Tikhonov regularization on the time derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ in the time-continuous formulation and on the update ('velocity') $\delta\boldsymbol{\theta}_k(\boldsymbol{\mu})$ in time-discrete formulation, for example,

$$\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}) = \operatorname*{arg\,min}_{\boldsymbol{\eta}\in\mathbb{R}^n} \|\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}))\boldsymbol{\eta} - \boldsymbol{F}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}); \boldsymbol{\mu})\|_2^2 + \alpha\|\boldsymbol{\eta}\|_2^2,$$

with the update $\boldsymbol{\theta}_k(\boldsymbol{\mu}) = \boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}) + \tau\delta\boldsymbol{\theta}_k(\boldsymbol{\mu})$. Here, the parameter $\alpha \geq 0$ controls the Tikhonov regularization, which acts as a smoothing filter in the sense that directions associated with small singular values of the system matrix of the least-squares problems are damped, thereby suppressing large and potentially unstable weight updates.

The analysis by Feischl *et al.* (2024) emphasizes that an accurate approximation of the solution itself is not sufficient for reliable time evolution. One additionally needs that the right-hand side (equivalently, the time derivative of the solution) is well approximated by the tangent space along the solution trajectory, because the error of approximating the right-hand side in the tangent space enters the overall approximation error; see also Lubich (2005) and Zhang *et al.* (2024). This observation is not specific to nonlinear model reduction. An analogous phenomenon already arises in linear model reduction, where an accurate approximation of the state alone does not guarantee an accurate approximation of the time evolution. This insight underlies the widespread use of Petrov–Galerkin projections in linear model reduction, where distinct trial and test spaces are chosen to improve the approximation of the system dynamics (Otto *et al.* 2022).

Furthermore, the regularization strength has to be chosen carefully: less regularization reduces bias but can lead to updates aligned with directions corresponding to small singular values, while more regularization helps stability but can ultimately prevent the residual from being small. Feischl *et al.* (2024) introduce an adaptation approach based on the defect, which is the minimal value of the regularized least-squares objective. Feischl *et al.* (2024) also relate Tikhonov regularization to truncated SVD regularization, in the sense that truncated SVD entirely discards directions corresponding to small singular values rather than only smoothing them as Tikhonov regularization does.

Lubich and Nick (2025) target stiff problems and therefore consider regularization for implicit instead of explicit time integration schemes with Dirac–Frenkel. At each time step a regularized nonlinear least-squares problem is solved, with a regularizer on the weight update. The nonlinear problems are solved with a few Gauss–Newton iterations.

7.5.3. Regularization via random subspace approximations

Berman and Peherstorfer (2023) propose regularizing the dynamics corresponding to the Dirac–Frenkel variational principle by solving for the weight update in a random subspace, which can be interpreted as sketching from the right; this is further developed in Dong, Schwerdtner and Peherstorfer (2025).

Building on the time-discrete problem (7.17)–(7.18), the weight update becomes

$$\boldsymbol{\theta}_k(\boldsymbol{\mu}) = \boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}) + \frac{\tau}{\ell} \sum_{i=1}^{\ell} \delta\boldsymbol{\theta}_k(\boldsymbol{\mu}; \boldsymbol{\Gamma}_{k,i}), \tag{7.20}$$

where $\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}; \boldsymbol{\Gamma}_{k,1}), \ldots, \delta\boldsymbol{\theta}_k(\boldsymbol{\mu}; \boldsymbol{\Gamma}_{k,\ell})$ are ℓ independent random updates depending on the independent random embeddings $\boldsymbol{\Gamma}_{k,1}, \ldots, \boldsymbol{\Gamma}_{k,\ell}$. Notice that the average over these ℓ updates is used in (7.20), which helps to control the variance of the random update. The updates are obtained from the least-squares problem

$$\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}; \boldsymbol{\Gamma}) = \operatorname*{arg\,min}_{\boldsymbol{\eta} \in \mathrm{Range}(\boldsymbol{\Gamma})} \|\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}))\boldsymbol{\eta} - \boldsymbol{F}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}); \boldsymbol{\mu})\|_2^2, \tag{7.21}$$

which restricts the solution to be in the column span of the random matrix $\mathbf{\Gamma}$. The matrix $\mathbf{\Gamma}$ is of size $n \times s$ with $s \leq n$.

Typical choices for $\mathbf{\Gamma}$ are Gaussian embeddings and random unitary embeddings. For such embeddings, Dong *et al.* (2025) have developed bounds on the condition number of the sketched batch gradient $\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}))\mathbf{\Gamma}$ with respect to the spectrum of $\boldsymbol{J}(\boldsymbol{\theta}_{k-1}(\boldsymbol{\mu}))$ and the sketching dimension s, which directly controls the condition number of the least-squares problem (7.21). Earlier, Berman and Peherstorfer (2023) considered sparse updates $\delta\boldsymbol{\theta}_k(\boldsymbol{\mu}; \mathbf{\Gamma})$ by choosing $\mathbf{\Gamma}$ to correspond to a randomized sparse coordinate embedding.

The randomized formulation (7.21) sketches the least-squares problem from the right, that is, it restricts the weight update to a random subspace. Consequently, the unknown in (7.21) is an s-dimensional vector and thus one has to solve for s unknowns only, rather than for all n unknowns corresponding to all components of the weight update. In contrast, there is a line of work on sketching from the left (Lam *et al.* 2025, Carrel 2026) for dynamic low-rank approximations as well as other more generic nonlinear parametrizations (Lindsey 2025). Sketching from the left can help to improve the conditioning, but the number of unknowns of the update remains n instead of s.

7.6. Instantaneous residual minimization with discretize-then-optimize (DtO) versus optimize-then-discretize (OtD) approaches

The Dirac–Frenkel variational principle can be interpreted as being of the optimize-then-discretize (OtD) type: one first derives a continuous-time evolution equation for the weights $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ by enforcing orthogonality of the residual against the tangent space of the trial manifold, and only afterwards discretizes this evolution in time.

As discussed in detail in Zhang *et al.* (2024), a conceptually different class of instantaneous (or sequential-in-time) residual minimization schemes arises from a discretize-then-optimize (DtO) strategy. In DtO schemes, time is discretized at the PDE level first, which leads to a sequence of boundary value problems in the spatial variable. Only after the time discretization is the nonlinear parametrization introduced and a residual minimization problem formulated for each boundary value problem corresponding to the discrete time steps. Examples of DtO residual minimization schemes for nonlinear parametrizations include the works by Kvaal, Lasser, Pedersen and Adamowicz (2023) and Chen *et al.* (2023*a*, 2024). We note that for linear parametrizations, DtO corresponds to the Rothe method (Rothe 1930, Deuflhard and Weiser 2012). Because residual minimization for each boundary value problem leads to a typically non-convex optimization problem at each time step, the computational costs of DtO schemes tend to be higher than the costs of OtD schemes that lead to a linear least-squares problem at each time step, at least for explicit time integration. At the same time, DtO schemes can avoid ill-conditioning issues that typically arise in OtD schemes, such as the tangent space collapse or matrix singularity issue. We refer to Zhang *et al.* (2024) for an in-depth discussion about OtD versus DtO schemes.

We remark that there is an analogue of OtD versus DtO formulations in linear model reduction. Carlberg *et al.* (2011, 2017) introduce and analyse the concept of least-squares Petrov–Galerkin reduced models based on linear parametrizations. In the terminology used in Carlberg *et al.* (2017), DtO corresponds to least-squares Petrov–Galerkin and OtD to Galerkin formulations.

7.7. Instantaneous residual minimization with trained nonlinear parametrizations for nonlinear model reduction

We now survey model reduction methods that apply instantaneous residual minimization in the online phase to parametrizations trained offline on snapshot data. The capability to train a parametrization typically requires it to depend on offline weights, denoted in vector form as $\boldsymbol{\theta}_{\text{off}}$ (see Section 4.3.1). Once the parametrization is trained, i.e. the offline weights are fixed, the online weights $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ can be obtained via instantaneous residual minimization in the online phase.

This section focuses on trained nonlinear parametrizations obtained with neural-network autoencoders, autoencoders with *a priori* fixed feature maps, and neural representations. We emphasize that there are also transformation-based methods and online adaptive model reduction that yield trained nonlinear parametrizations that can be used online with residual minimization; we cover these in the respective sections.

7.7.1. Nonlinear parametrizations given by neural-network autoencoders

The pioneering work by Lee and Carlberg (2020) proposes using autoencoders parametrized by deep convolutional neural networks, train them on snapshot data in the offline phase, and then use the decoder function of the autoencoder to induce a nonlinear parametrization for representing the reduced solution in the online phase. The vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ in this setting corresponds to a latent state of the reduced solution and is computed with instantaneous residual minimization. While there is earlier work on using neural-network autoencoders for model reduction (Kashima 2016, Hartman and Mestha 2017), the work by Lee and Carlberg (2020) was the first that presented a comprehensive framework that sparked a series of publications on using autoencoders for nonlinear model reduction. A general differential–geometric framework for using autoencoders in nonlinear model reduction is introduced in Buchfink *et al.* (2024*b*). Our presentation loosely follows Lee and Carlberg (2020).

Autoencoders. An autoencoder is the composition $\Phi^{\uparrow} \circ \Phi^{\downarrow} \colon \mathbb{R}^N \to \mathbb{R}^N$ of two maps, the encoder, $\Phi^{\downarrow} \colon \mathbb{R}^N \to \mathbb{R}^n$, and the decoder, $\Phi^{\uparrow} \colon \mathbb{R}^n \to \mathbb{R}^N$. The encoder $\Phi^{\downarrow}$ maps a high-dimensional vector of dimension N onto a low-dimensional vector of dimension $n \ll N$. The decoder operates in the opposite direction and lifts a low-dimensional vector to a high-dimensional one.

In the setting of model reduction, we are typically interested in encoder–decoder pairs so that the composition $\Phi^{\uparrow} \circ \Phi^{\downarrow}$ is close to the identity map on data corresponding to the solution manifold (2.5) induced by the full model. When applying

the autoencoder to a state $\boldsymbol{q}(t,\boldsymbol{\mu}) \in \mathbb{R}^N$ of the semi-discrete full model (2.4), then the encoder maps the full-model state $\boldsymbol{q}(t,\boldsymbol{\mu})$ onto a low-dimensional approximation $\tilde{\boldsymbol{q}}(t,\boldsymbol{\mu}) = \Phi^{\downarrow}(\boldsymbol{q}(t,\boldsymbol{\mu})) \in \mathbb{R}^n$. The decoder lifts the n-dimensional approximation $\tilde{\boldsymbol{q}}(t,\boldsymbol{\mu})$ onto an N-dimensional vector $\Phi^{\uparrow}(\tilde{\boldsymbol{q}}(t,\boldsymbol{\mu})) \in \mathbb{R}^N$ that approximates the full-model state $\boldsymbol{q}(t,\boldsymbol{\mu})$. In view of linear model reduction in the Galerkin setting (2.12), the encoder is given by $\Phi^{\downarrow}(\boldsymbol{q}) = \boldsymbol{V}^{\top}\boldsymbol{q}$ and the decoder by $\Phi^{\uparrow}(\tilde{\boldsymbol{q}}) = \boldsymbol{V}\tilde{\boldsymbol{q}}$. In the Petrov–Galerkin setting, one would replace the matrix $\boldsymbol{V}^{\top}$ in the encoder with another matrix $\boldsymbol{W}^{\top}$.

Training autoencoders on snapshot data. Lee and Carlberg (2020) parametrize the encoder and decoder maps with neural networks. In the spirit of Section 4.3, we denote the parametrized encoder as $\Phi^{\downarrow}\big(\cdot;\boldsymbol{\theta}^{\downarrow}_{\text{off}}\big)\colon \mathbb{R}^N \to \mathbb{R}^n$ and the parametrized decoder as $\Phi^{\uparrow}\big(\cdot;\boldsymbol{\theta}^{\uparrow}_{\text{off}}\big)\colon \mathbb{R}^n \to \mathbb{R}^N$ with offline weight vectors $\boldsymbol{\theta}^{\downarrow}_{\text{off}}$ and $\boldsymbol{\theta}^{\uparrow}_{\text{off}}$ of appropriate size. The architecture of the parametrization depends on the problem at hand. For example, Lee and Carlberg (2020) and Buchfink, Glas and Haasdonk (2023) use deep convolutional neural networks. In contrast, Kim, Choi, Widemann and Zohdi (2022) use a shallow network with a sparse mask on the output layer for reducing costs during the online phase. A comparison of different architectures is given by Gruber, Gunzburger, Ju and Wang (2022).

The offline weights $\boldsymbol{\theta}^{\downarrow}_{\text{off}}$ and $\boldsymbol{\theta}^{\uparrow}_{\text{off}}$ are trained on snapshot data (2.9) from the full model (2.4). A standard loss function, which is used in Lee and Carlberg (2020), is the mean-squared error of the reconstruction of the snapshots,

$$\min_{\boldsymbol{\theta}^{\downarrow}_{\text{off}},\boldsymbol{\theta}^{\uparrow}_{\text{off}}} \left\| \boldsymbol{Q}_{\text{train}} - \Phi^{\uparrow}\big(\Phi^{\downarrow}\big(\boldsymbol{Q}_{\text{train}};\boldsymbol{\theta}^{\downarrow}_{\text{off}}\big);\boldsymbol{\theta}^{\uparrow}_{\text{off}}\big)\right\|_F^2, \tag{7.22}$$

where $\boldsymbol{Q}_{\text{train}}$ is the snapshot matrix and the encoder and decoder are applied column-wise to their matrix inputs.

Additional terms can be appended to (7.22) to penalize structure violation. For example, Buchfink *et al.* (2023) add a penalty term that penalizes violations of symplecticity of the decoder, yielding a weakly symplectic autoencoder. A physics-informed penalization term is added in Lee and Carlberg (2021). Otto *et al.* (2023) additionally train a projection operator that is tailored to the learned autoencoder.

Online residual minimization with autoencoder-induced nonlinear parametrizations. Lee and Carlberg (2020) propose using the decoder $\Phi^{\uparrow}$ to induce a nonlinear parametrization. For ease of exposition, we now drop the dependence on the offline weights in the notation of the decoder. The induced parametrization $\Phi^{\uparrow}\colon \mathbb{R}^n \to \mathbb{R}^N$ is vector-valued and does not take the spatial coordinate as an input. Instead, we can interpret $\Phi^{\uparrow}$ as

$$\Phi^{\uparrow}(\boldsymbol{\theta}(t,\boldsymbol{\mu})) = \begin{bmatrix} \hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\boldsymbol{x}_1) & \cdots & \hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\boldsymbol{x}_N) \end{bmatrix}^{\top}, \tag{7.23}$$

where $\hat{q}(\boldsymbol{\theta}(t,\boldsymbol{\mu}),\cdot)\colon \Omega \to \mathbb{R}$ is an implicitly given scalar-valued nonlinear parametrization of the form (4.1) that is evaluated at the points $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N \in \Omega$ corresponding

to the spatial coordinates over which the semi-discrete full model (2.4) is defined. We stress that $\hat{q}$, which allows spatial coordinates other than $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ as inputs, is not directly available from the decoder function $\Phi^{\uparrow}$ in general. In particular, when autoencoders are typically used in model reduction, the autoencoders operate directly on the N-dimensional state vectors of the semi-discrete full model and are therefore inherently tied to the underlying full-model discretization. We refer to Section 7.7.3 for neural representations that are independent of the underlying full-model discretization.

Corresponding to the vector-valued parametrization induced by the decoder $\Phi^{\uparrow}$, one can define the vector-valued residual function $\boldsymbol{r} : \mathbb{R}^n \times \mathbb{R}^n \times \mathcal{D} \to \mathbb{R}^N$,

$$\boldsymbol{r}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}); \boldsymbol{\mu}) = \boldsymbol{J}(\boldsymbol{\theta}(t, \boldsymbol{\mu}))\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}) - \boldsymbol{f}(\Phi^{\uparrow}(\boldsymbol{\theta}(t, \boldsymbol{\mu})); \boldsymbol{\mu}),$$

where $\boldsymbol{f}$ is the right-hand side function of the semi-discrete full model (2.4) and $\boldsymbol{J}(\boldsymbol{\theta}(t, \boldsymbol{\mu}))$ is the Jacobian matrix of $\Phi^{\uparrow}$ evaluated at $\boldsymbol{\theta}(t, \boldsymbol{\mu})$. Recall that the residual function defined in (7.1) for scalar-valued nonlinear parametrizations depends on the gradient of $\hat{q}$ with respect to $\boldsymbol{\theta}(t, \boldsymbol{\mu})$, which becomes the batch gradient (7.14) corresponding to sampling points in the semi-discrete case. Here, because the parametrization is vector-valued over the N coordinates $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, $\boldsymbol{J}(\boldsymbol{\theta}(t, \boldsymbol{\mu}))$ is the Jacobian matrix of $\Phi^{\uparrow}$.

The time derivative $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ is then obtained by minimizing the norm of the vector-valued residual function,

$$\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu}) \in \operatorname*{arg\,min}_{\boldsymbol{\eta} \in \mathbb{R}^n} \|\boldsymbol{r}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \boldsymbol{\eta}; \boldsymbol{\mu})\|_2^2. \tag{7.24}$$

Lee and Carlberg (2020) focus on implicit time discretizations, which lead to nonlinear least-squares problems that have to be solved numerically at each time step. Constraints can be added to the residual minimization problem (7.24) (or its discrete counterpart) to preserve quantities such as mass, momentum and energy (Lee and Carlberg 2021).

The instantaneous residual minimization approach described by Lee and Carlberg (2020) is closely related to the Dirac–Frenkel variational principle, except that Lee and Carlberg (2020) directly work in the N coordinates corresponding to the semi-discrete full model.

Online costs and empirical interpolation for autoencoder-based nonlinear parametrizations. Achieving runtime speedups in the online phase compared to the full model is challenging with autoencoder-based nonlinear parametrization as used in Lee and Carlberg (2020). Although the optimization variable representing $\dot{\boldsymbol{\theta}}(t, \boldsymbol{\mu})$ is low-dimensional with dimension $n \ll N$, the residual itself is defined over $\mathbb{R}^N$. Consequently, the minimization problem (7.24) is carried out in the full-model state space $\mathbb{R}^N$. This issue becomes particularly challenging when an implicit time discretization is employed, leading to a nonlinear optimization problem that must be solved at each time step. For instance, if the corresponding nonlinear optimization

problem is solved with Gauss–Newton (or similar) methods, then at each optimization iteration, the weight $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ is lifted to a full-model state approximation of dimension N with the decoder $\Phi^{\uparrow}$ and then the residual needs to be computed with the full-model right-hand side function $\boldsymbol{f}$ at all N components. Additionally, the Jacobian matrix is computed corresponding to all N spatial coordinates over which the full model is defined.

The issue of lifting back to the high-dimensional full-model state space to compute the residual is analogous to the lifting bottleneck in linear model reduction, when the full-model dynamics are nonlinear in the state; see Section 2.4. Correspondingly, there is work that has developed autoencoder parametrizations that are compatible with empirical interpolation-like techniques. Kim *et al.* (2022) propose using a shallow, masked decoder function that can be combined with sparse sampling methods analogous to empirical interpolation. Romor, Stabile and Rozza (2023) go a step further and propose two complementary mechanisms to reduce online costs. First, the time-discrete residual minimization is performed only on a small set of selected collocation points, which builds on the over-collocation approach introduced in Chen, Gottlieb, Ji and Maday (2021) to sparsely evaluate the decoder function. Second, a compressed decoder is used that evaluates to values at the set of collocation points and so avoids evaluations at all N spatial coordinates corresponding to the full-model state in the online phase.

Remark 7.4. There are methods that train neural-network autoencoders to represent solution fields and additionally learn the reduced dynamics from data, rather than computing the online weights via instantaneous residual minimization (Xu and Duraisamy 2020, Maulik, Lusch and Balaprakash 2021, Fresca and Manzoni 2022). We briefly review such non-intrusive model reduction methods in Section 8.

7.7.2. *Autoencoders with* a priori *feature maps and quadratic approximations*

We now survey nonlinear model reduction techniques based on autoencoders with decoders that are formulated via *a priori* chosen, fixed nonlinear feature maps and encoder functions that are linear. In particular, we focus on polynomial feature maps, and especially quadratic feature maps, which lead to so-called quadratic (trial) manifolds.

Autoencoders with a priori *chosen, fixed feature maps.* From Section 7.7.1, recall that an autoencoder consists of an encoder $\Phi^{\downarrow}\left(\cdot; \boldsymbol{\theta}^{\downarrow}_{\text{off}}\right) \colon \mathbb{R}^N \to \mathbb{R}^n$ and a decoder $\Phi^{\uparrow}\left(\cdot; \boldsymbol{\theta}^{\uparrow}_{\text{off}}\right) \colon \mathbb{R}^n \to \mathbb{R}^N$, which depend on trainable offline weights $\boldsymbol{\theta}^{\downarrow}_{\text{off}}$ and $\boldsymbol{\theta}^{\uparrow}_{\text{off}}$, respectively. Recall further that we can interpret the decoder $\Phi^{\uparrow}(\cdot; \boldsymbol{\theta}_{\text{off}}) \colon \mathbb{R}^n \to \mathbb{R}^N$ as inducing a vector-valued nonlinear parametrization as in (7.23).

We now consider decoders $\Phi^{\uparrow}$, and thus equivalently vector-valued nonlinear parametrizations, that have the form

$$\Phi^{\uparrow}\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{\uparrow}_{\text{off}}\right) = \boldsymbol{V}_1 \boldsymbol{\theta} + \boldsymbol{V}_2 g(\boldsymbol{\theta}), \tag{7.25}$$

where $\boldsymbol{V}_1 \in \mathbb{R}^{N\times n}$ and $\boldsymbol{V}_2 \in \mathbb{R}^{N\times S}$ are matrices that are trainable offline, i.e. $\boldsymbol{\theta}^{\uparrow}_{\text{off}} = [\text{vec}(\boldsymbol{V}_1); \text{vec}(\boldsymbol{V}_2)]$. The function $g\colon \mathbb{R}^n \to \mathbb{R}^S$ is a nonlinear feature map function that is given *a priori*. The rationale behind ansatz (7.25) is twofold: (i) to enrich the linear subspace spanned by the columns of $\boldsymbol{V}_1$ by nonlinear terms (as in a truncated Taylor expansion), and (ii) for certain applications, Cohen *et al.* (2024) prove that coefficients of higher-order POD modes can be expressed as functions of lower-order ones.

A feature map g, which is common in the context of nonlinear model reduction, is the quadratic function that collects all unique degree-2 monomials

$$g(\boldsymbol{\theta}) = \boldsymbol{\theta}\hat{\otimes}\boldsymbol{\theta}, \tag{7.26}$$

where $\hat{\otimes}$ denotes the symmetric (duplicate-free) Kronecker product, i.e. the vector containing the products $\{\theta_i\theta_j\}_{1\le i\le j\le n}$ of all n components of $\boldsymbol{\theta}$ in a fixed ordering. For such a quadratic feature map, the number of features is $S = n(n+1)/2$. We can interpret the two terms of (7.25) as follows: the first term $\boldsymbol{V}_1\boldsymbol{\theta}$ provides a linear approximation and the second term $\boldsymbol{V}_2 g(\boldsymbol{\theta})$ is a nonlinear correction. A nonlinear parametrization of the form (7.25) gives rise to a trial manifold, which is sometimes called quadratic manifold if the feature map is quadratic.

Quadratic and polynomial approximations are pursued by Jain, Tiso, Rutzmoser and Rixen (2017), Rutzmoser, Rixen, Tiso and Jain (2017), Barnett and Farhat (2022), Cenedese *et al.* (2022) and Geelen, Wright and Willcox (2023*b*), and have led to a series of publications on quadratic approximations for nonlinear model reduction, including methods for training quadratic approximations on snapshot data (Geelen, Balzano and Willcox 2023*a*, Geelen, Balzano, Wright and Willcox 2024, Schwerdtner and Peherstorfer 2024, Schwerdtner *et al.* 2025*b*) and efficiently using them in the online phase (Benner, Goyal, Heiland and Pontes Duff 2023, Sharma *et al.* 2023, Goyal and Benner 2024, Yıldız, Goyal, Bendokat and Benner 2024, Glas and Mu 2025, Weder *et al.* 2025). Earlier, Gu (2012) also discussed nonlinear parametrizations based on polynomial feature maps in the context of model reduction. In general, we remark that quadratic systems and quadratic dynamics as well as leveraging the corresponding tensor structure are omnipresent in model reduction; see e.g. Gu (2011), Benner and Breiten (2015), Schlegel and Noack (2015), Benner, Goyal and Gugercin (2018), Kramer and Willcox (2019) and Qian, Kramer, Peherstorfer and Willcox (2020). An extension to rational manifolds is presented by Klein *et al.* (2025).

The expressivity of nonlinear parametrizations of the form (7.25) is dominated by the *a priori* chosen feature map g. Buchfink, Glas and Haasdonk (2024*a*) provide lower bounds on the Kolmogorov n-width for such nonlinear parametrizations with polynomial feature maps g. In particular, roughly speaking, Buchfink *et al.* (2024*a*) show that for the linear advection example studied in Section 3.3, the lower bound of the n-width behaves as $O(n^{-\alpha/2})$, where α is the degree of the polynomial.

Relying on approximation-theoretic results regarding stable manifold widths and sensing numbers (see Section 4.2) and popularized by Barnett and Farhat (2022),

Geelen *et al.* (2023*b*) and Cohen *et al.* (2024), it is a common choice to use a linear encoder $\Phi^{\downarrow}(\cdot;\boldsymbol{\theta}^{\downarrow}_{\text{off}}) : \mathbb{R}^N \to \mathbb{R}^n$, that is,

$$\Phi^{\downarrow}(\boldsymbol{q}) = \boldsymbol{W}^{\top}\boldsymbol{q}, \tag{7.27}$$

for some matrix $\boldsymbol{W} \in \mathbb{R}^{N\times n}$. While general conditions for the matrix $\boldsymbol{W}$ can be inferred from the general framework presented in Buchfink *et al.* (2024*b*), a common choice is to use $\boldsymbol{W}^{\top} = \boldsymbol{V}_1^{+}$, corresponding to the Moore–Penrose pseudo-inverse of the matrix $\boldsymbol{V}_1$. Note that if $\boldsymbol{V}_1$ has orthonormal columns, then $\boldsymbol{V}_1^{+} = \boldsymbol{V}_1^{\top}$. Linear encoders are also used with more generic nonlinear decoders than quadratic ones in Barnett, Farhat and Maday (2023*a*,*b*), Ballout, Maday and Prud'homme (2024) and Cohen *et al.* (2024). In particular, Cohen *et al.* (2024) investigate such linear encoder and nonlinear decoder pairs in terms of the sensing number and show that the approximation error for functions representing transported features can decay substantially faster than the Kolmogorov n-width with respect to the dimension n. A similar strategy via a linear encoder and a nonlinear decoder is taken in Bensalah, Nouy and Soffo (2025). Furthermore, it is also observed empirically that linear encoders achieve reconstruction errors similar to more generic nonlinear encoders with the additional benefit of an improved training robustness (Glas and Unger 2025, Schwerdtner, Gugercin and Peherstorfer 2025*a*).

Leveraging the structural form for efficient training on snapshot data. Recall the linear encoder (7.27) with the particular choice $\boldsymbol{W}^{\top} = \boldsymbol{V}_1^{+}$ and nonlinear decoder pair (7.25) with the trainable matrices $\boldsymbol{V}_1 \in \mathbb{R}^{N\times n}$ and $\boldsymbol{V}_2 \in \mathbb{R}^{N\times S}$, which constituted the offline weights $\boldsymbol{\theta}^{\downarrow}_{\text{off}} = \text{vec}(\boldsymbol{V}_1)$ and $\boldsymbol{\theta}^{\uparrow}_{\text{off}} = [\text{vec}(\boldsymbol{V}_1); \text{vec}(\boldsymbol{V}_2)]$. Recall further that the feature map g is given *a priori*.

A typical training strategy to train such nonlinear parametrizations is to first choose the matrix $\boldsymbol{V}_1$ and then to fit $\boldsymbol{V}_2$ with a linear least-squares problem,

$$\boldsymbol{V}_2 = \underset{\boldsymbol{Z}\in\mathbb{R}^{N\times S}}{\arg\min} \Big\| \boldsymbol{Q}_{\text{train}} - \boldsymbol{V}_1 \underbrace{\Phi^{\downarrow}\big(\boldsymbol{Q}_{\text{train}};\boldsymbol{\theta}^{\downarrow}_{\text{off}}\big)}_{\boldsymbol{V}_1^{+}\boldsymbol{Q}_{\text{train}}} - \boldsymbol{Z}\, g\Big(\underbrace{\Phi^{\downarrow}\big(\boldsymbol{Q}_{\text{train}};\boldsymbol{\theta}^{\downarrow}_{\text{off}}\big)}_{\boldsymbol{V}_1^{+}\boldsymbol{Q}_{\text{train}}}\Big) \Big\|_F^2 + \lambda \|\boldsymbol{Z}\|_F^2, \tag{7.28}$$

which corresponds to the reconstruction error

$$\boldsymbol{Q}_{\text{train}} - \Phi^{\uparrow}\big(\Phi^{\downarrow}\big(\boldsymbol{Q}_{\text{train}};\boldsymbol{\theta}^{\downarrow}_{\text{off}}\big);\boldsymbol{\theta}^{\uparrow}_{\text{off}}\big)$$

of the snapshots (2.9) given as columns in the snapshot matrix $\boldsymbol{Q}_{\text{train}}$. The encoder and feature map are applied column-wise to the snapshot matrix. A regularizer is added to the least-squares problem with regularization parameter $\lambda > 0$.

Let us now consider fitting the matrix $\boldsymbol{V}_1$, which fully determines the encoder but also enters in the linear term in the decoder. Barnett and Farhat (2022) and Geelen *et al.* (2023*b*) follow the interpretation that $\Phi^{\uparrow}\big(\boldsymbol{\theta};\boldsymbol{\theta}^{\uparrow}_{\text{off}}\big) = \boldsymbol{V}_1\boldsymbol{\theta} + \boldsymbol{V}_2 g(\boldsymbol{\theta})$ as defined in (7.25) is a linear approximation $\boldsymbol{V}_1\boldsymbol{\theta}$ with a nonlinear correction term $\boldsymbol{V}_2 g(\boldsymbol{\theta})$. Thus Barnett and Farhat (2022) and Geelen *et al.* (2023*b*) fit $\boldsymbol{V}_1$ so that the corresponding linear term $\boldsymbol{V}_1\boldsymbol{\theta}$ leads to the lowest linear approximation error on the snapshot data

by setting the columns of the matrix $\boldsymbol{V}_1 \in \mathbb{R}^{N\times n}$ to the first n leading left-singular vectors of the snapshot matrix $\boldsymbol{Q}_{\text{train}}$. However, Cohen *et al.* (2024) and Schwerdtner and Peherstorfer (2024) stress the perspective that the nonlinear correction term evaluates the feature map g at the encoded point $\boldsymbol{\theta} = \Phi^{\downarrow}\left(\boldsymbol{q}; \boldsymbol{\theta}^{\downarrow}_{\text{off}}\right) = \boldsymbol{V}_1^{+}\boldsymbol{q}$. Thus, purely fitting $\boldsymbol{V}_1$ to minimize the approximation error of the linear approximation can lead to an encoding $\Phi^{\downarrow}\left(\boldsymbol{q}; \boldsymbol{\theta}^{\downarrow}_{\text{off}}\right)$ that discards information that is crucial for the feature map g to be informative and to provide an effective correction. To address this, Schwerdtner and Peherstorfer (2024) introduce a greedy strategy to build $\boldsymbol{V}_1$ column by column. Rather than fixing $\boldsymbol{V}_1$ to have columns corresponding to the first n left-singular vectors (principal components) of the snapshot matrix $\boldsymbol{Q}_{\text{train}}$, the greedy method introduced in Schwerdtner and Peherstorfer (2024) selects n basis vectors from a larger pool of leading and later principal components that can have index greater than n. At each step of the greedy procedure, Schwerdtner and Peherstorfer (2024) choose a vector that yields the largest improvement in terms of the reconstruction error, after fitting $\boldsymbol{V}_2$ with the linear least-squares problem (7.28). The resulting $\boldsymbol{V}_1$ can therefore include non-leading left-singular vectors (i.e. with index greater than n) if they are important for making the feature map effective in terms of reducing the reconstruction-error norm in the objective (7.28).

In another line of work, Geelen *et al.* (2023*a*, 2024) formulate optimization problems that construct $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ together, which can also result in matrices that achieve lower approximation errors than defining the matrix $\boldsymbol{V}_1$ via the n leading singular vectors of the snapshot matrix.

Leveraging the structural form for efficient online phase. A key advantage of quadratic (or polynomial) approximations of the form (7.25) is that they are highly structured, which can be leveraged in the residual minimization in the online phase.

Recall the semi-discrete least-squares problem (7.16) corresponding to instantaneous residual minimization with the Dirac–Frenkel variational principle. Recall further that the batch gradient becomes a Jacobian matrix because $\Phi^{\uparrow} : \mathbb{R}^n \to \mathbb{R}^N$ is a vector-valued nonlinear parametrization. For parametrizations with quadratic feature maps (7.26), the Jacobian matrix $\boldsymbol{J}(\boldsymbol{\theta}(t,\boldsymbol{\mu})) \in \mathbb{R}^{N\times n}$ of $\Phi^{\uparrow}$ admits a closed-form expression and is affine in $\boldsymbol{\theta}(t,\boldsymbol{\mu})$. More precisely, because $\partial(\boldsymbol{\theta}\hat{\otimes}\boldsymbol{\theta})/\partial\boldsymbol{\theta}$ is linear in $\boldsymbol{\theta}$, one can write

$$\boldsymbol{J}(\boldsymbol{\theta}(t,\boldsymbol{\mu})) = \boldsymbol{V}_1 + \sum_{\ell=1}^{n} \theta_\ell(t,\boldsymbol{\mu})\, \boldsymbol{J}_\ell \tag{7.29}$$

for matrices $\boldsymbol{J}_\ell \in \mathbb{R}^{N\times n}$ that can be pre-computed in the offline phase independently of the online weight vector $\boldsymbol{\theta}(t,\boldsymbol{\mu})$.

When the full model is linear in the state and has an affine parameter dependence, then the objective of the least-squares problem for instantaneous residual minimization can be assembled in the online phase with no computations that scale with the ambient dimension N; this offline–online separation and N-independent online assembly is shown and leveraged in Sharma *et al.* (2023) and Weder *et al.* (2025).

When the full model has a nonlinear state dependence, the pre-computation of the Jacobian components can still be useful. In particular, Weder *et al.* (2025) propose a quadratic approximation formulation that is not tied to the N full-model spatial coordinates but can be realized on a set of selected collocation points, which avoids more expensive online costs.

Yet another way to leverage the quadratic structure is proposed by Barnett and Farhat (2022), who plug the quadratic parametrization into an implicit-in-time least-squares Petrov–Galerkin discretization. Because the quadratic parametrization has a closed-form affine Jacobian, sparse sampling (analogous to empirical interpolation; see Section 2.4) can be tailored to it and efficiently applied. As a result, online residual and gradient evaluations are restricted to a set of a few selected points, which leads to online runtime speedups compared to the full model in the presented numerical experiments.

Geelen *et al.* (2023*b*) even go a step further in leveraging structure, and approximate the Jacobian (7.29) with its zeroth-order approximation, which is just the constant function that evaluates to $\boldsymbol{V}_1$. In this case, the Jacobian matrix is independent of the online weight $\boldsymbol{\theta}(t, \boldsymbol{\mu})$, which leads to faster reduced-model evaluations in the online phase; see Weder *et al.* (2025) for a comparison in terms of online runtime. Furthermore, Geelen *et al.* (2023*b*) demonstrate that approximating the Jacobian matrix with its zeroth-order approximation $\boldsymbol{V}_1$ is a key step in deriving a non-intrusive reduced modelling approach based on quadratic manifolds; see Section 8 for a more detailed discussion about intrusive versus non-intrusive model reduction.

The structure offered by quadratic (and other *a priori* fixed feature-map approximations) in the Jacobian (7.29) is in stark contrast to the gradients corresponding to generic autoencoder parametrizations such as neural networks, that have no comparable low-order structure and so must be obtained repeatedly online via automatic differentiation, possibly within each Gauss–Newton iteration at each time step. However, we stress that *a priori* fixed feature maps often lead to nonlinear parametrizations with lower expressivity than more generic architectures given by deep neural networks with representation learning. Again, we refer to Buchfink *et al.* (2024*a*) for lower approximation bounds in this context.

7.7.3. Neural representations with offline–online weights

We now consider neural representations, which use a neural network to directly represent the nonlinear parametrization as $\hat{q}$.

Neural representations of solution functions in model reduction. Neural-network parametrizations of solution fields are sometimes referred to as implicit neural representations, which originated in the computer graphics community (Sitzmann *et al.* 2020) but have also been used for reducing (compressing) physics solution fields; see e.g. Pan, Brunton and Kutz (2023).

A key feature of neural representations is that they can be trained on snapshot data coming from different spatial discretizations because they represent fields

as functions of spatial coordinates, which is in contrast to typical neural-network autoencoders as discussed above that depend on a fixed spatial discretization. This is particularly advantageous when full-model solutions are computed on adaptive meshes or point clouds, which is leveraged in the context of nonlinear model reduction in Chen *et al.* (2023*b*).

To use neural representations in model reduction, an important design choice is how to decompose the weights of the neural network into an offline weight vector $\boldsymbol{\theta}_{\text{off}}$ that can be trained offline once for all snapshot data, and an online weight vector $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ with few components that are obtained online with, for example, instantaneous residual minimization. Based on an offline–online decomposition of the weights, one can represent the nonlinear parametrization as $\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \cdot; \boldsymbol{\theta}_{\text{off}}) \colon \Omega \to \mathbb{R}$, which is the form given in (4.7), where $\boldsymbol{\theta}(t, \boldsymbol{\mu})$ corresponds to the online weights of the network that may vary with time t and parameter $\boldsymbol{\mu}$. The input to the neural network is the spatial coordinate in Ω. Moreover, the offline weights $\boldsymbol{\theta}_{\text{off}}$ are independent of time t and the parameter $\boldsymbol{\mu}$ and fixed during the online phase.

Neural representations with offline–online weights. One method to achieve an offline–online decomposition of the neural-network weights is to let the online weights enter via modulations to the offline weights. A common weight modulation approach is via low-rank matrices introduced in Hu *et al.* (2022), which leads to low-rank adaptation layers. For model reduction, continuous low-rank adaptation (CoLoRA) layers have been introduced by Berman and Peherstorfer (2024), where the modulation can depend continuously on time and a parameter $\boldsymbol{\mu}$.

A CoLoRA layer has the form

$$\mathcal{C}_\ell(\boldsymbol{y}) = \boldsymbol{W}_\ell \boldsymbol{y} + \alpha_\ell(t, \boldsymbol{\mu}) \boldsymbol{A}_\ell \boldsymbol{B}_\ell \boldsymbol{y} + \boldsymbol{b}_\ell, \tag{7.30}$$

where $\boldsymbol{W}_\ell$ is a weight matrix and $\boldsymbol{b}_\ell$ is a bias term. Additionally, there is a low-rank matrix $\boldsymbol{A}_\ell \boldsymbol{B}_\ell$ of rank ρ with the rank-ρ factors $\boldsymbol{A}_\ell$ and $\boldsymbol{B}_\ell$ and the modulation $\alpha_\ell(t, \boldsymbol{\mu}) \in \mathbb{R}$. The layers (7.30) can be composed into CoLoRA networks as

$$\hat{q}(\boldsymbol{\theta}(t, \boldsymbol{\mu}), \boldsymbol{x}; \boldsymbol{\theta}_{\text{off}}) = \mathcal{C}_L(\sigma(\mathcal{C}_{L-1}(\ldots \sigma(\mathcal{C}_1(\boldsymbol{x}))))), \tag{7.31}$$

which is a nonlinear parametrization that has the form (4.7). The offline weight vector $\boldsymbol{\theta}_{\text{off}}$ consists of all weights in the network (7.31) that are independent of time t and parameter $\boldsymbol{\mu}$, including all weight matrices $\boldsymbol{W}_1, \ldots, \boldsymbol{W}_L$, low-rank matrices $\boldsymbol{A}_1, \ldots, \boldsymbol{A}_L, \boldsymbol{B}_1, \ldots, \boldsymbol{B}_L$, and bias terms $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_L$. The online weight vector consists of the modulations

$$\boldsymbol{\theta}(t, \boldsymbol{\mu}) = \left[\alpha_1(t, \boldsymbol{\mu}), \ldots, \alpha_L(t, \boldsymbol{\mu})\right]^\top \in \mathbb{R}^L.$$

In the particular architecture of CoLoRA networks given in (7.31), the number of layers L determines the number of online weights n as $n = L$. There are other variants of CoLoRA networks, for example replacing some of the L CoLoRA layers with regular linear layers that do not have a modulation, so that one can achieve $n < L$ online weights. Furthermore, one can introduce multiple modulations per

layer by introducing a diagonal matrix to obtain

$$\mathcal{C}_\ell(\boldsymbol{y}) = \boldsymbol{W}_\ell \boldsymbol{y} + \boldsymbol{A}_\ell \operatorname{diag}\big(\alpha_\ell^{(1)}(t,\boldsymbol{\mu}),\ldots,\alpha_\ell^{(\rho)}(t,\boldsymbol{\mu})\big)\boldsymbol{B}_\ell \boldsymbol{y} + \boldsymbol{b}_\ell,$$

where the low-rank factors $\boldsymbol{A}_\ell$ and $\boldsymbol{B}_\ell$ have rank ρ. Such variants are discussed in Berman and Peherstorfer (2024). In general, there are many ways to modulate the weights (Perez *et al.* 2018).

There are also other ways of formulating neural networks with offline and online weights, besides weight modulation. For example, Chen *et al.* (2023*b*) consider multilayer perceptrons to represent $\hat{q}$ and let a latent state enter as input to the network. The latent state is analogous to what we refer to as online weight vector $\boldsymbol{\theta}(t,\boldsymbol{\mu})$, except that it enters as input. The weights of the multilayer perceptron are the offline weights in our terminology.

Training offline weights of neural representations on snapshot data. A direct training of the offline weight vector $\boldsymbol{\theta}_{\text{off}}$ and the online weight vector $\boldsymbol{\theta}(t,\boldsymbol{\mu})$ over all times and parameters would allow the online weights to vary arbitrarily over time, which can lead to non-smooth dependence of the online weights on time. To encourage regularity of the online weights in time t, Berman and Peherstorfer (2024) use a so-called modulation (or hyper-) network during training: let $h_{\boldsymbol{\psi}} : [0,T]\times\mathcal{D}\to\mathbb{R}^n$ be a neural network that maps the time t and the parameter $\boldsymbol{\mu}$ onto an n-dimensional vector $\boldsymbol{\theta}(t,\boldsymbol{\mu})$. The trainable weight vector of the network $h_{\boldsymbol{\psi}}$ is $\boldsymbol{\psi}\in\mathbb{R}^{n'}$, which is typically of low dimension.

Using the network $h_{\boldsymbol{\psi}}$ leads to the training problem

$$\min_{\boldsymbol{\theta}_{\text{off}},\boldsymbol{\psi}} \frac{1}{Mm(K+1)} \sum_{i=1}^{M}\sum_{j=1}^{m}\sum_{k=0}^{K} \frac{|q(t_k,\boldsymbol{x}_j;\boldsymbol{\mu}_i) - \hat{q}(h_{\boldsymbol{\psi}}(t_k,\boldsymbol{\mu}_i);\boldsymbol{x}_j;\boldsymbol{\theta}_{\text{off}})|^2}{|q(t_k,\boldsymbol{x}_j;\boldsymbol{\mu}_i)|^2} \tag{7.32}$$

over the offline weights $\boldsymbol{\theta}_{\text{off}}$ and the weights $\boldsymbol{\psi}$ of the network $h_{\boldsymbol{\psi}}$. Problem (7.32) depends on M training parameters $\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_M$ and $K+1$ time steps $0 = t_0 < t_1 < \cdots < t_K = T$, as in the training data (2.9). Additionally, we use m spatial points $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_m\in\Omega$. Recall that $q(t_k,\boldsymbol{x}_j;\boldsymbol{\mu}_i)$ is the full-model solution (2.2) at time t_k and parameter $\boldsymbol{\mu}_i$ evaluated at the spatial coordinate $\boldsymbol{x}_j$. In particular, when training neural representations, one can directly use the full-model solution function q instead of the semi-discrete solution vector, which emphasizes the independence of the neural representation from the mesh and discretization used for the full model.

The modulation network $h_{\boldsymbol{\psi}}$ can be seen as playing an analogous role to the encoder function in autoencoder-based parametrizations. The major difference however is that the modulation network receives as inputs the time t and the parameter $\boldsymbol{\mu}$ (and possibly other inputs) but not the full-model state vector.

Online residual minimization with trained neural representations. Once trained, the neural representation can be combined with instantaneous residual minimization in the online phase.

In stark contrast to previously considered nonlinear parametrizations that are formulated with respect to semi-discrete full models, which evaluate to vectors of dimension N (see Sections 7.7.1 and 7.7.2), neural representations are coordinate-to-value maps and exactly of the form of the nonlinear parametrizations in (4.1) (and with offline weights in (4.7)). Consequently, when applying schemes based on the Dirac–Frenkel variational principle for residual minimization, the inner products and residual norms can be numerically estimated with a set of m sampling points, where m can be chosen independently of the full-model dimension N and its mesh. In particular, Berman and Peherstorfer (2024) leverage Neural Galerkin schemes in the online phase. Additionally, quantities such as mass, momentum and energy can be conserved by adding constraints and nonlinear embedding steps to the instantaneous residual minimization (Schwerdtner *et al.* 2024).

At each time step in the online phase, a single linear least-squares problem has to be solved if an explicit time integration scheme is used (Berman and Peherstorfer 2024). The number of unknowns is n, with the aim of keeping n small when training the parametrization on snapshot data. The online costs are therefore primarily governed by the number m of sampling points used to approximate the residual norm in the least-squares problem and by computing the batch gradient and evaluating the full-model right-hand side function at the sampling points. In particular, the full-model right-hand side function can contain spatial derivatives, which can be obtained via automatic differentiation through the neural representation at the sampling points.

Because the online phase operates on the m sampling points rather than on the N points corresponding to the N-dimensional state vector of the semi-discrete full model, it avoids the classical lifting bottleneck that is typical for reduced models formulated over the semi-discrete full model. At the same time, accuracy depends on selecting sufficiently many and sufficiently informative sampling points. In particular, while the number m of sampling points and the number N of full-model grid points are formally independent, they are intrinsically related in the sense that if the full model needs a fine grid to resolve fine-scale solution features, then the number of sampling points m often has to reflect this, even though adaptivity can be helpful in keeping m small, which motivates adaptive sampling strategies, as discussed in the context of Neural Galerkin schemes in Section 7.3.3.

Remark 7.5. An alternative to online residual minimization is to reuse the trained modulation network h_ψ in the online phase to compute the online weights as $\boldsymbol{\theta}(t, \boldsymbol{\mu}) = h_\psi(t, \boldsymbol{\mu})$, thereby obtaining the solution approximation by only evaluating a neural network. Since the modulation network h_ψ depends only on known inputs such as t and $\boldsymbol{\mu}$ (rather than on the full state as an encoder function of an autoencoder), the resulting online phase is fully non-intrusive and data-driven, but it loses the Galerkin optimality associated with Dirac–Frenkel residual minimization. Related approaches for computing weights online via modulation networks or hyper-networks for non-intrusive model reduction are considered in, for example,

de Avila Belbute-Peres, Chen and Sha (2021), Cho, Lee, Rim and Park (2023) and Berman and Peherstorfer (2024); see Section 8 for further discussion.

8. Conclusions and outlook

Significant progress has been made on model reduction for transport-dominated problems, as reflected by the large and rapidly growing body of literature. A large class of nonlinear model reduction methods can be viewed through the unifying lens of generic nonlinear parametrizations combined with instantaneous residual minimization. At the same time, we have deliberately distinguished transformation-based and online adaptive methods from this generic perspective, as this classification highlights additional structural features that can be leveraged for improved efficiency and robustness.

Despite the breadth of the work cited in this survey, several important research directions remain under-represented, including structure-preserving model reduction, the reduction of stochastic systems, and model reduction for control applications. An entirely different line of research, still in its early stages, recasts hyperbolic PDEs as delay equations and then applies model reduction to this alternative representation. We refer to Schulze, Unger, Beattie and Gugercin (2018) for some first developments in this direction.

A particularly fruitful and rapidly evolving area is non-intrusive model reduction for transport-dominated problems. In contrast to the intrusive methods emphasized in this survey, non-intrusive model reduction approaches aim to learn both an efficient low-dimensional representation and the corresponding reduced dynamics primarily from snapshot data, rather than directly deriving the reduced dynamics from the governing equations. We emphasize that non-intrusive model reduction is distinct from equation discovery, which uses data to infer the governing physical laws themselves. A large literature on non-intrusive model reduction exists; see Ghattas and Willcox (2021) and Kramer *et al.* (2024) for an overview.

Several non-intrusive methods explicitly target transport-dominated regimes where linear model reduction is inefficient. One line of work incorporates nonlinear transformations that factor out transport before applying linear non-intrusive model reduction methods (Sesterhenn and Shahirpour 2019, Lu and Tartakovsky 2020, Sarna and Benner 2021, Issan and Kramer 2023, Gowrachari, Demo, Stabile and Rozza 2025), analogous to the transformation-based intrusive methods that we reviewed above. Another class of non-intrusive methods learns a nonlinear low-dimensional representation (e.g. via autoencoders) and then learns the reduced dynamics in the latent representation using a separate polynomial or neural-network model (Fresca, Dede' and Manzoni 2021, Maulik *et al.* 2021, Cenedese *et al.* 2022, Fresca and Manzoni 2022, Franco, Manzoni and Zunino 2023, Geelen *et al.* 2023*b*, Regazzoni *et al.* 2024, Tomasetto *et al.* 2025). We stress that the references cited in this outlook provide only a selective glimpse into the rapidly expanding field of non-intrusive model reduction. They are intended to illustrate the opportunities

of non-intrusive methods for nonlinear model reduction and not to provide a comprehensive review.

Acknowledgements
BP was partially supported by the National Science Foundation grant 2046521 and the Air Force Office of Scientific Research (AFOSR), USA, award FA9550-24-1-0327. BU is funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – Project-ID 258734477 – SFB 1173, and by the BMBF (grant no. 05M22VSA).

References

R. Abgrall, D. Amsallem and R. Crisovan (2016), Robust model reduction by l^1-norm minimization and approximation via dictionaries: Application to nonlinear hyperbolic problems, *Adv. Model. Simul. Engrg Sci.* **3**, art. 1.

V. M. Adamyan, D. Z. Arov and M. G. Krein (1971), Analytic properties of Schmidt pairs for a Hankel operator and the generalized Schur–Takagi problem, *Math. USSR Sbornik* **15**, 31–73.

J. Aghili, J. Z. Atokple, M. Billaud-Friess, G. Garnier, O. Mula and N. Tognon (2025), A dynamical neural Galerkin scheme for filtering problems, *ESAIM Proc. Surveys* **81**, 2–15.

A. Aitzhan, A. G. Nouri, P. Givi and H. Babaee (2025), Reduced order modeling of turbulent reacting flows on low-rank matrix manifolds, *J. Comput. Phys.* **521**, art. 113549.

M. Alireza Mirhoseini and M. J. Zahr (2023), Model reduction of convection-dominated partial differential equations via optimization-based implicit feature tracking, *J. Comput. Phys.* **473**, art. 111739.

D. Amsallem and C. Farhat (2008), Interpolation method for adapting reduced-order models and application to aeroelasticity, *AIAA J.* **46**, 1803–1813.

D. Amsallem and B. Haasdonk (2016), PEBL-ROM: Projection-error based local reduced-order models, *Adv. Model. Simul. Engrg Sci.* **3**, 1–25.

D. Amsallem, M. J. Zahr and C. Farhat (2012), Nonlinear model order reduction based on local reduced-order bases, *Internat. J. Numer. Methods Engrg* **92**, 891–916.

D. Amsallem, M. J. Zahr and K. Washabaugh (2015), Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction, *Adv. Comput. Math.* **41**, 1187–1230.

W. Anderson and M. Farazmand (2022), Evolution of nonlinear reduced-order solutions for PDEs with conserved quantities, *SIAM J. Sci. Comput.* **44**, A176–A197.

W. Anderson and M. Farazmand (2024), Fast and scalable computation of shape-morphing nonlinear solutions with application to evolutional neural networks, *J. Comput. Phys.* **498**, art. 112649.

A. C. Antoulas (2005), *Approximation of Large-Scale Dynamical Systems*, Advances in Design and Control, SIAM.

A. C. Antoulas, C. Beattie and S. Gugercin (2020), *Interpolatory Methods for Model Reduction*, Computational Science & Engineering, SIAM.

A. C. Antoulas, D. C. Sorensen and S. Gugercin (2001), A survey of model reduction methods for large-scale systems, *Contemp. Math.* **280**, 193–220.

F. Arbes, C. Greif and K. Urban (2025), The Kolmogorov N-width for linear transport: Exact representation and the influence of the data, *Adv. Comput. Math.* **51**, art. 13.

J.-P. Argaud, B. Bouriquet, F. de Caso, H. Gong, Y. Maday and O. Mula (2018), Sensor placement in nuclear reactors based on the generalized empirical interpolation method, *J. Comput. Phys.* **363**, 354–370.

E. Arian, M. Fahl and E. W. Sachs (2000), Trust-region proper orthogonal decomposition for flow control. Technical report 2000-25, ICASE.

A. Arnold and T. Jahnke (2014), On the approximation of high-dimensional differential equations in the hierarchical Tucker format, *BIT Numer. Math.* **54**, 305–341.

P. Astrid, S. Weiland, K. Willcox and T. Backx (2004), Missing point estimation in models described by proper orthogonal decomposition, in *43rd IEEE Conference on Decision and Control (CDC)*, Vol. 2, pp. 1767–1772. Available at doi:10.1109/CDC.2004.1430301.

P. Astrid, S. Weiland, K. Willcox and T. Backx (2008), Missing point estimation in models described by proper orthogonal decomposition, *IEEE Trans. Automat. Control* **53**, 2237–2251.

H. Babaee and T. P. Sapsis (2016), A minimization principle for the description of modes associated with finite-time instabilities, *Proc. R. Soc. A* **472**(2186), art. 20150779.

I. Babuvška and W. C. Rheinboldt (1978), Error estimates for adaptive finite element computations, *SIAM J. Numer. Anal.* **15**, 736–754.

M. Bachmayr and A. Cohen (2017), Kolmogorov widths and low-rank approximations of parametric elliptic PDEs, *Math. Comp.* **86**, 701–724.

M. Bachmayr, W. Dahmen and M. Oster (2025*a*), Variationally correct neural residual regression for parametric PDEs: On the viability of controlled accuracy, *IMA J. Numer. Anal*. Available at doi:10.1093/imanum/draf073.

M. Bachmayr, H. Eisenmann and A. Uschmajew (2025*b*), Dynamical low-rank tensor approximations to high-dimensional parabolic problems: Existence and convergence of spatial discretizations, *Numer. Math.* **157**, 781–822.

O. Balabanov and A. Nouy (2021), Randomized linear algebra for model reduction II: Minimal residual methods and dictionary-based approximation, *Adv. Comput. Math.* **47**, art. 26.

M. Balajewicz, I. Tezaur and E. Dowell (2016), Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier–Stokes equations, *J. Comput. Phys.* **321**, 224–241.

H. Ballout, Y. Maday and C. Prud'homme (2024), Nonlinear compressive reduced basis approximation for multi-parameter elliptic problem, in *Multiscale, Nonlinear and Adaptive Approximation II* (R. DeVore and A. Kunoth, eds), Springer, pp. 55–73.

L. Balzano, R. Nowak and B. Recht (2010), Online identification and tracking of subspaces from highly incomplete information, in *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, IEEE, pp. 704–711.

J. Barnett and C. Farhat (2022), Quadratic approximation manifold for mitigating the Kolmogorov barrier in nonlinear projection-based model order reduction, *J. Comput. Phys.* **464**, art. 111348.

J. Barnett, C. Farhat and Y. Maday (2023*a*), Neural-network-augmented projection-based model order reduction for mitigating the Kolmogorov barrier to reducibility, *J. Comput. Phys.* **492**, art. 112420.

J. L. Barnett, C. Farhat and Y. Maday (2023*b*), Mitigating the Kolmogorov barrier for the reduction of aerodynamic models using neural-network-augmented reduced-order models, in *AIAA SCITECH 2023 Forum*. Available at doi:10.2514/6.2023-0535.

M. Barrault, Y. Maday, N. C. Nguyen and A. T. Patera (2004), An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations, *C.R. Math. Acad. Sci. Paris* **339**, 667–672.

B. Battisti, T. Blickhan, G. Enchery, V. Ehrlacher, D. Lombardi and O. Mula (2023), Wasserstein model reduction approach for parametrized flow problems in porous media, *ESAIM Proc. Surveys* **73**, 28–47.

U. Baur, C. Beattie, P. Benner and S. Gugercin (2011), Interpolatory projection methods for parameterized model reduction, *SIAM J. Sci. Comput.* **33**, 2489–2518.

M. H. Beck, A. Jäckle, G. A. Worth and H.-D. Meyer (2000), The multiconfiguration time-dependent hartree (MCTDH) method: A highly efficient algorithm for propagating wavepackets, *Phys. Rep.* **324**, 1–105.

Y. Bengio, A. Courville and P. Vincent (2013), Representation learning: A review and new perspectives, *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1798–1828.

P. Benner and T. Breiten (2015), Two-sided projection methods for nonlinear model order reduction, *SIAM J. Sci. Comput.* **37**, B239–B260.

P. Benner and S. W. R. Werner (2020), Hankel-norm approximation of large-scale descriptor systems, *Adv. Comput. Math.* **46**, art. 40.

P. Benner, A. Cohen, M. Ohlberger and K. Willcox (2017), *Model Reduction and Approximation*, Advances in Design and Control, SIAM.

P. Benner, P. Goyal and S. Gugercin (2018), $\mathcal{H}_2$-quasi-optimal model order reduction for quadratic-bilinear control systems, *SIAM J. Matrix Anal. Appl.* **39**, 983–1032.

P. Benner, P. Goyal, J. Heiland and I. Pontes Duff (2023), A quadratic decoder approach to nonintrusive reduced-order modeling of nonlinear dynamical systems, *Proc. Appl. Math. Mech.* **23**, art. e202200049.

P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. Schilders and L. M. Silveira (2021*a*), *Model Order Reduction*, Vol. 1, *System- and Data-Driven Methods and Algorithms*, De Gruyter.

P. Benner, S. Grivet-Talocia, A. Quarteroni, G. Rozza, W. Schilders and L. M. Silveira (2021*b*), *Model Order Reduction*, Vol. 2, *Snapshot-Based Methods and Algorithms*, De Gruyter.

P. Benner, S. Gugercin and K. Willcox (2015), A survey of projection-based model reduction methods for parametric dynamical systems, *SIAM Rev.* **57**, 483–531.

A. Bensalah, A. Nouy and J. Soffo (2025), Nonlinear manifold approximation using compositional polynomial networks. Available at arXiv:2502.05088. To appear in *SIAM J. Sci. Comput.*

M. J. Berger and P. Colella (1989), Local adaptive mesh refinement for shock hydrodynamics, *J. Comput. Phys.* **82**, 64–84.

M. J. Berger and J. Oliger (1984), Adaptive mesh refinement for hyperbolic partial differential equations, *J. Comput. Phys.* **53**, 484–512.

J. Berman and B. Peherstorfer (2023), Randomized sparse Neural Galerkin schemes for solving evolution equations with deep networks, in *Advances in Neural Information Processing Systems 36* (A. Oh *et al.*, eds), Curran Associates, pp. 4097–4114.

J. Berman and B. Peherstorfer (2024), CoLoRA: Continuous low-rank adaptation for reduced implicit neural modeling of parameterized partial differential equations, in *Proceedings of the 41st International Conference on Machine Learning* (R. Salakhutdinov *et al.*, eds), Vol. 235 of Proceedings of Machine Learning Research, PMLR, pp. 3565–3583.

J. Berman, P. Schwerdtner and B. Peherstorfer (2024), Neural Galerkin schemes for sequential-in-time solving of partial differential equations with deep networks, in *Numerical Analysis Meets Machine Learning* (S. Mishra and A. Townsend, eds), Vol. 25 of Handbook of Numerical Analysis, Elsevier, pp. 389–418.

W. J. Beyn and V. Thümmler (2004), Freezing solutions of equivariant evolution equations, *SIAM J. Appl. Dyn. Syst.* **3**, 85–116.

M. Billaud-Friess and A. Nouy (2017), Dynamical model reduction method for solving parameter-dependent dynamical systems, *SIAM J. Sci. Comput.* **39**, A1766–A1792.

P. Binev, A. Cohen, W. Dahmen, R. DeVore, G. Petrova and P. Wojtaszczyk (2011), Convergence rates for greedy algorithms in reduced basis methods, *SIAM J. Math. Anal.* **43**, 1457–1472.

F. Black, P. Schulze and B. Unger (2020), Projection-based model reduction with dynamically transformed modes, *ESAIM Math. Model. Numer. Anal.* **54**, 2011–2043.

F. Black, P. Schulze and B. Unger (2021*a*), Efficient wildland fire simulation via nonlinear model order reduction, *Fluids* **6**, art. 280.

F. Black, P. Schulze and B. Unger (2021*b*), Model order reduction with dynamically transformed modes for the wave equation, *Proc. Appl. Math. Mech.* **20**, art. e202000321.

F. Black, P. Schulze and B. Unger (2022), Modal decomposition of flow data via gradient-based transport optimization, in *Active Flow and Combustion Control 2021* (R. King and D. Peitsch, eds), Springer, pp. 203–224.

T. Blickhan (2024), A registration method for reduced basis problems using linear optimal transport, *SIAM J. Sci. Comput.* **46**, A3177–A3204.

D. Bon, B. Caris and O. Mula (2025), Stable nonlinear dynamical approximation with dynamical sampling. Available at arXiv:2505.11938.

A. Bonito, A. Cohen, R. DeVore, D. Guignard, P. Jantsch and G. Petrova (2021), Nonlinear methods for model reduction, *ESAIM Math. Model. Numer. Anal.* **55**, 507–531.

D. Braess (1986), *Nonlinear Approximation Theory*, Springer.

J. Broeckhove, L. Lathouwers, E. Kesteloot and P. Van Leuven (1988), On the equivalence of time-dependent variational principles, *Chem. Phys. Lett.* **149**, 547–550.

J. Bruna, B. Peherstorfer and E. Vanden-Eijnden (2024), Neural Galerkin schemes with active learning for high-dimensional evolution equations, *J. Comput. Phys.* **496**, art. 112588.

J. Brunken, K. Smetana and K. Urban (2019), (Parametrized) first order transport equations: Realization of optimally stable Petrov–Galerkin methods, *SIAM J. Sci. Comput.* **41**, A592–A621.

P. Buchfink, S. Glas and B. Haasdonk (2023), Symplectic model reduction of Hamiltonian systems on nonlinear manifolds and approximation with weakly symplectic autoencoder, *SIAM J. Sci. Comput.* **45**, A289–A311.

P. Buchfink, S. Glas and B. Haasdonk (2024*a*), Approximation bounds for model reduction on polynomially mapped manifolds, *C.R. Math. Acad. Sci. Paris* **362**, 1211–1226.

P. Buchfink, S. Glas, B. Haasdonk and B. Unger (2024*b*), Model reduction on manifolds: A differential geometric framework, *Phys. D* **468**, art. 134299.

A. Buffa, Y. Maday, A. T. Patera, C. Prud'homme and G. Turinici (2012), *A priori* convergence of the greedy algorithm for the parametrized reduced basis method, *ESAIM Math. Model. Numer. Anal.* **46**, 595–603.

A. Buhr and K. Smetana (2018), Randomized local model order reduction, *SIAM J. Sci. Comput.* **40**, A2120–A2151.

A. Buhr, C. Engwer, M. Ohlberger and S. Rave (2017), ArbiLoMod, a simulation technique designed for arbitrary local modifications, *SIAM J. Sci. Comput.* **39**, A1435–A1465.

T. Bui-Thanh, K. Willcox, O. Ghattas and B. van Bloemen Waanders (2007), Goal-oriented, model-constrained optimization for reduction of large-scale systems, *J. Comput. Phys.* **224**, 880–896.

S. Burela, P. Krah and J. Reiss (2025), Parametric model order reduction for a wildland fire model via the shifted POD-based deep learning method, *Adv. Comput. Math.* **51**, art. 9.

N. Cagniart, Y. Maday and B. Stamm (2019), Model order reduction for problems with large convection effects, in *Contributions to Partial Differential Equations and Applications* (B. N. Chetverushkin *et al.*, eds), Computational Methods in Applied Sciences, Springer, pp. 131–150.

X. Cao, M. B. Saltik and S. Weiland (2015), Hankel model reduction for descriptor systems, in *54th IEEE Conference on Decision and Control (CDC)*, pp. 4668–4673. Available at doi:10.1109/CDC.2015.7402947.

K. Carlberg (2015), Adaptive *h*-refinement for reduced-order models, *Internat. J. Numer. Methods Engrg* **102**, 1192–1210.

K. Carlberg, M. Barone and H. Antil (2017), Galerkin v. least-squares Petrov–Galerkin projection in nonlinear model reduction, *J. Comput. Phys.* **330**, 693–734.

K. Carlberg, C. Bou-Mosleh and C. Farhat (2011), Efficient non-linear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations, *Internat. J. Numer. Methods Engrg* **86**, 155–181.

B. Carrel (2026), Randomized methods for dynamical low-rank approximation, *J. Comput. Phys.* **544**, art. 114421.

M. Cenedese, J. Axås, B. Bäuerlein, K. Avila and G. Haller (2022), Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds, *Nat. Commun.* **13**, art. 872.

G. Ceruti and C. Lubich (2022), An unconventional robust integrator for dynamical low-rank approximation, *BIT Numer. Math.* **62**, 23–44.

G. Ceruti, L. Einkemmer, J. Kusch and C. Lubich (2024*a*), A robust second-order low-rank BUG integrator based on the midpoint rule, *BIT Numer. Math.* **64**, art. 30.

G. Ceruti, J. Kusch and C. Lubich (2022), A rank-adaptive robust integrator for dynamical low-rank approximation, *BIT Numer. Math.* **62**, 1149–1174.

G. Ceruti, J. Kusch and C. Lubich (2024*b*), A parallel rank-adaptive integrator for dynamical low-rank approximation, *SIAM J. Sci. Comput.* **46**, B205–B228.

A. Charous and P. F. J. Lermusiaux (2023), Dynamically orthogonal Runge–Kutta schemes with perturbative retractions for the dynamical low-rank approximation, *SIAM J. Sci. Comput.* **45**, A872–A897.

A. Charous and P. F. J. Lermusiaux (2024), Stable rank-adaptive dynamically orthogonal Runge–Kutta schemes, *SIAM J. Sci. Comput.* **46**, A529–A560.

S. Chaturantabut and D. C. Sorensen (2010), Nonlinear model reduction via discrete empirical interpolation, *SIAM J. Sci. Comput.* **32**, 2737–2764.

H. Chen, R. Wu, E. Grinspun, C. Zheng and P. Y. Chen (2023*a*), Implicit neural spatial representations for time-dependent PDEs, in *Proceedings of the 40th International Conference on Machine Learning* (A. Krause *et al.*, eds), Vol. 202 of Proceedings of Machine Learning Research, PMLR, pp. 5162–5177.

P. Y. Chen, J. Xiang, D. H. Cho, Y. Chang, G. A. Pershing, H. T. Maia, M. M. Chiaramonte, K. T. Carlberg and E. Grinspun (2023*b*), CROM: Continuous reduced-order modeling of PDEs using implicit neural representations, in *Eleventh International Conference on Learning Representations (ICLR 2023)*. Available at https://openreview.net/forum?id=FUORz1tG8Og.

Y. Chen, S. Gottlieb, L. Ji and Y. Maday (2021), An EIM-degradation free reduced basis method via over collocation and residual hyper reduction-based error estimation, *J. Comput. Phys.* **444**, art. 110545.

Z. Chen, J. Mccarran, E. Vizcaino, M. Soljacic and D. Luo (2024), TENG: Time-evolving natural gradient for solving PDEs with deep neural nets toward machine precision, in *Proceedings of the 41st International Conference on Machine Learning* (R. Salakhutdinov *et al.*, eds), Vol. 235 of Proceedings of Machine Learning Research, PMLR, pp. 7143–7162.

W. Cho, K. Lee, D. Rim and N. Park (2023), Hypernetwork-based meta-learning for low-rank physics-informed neural networks, in *Thirty-Seventh Conference on Neural Information Processing Systems*. Available at https://openreview.net/forum?id=dzqKAM2sKa.

A. Cohen and R. DeVore (2015), Approximation of high-dimensional parametric PDEs, *Acta Numer.* **24**, 1–159.

A. Cohen and R. DeVore (2016), Kolmogorov widths under holomorphic mappings, *IMA J. Numer. Anal.* **36**, 1–12.

A. Cohen, W. Dahmen, R. DeVore and J. Nichols (2020), Reduced basis greedy selection using random training sets, *ESAIM Math. Model. Numer. Anal.* **54**, 1509–1524.

A. Cohen, W. Dahmen, O. Mula and J. Nichols (2022*a*), Nonlinear reduced models for state and parameter estimation, *SIAM/ASA J. Uncertain. Quantif.* **10**, 227–267.

A. Cohen, R. DeVore and C. Schwab (2011), Analytic regularity and polynomial approximation of parametric and stochastic elliptic PDEs, *Anal. Appl.* **9**, 11–47.

A. Cohen, R. DeVore, G. Petrova and P. Wojtaszczyk (2022*b*), Optimal stable nonlinear approximation, *Found. Comput. Math* **22**, 607–648.

A. Cohen, C. Farhat, Y. Maday and A. Somacal (2024), Nonlinear compressive reduced basis approximation for PDE's, *C.R. Méc.* **352**, 267–279.

P. G. Constantine (2015), *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*, SIAM Spotlights, SIAM.

D. M. Copeland, S. W. Cheung, K. Huynh and Y. Choi (2022), Reduced order models for Lagrangian hydrodynamics, *Comput. Methods Appl. Mech. Engrg* **388**, art. 114259.

A. Cortinovis, D. Kressner, S. Massei and B. Peherstorfer (2020), Quasi-optimal sampling to learn basis updates for online adaptive model reduction with adaptive empirical interpolation, in *American Control Conference (ACC) 2020*, IEEE, pp. 2472–2477.

J. Coughlin, J. Hu and U. Shumlak (2024), Robust and conservative dynamical low-rank methods for the Vlasov equation via a novel macro–micro decomposition, *J. Comput. Phys.* **509**, art. 113055.

W. Dahmen, C. Plesken and G. Welper (2014), Double greedy algorithms: Reduced basis methods for transport dominated problems, *ESAIM Math. Model. Numer. Anal.* **48**, 623–663.

T. Daniel, F. Casenave, N. Akkari and D. Ryckelynck (2020), Model order reduction assisted by deep neural networks (ROM-net), *Adv. Model. Simul. Engrg Sci.* **7**, art. 16.

F. de Avila Belbute-Peres, Y. Chen and F. Sha (2021), HyperPINN: Learning parameterized differential equations with physics-informed hypernetworks, in *The Symbiosis of Deep Learning and Differential Equations*. Available at https://openreview.net/forum?id=LxUuRDUhRjM.

P. Deuflhard and M. Weiser (2012), *Adaptive Numerical Solution of PDEs*, De Gruyter.

R. A. DeVore (1998), Nonlinear approximation, *Acta Numer.* **7**, 51–150.

R. A. DeVore, G. Kyriazis, D. Leviatan and V. M. Tikhomirov (1993), Wavelet compression and nonlinear n-widths, *Adv. Comput. Math.* **1**, 197–214.

R. DeVore, B. Hanin and G. Petrova (2021), Neural network approximation, *Acta Numer.* **30**, 327–444.

M. Dihlmann, M. Drohmann and B. Haasdonk (2011), Model reduction of parametrized evolution problems using the reduced basis method with adaptive time partitioning, in *Adaptive Modeling and Simulation 2011 (ADMOS 2011)* (D. Aubry *et al.*, eds), International Center for Numerical Methods in Engineering (CIMNE).

P. A. Dirac (1930), Note on exchange phenomena in the Thomas atom, *Math. Proc. Camb. Phil. Soc.* **26**, 376–385.

M.-H. Do, J. Feydy and O. Mula (2025), Sparse Wasserstein barycenters and application to reduced order modeling, *J. Sci. Comput.* **102**, art. 64.

Y. Dong, P. Schwerdtner and B. Peherstorfer (2025), Randomized time stepping of nonlinearly parametrized solutions of evolution problems. Available at arXiv:2512.19009.

Z. Drmač and S. Gugercin (2016), A new selection operator for the Discrete Empirical Interpolation Method: Improved *a priori* error bound and extensions, *SIAM J. Sci. Comput.* **38**, A631–A648.

M. Drohmann, B. Haasdonk and M. Ohlberger (2011), Adaptive reduced basis methods for nonlinear convection–diffusion equations, in *Finite Volumes for Complex Applications VI Problems & Perspectives* (J. Fořt *et al.*, eds), Springer, pp. 369–377.

M. Drohmann, B. Haasdonk and M. Ohlberger (2012), Reduced basis approximation for nonlinear parametrized evolution equations based on empirical operator interpolation, *SIAM J. Sci. Comput.* **34**, A937–A969.

Y. Du and T. A. Zaki (2021), Evolutional deep neural network, *Phys. Rev. E* **104**, art. 045303.

J. L. Eftang and B. Stamm (2012), Parameter multi-domain 'hp' empirical interpolation, *Internat. J. Numer. Methods Engrg* **90**, 412–428.

J. L. Eftang, D. J. Knezevic and A. T. Patera (2011), An *hp* certified reduced basis method for parametrized parabolic partial differential equations, *Math. Comput. Model. Dyn. Sys.* **17**, 395–422.

J. L. Eftang, A. T. Patera and E. M. Rønquist (2010), An *hp* certified reduced basis method for parametrized elliptic partial differential equations, *SIAM J. Sci. Comput.* **32**, 3170–3200.

V. Ehrlacher, D. Lombardi, O. Mula and F.-X. Vialard (2020), Nonlinear model reduction on metric spaces: Application to one-dimensional conservative PDEs in Wasserstein spaces, *ESAIM Math. Model. Numer. Anal.* **54**, 2159–2197.

L. Einkemmer and I. Joseph (2021), A mass, momentum, and energy conservative dynamical low-rank scheme for the Vlasov equation, *J. Comput. Phys.* **443**, art. 110495.

L. Einkemmer and C. Lubich (2019), A quasi-conservative dynamical low-rank algorithm for the Vlasov equation, *SIAM J. Sci. Comput.* **41**, B1061–B1081.

L. Einkemmer, J. Hu and Y. Wang (2021), An asymptotic-preserving dynamical low-rank method for the multi-scale multi-dimensional linear transport equation, *J. Comput. Phys.* **439**, art. 110353.

L. Einkemmer, K. Kormann, J. Kusch, R. G. McClarren and J.-M. Qiu (2025), A review of low-rank methods for time-dependent kinetic simulations, *J. Comput. Phys.* **538**, art. 114191.
L. Einkemmer, A. Ostermann and C. Scalone (2023), A robust and conservative dynamical low-rank algorithm, *J. Comput. Phys.* **484**, art. 112060.
C. Engwer, M. Ohlberger and L. Renelt (2024), Model order reduction of an ultraweak and optimally stable variational formulation for parametrized reactive transport problems, *SIAM J. Sci. Comput.* **46**, A3205–A3229.
A. Ern and J.-L. Guermond (2004), *Theory and Practice of Finite Elements*, Springer.
P. A. Etter and K. T. Carlberg (2020), Online adaptive basis refinement and compression for reduced-order models via vector-space sieving, *Comput. Methods Appl. Mech. Engrg* **364**, art. 112931.
R. Everson and L. Sirovich (1995), The Karhunen–Loève procedure for gappy data, *J. Opt. Soc. Amer. A* **12**, 1657–1664.
C. Farhat, P. Avery, T. Chapman and J. Cortial (2014), Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency, *Internat. J. Numer. Methods Engrg* **98**, 625–662.
C. Farhat, T. Chapman and P. Avery (2015), Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models, *Internat. J. Numer. Methods Engrg* **102**, 1077–1110.
M. Feischl, C. Lasser, C. Lubich and J. Nick (2024), Regularized dynamical parametric approximation. Available at arXiv:2403.19234.
F. Feppon and P. F. J. Lermusiaux (2018), Dynamically orthogonal numerical schemes for efficient stochastic advection and Lagrangian transport, *SIAM Rev.* **60**, 595–625.
A. Ferrero, T. Taddei and L. Zhang (2022), Registration-based model reduction of parameterized two-dimensional conservation laws, *J. Comput. Phys.* **457**, art. 111068.
M. A. Finzi, A. Potapczynski, M. Choptuik and A. G. Wilson (2023), A stable and scalable method for solving initial value PDEs with neural networks, in *Eleventh International Conference on Learning Representations (ICLR 2023)*. Available at https://openreview.net/forum?id=vsMyHUq_C1c.
B. A. Francis (1987), *A Course in H_∞ Control Theory*, Springer.
N. R. Franco, A. Manzoni and P. Zunino (2023), A deep learning approach to Reduced Order Modelling of parameter dependent partial differential equations, *Math. Comp.* **92**, 483–524.
J. Frenkel (1934), *Wave Mechanics: Advanced General Theory*, Oxford.
S. Fresca and A. Manzoni (2022), POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, *Comput. Methods Appl. Mech. Engrg* **388**, art. 114181.
S. Fresca, L. Dede' and A. Manzoni (2021), A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs, *J. Sci. Comput.* **87**, art. 61.
R. Geelen and K. Willcox (2022), Localized non-intrusive reduced-order modelling in the operator inference framework, *Philos. Trans. R. Soc. A* **380**(2229), art. 20210206.
R. Geelen, L. Balzano and K. Willcox (2023*a*), Learning latent representations in high-dimensional state spaces using polynomial manifold constructions, in *62nd IEEE Conference on Decision and Control (CDC)*, pp. 4960–4965. Available at doi:10.1109/CDC49753.2023.10384209.

R. Geelen, L. Balzano, S. Wright and K. Willcox (2024), Learning physics-based reduced-order models from data using nonlinear manifolds, *Chaos* **34**, art. 033122.
R. Geelen, S. Wright and K. Willcox (2023*b*), Operator inference for non-intrusive model reduction with quadratic manifolds, *Comput. Methods Appl. Mech. Engrg* **403**, art. 115717.
J.-F. Gerbeau and D. Lombardi (2014), Approximated Lax pairs for the reduced order integration of nonlinear evolution equations, *J. Comput. Phys.* **265**, 246–269.
O. Ghattas and K. Willcox (2021), Learning physics-based models from data: Perspectives from inverse problems and model reduction, *Acta Numer.* **30**, 445–554.
S. Glas and H. Mu (2025), Piece-wise symplectic model reduction on quadratically embedded manifolds, in *Numerical Mathematics and Advanced Applications (ENUMATH 2023)* (A. Sequeira *et al.*, eds), Vol. 1, Springer, pp. 355–364.
S. Glas and B. Unger (2025), Leveraging time and parameters for nonlinear model reduction methods, *IFAC-PapersOnLine* **59**, 550–555.
S. Glas, A. Mayerhofer and K. Urban (2017), Two ways to treat time in reduced basis methods, in *Model Reduction of Parametrized Systems* (P. Benner *et al.*, eds), Springer, pp. 1–16.
K. Glover (1984), All optimal Hankel-norm approximations of linear multivariable systems and their l^∞-error bounds, *Internat. J. Control* **39**, 1115–1193.
I. Goodfellow, Y. Bengio and A. Courville (2016), *Deep Learning*, MIT Press. Available at http://www.deeplearningbook.org.
I. V. Gosea, S. Gugercin and B. Unger (2021), Parametric model reduction via rational interpolation along parameters, in *60th IEEE Conference on Decision and Control (CDC)*, pp. 6895–6900. Available at doi:10.1109/CDC45484.2021.9682841.
H. Gowrachari, N. Demo, G. Stabile and G. Rozza (2025), Non-intrusive model reduction of advection-dominated hyperbolic problems using neural network shift augmented manifold transformation, *Comput. Fluids* **300**, art. 106758.
P. Goyal and P. Benner (2024), Generalized quadratic embeddings for nonlinear dynamics using deep learning, *Phys. D* **463**, art. 134158.
C. Greif and K. Urban (2019), Decay of the Kolmogorov n-width for wave problems, *Appl. Math. Lett.* **96**, 216–222.
C. Greif, P. Junk and K. Urban (2022), Linear/ridge expansions: Enhancing linear approximations by ridge functions, *Adv. Comput. Math.* **48**, art. 15.
M. A. Grepl and A. T. Patera (2005), *A posteriori* error bounds for reduced-basis approximations of parametrized parabolic partial differential equations, *ESAIM Math. Model. Numer. Anal.* **39**, 157–181.
S. Grimberg, C. Farhat, R. Tezaur and C. Bou-Mosleh (2021), Mesh sampling and weighting for the hyperreduction of nonlinear Petrov–Galerkin reduced-order models with local reduced-order bases, *Internat. J. Numer. Methods Engrg* **122**, 1846–1874.
A. Gruber, M. Gunzburger, L. Ju and Z. Wang (2022), A comparison of neural network architectures for data-driven reduced-order modeling, *Comput. Methods Appl. Mech. Engrg* **393**, art. 114764.
R. Gruhlke, A. Nouy and P. Trunschke (2024), Optimal sampling for stochastic and natural gradient descent. Available at arXiv:2402.03113.
S. Grundel and N. Sarna (2024), Hyper-reduction for parametrized transport dominated problems via adaptive reduced meshes, *Partial Differ. Equ. Appl.* **5**, art. 3.
C. Gu (2011), QLMOR: A projection-based nonlinear model order reduction approach using quadratic–linear representation of nonlinear systems, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **30**, 1307–1320.

C. Gu (2012), Model order reduction of nonlinear dynamical systems. PhD thesis, UCB/EECS-2012-217, EECS Department, University of California, Berkeley.
M. Gubisch and S. Volkwein (2017), Proper orthogonal decomposition for linear-quadratic optimal control, in *Model Reduction and Approximation* (P. Benner *et al.*, eds), SIAM, pp. 3–63.
N. Guglielmi and M. Manucci (2023), Model order reduction in contour integral methods for parametric PDEs, *SIAM J. Sci. Comput.* **45**, A1711–A1740.
N. Guglielmi, M. López-Fernández and M. Manucci (2021), Pseudospectral roaming contour integral methods for convection–diffusion equations, *J. Sci. Comput.* **89**, art. 22.
B. Haasdonk (2017), Reduced basis methods for parametrized PDEs: A tutorial introduction for stationary and instationary problems, in *Model Reduction and Approximation* (P. Benner *et al.*, eds), SIAM, pp. 65–136.
B. Haasdonk and M. Ohlberger (2008), Reduced basis method for finite volume approximations of parametrized linear evolution equations, *ESAIM Math. Model. Numer. Anal.* **42**, 277–302.
B. Haasdonk and M. Ohlberger (2011), Efficient reduced models and *a posteriori* error estimation for parametrized dynamical systems by offline/online decomposition, *Math. Comput. Model. Dyn. Sys.* **17**, 145–161.
B. Haasdonk, M. Dihlmann and M. Ohlberger (2011), A training set and multiple bases generation approach for parameterized model reduction based on adaptive grids in parameter space, *Math. Comput. Model. Dyn. Sys.* **17**, 423–442.
B. Haasdonk, M. Ohlberger and G. Rozza (2008), A reduced basis method for evolution schemes with parameter-dependent explicit operators, *Electron. Trans. Numer. Anal.* **32**, 145–161.
D. Hartman and L. K. Mestha (2017), A deep learning framework for model reduction of dynamical systems, in *2017 IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, pp. 1917–1922.
C. D. Hauck and S. Schnake (2023), A predictor–corrector strategy for adaptivity in dynamical low-rank approximations, *SIAM J. Matrix Anal. Appl.* **44**, 971–1005.
E. J. Heller (1976), Time dependent variational approach to semiclassical dynamics, *J. Chem. Phys.* **64**, 63–73.
R. Herkert, P. Buchfink and B. Haasdonk (2024), Dictionary-based online-adaptive structure-preserving model order reduction for parametric Hamiltonian systems, *Adv. Comput. Math.* **50**, art. 12.
M. Hess, A. Alla, A. Quaini, G. Rozza and M. Gunzburger (2019), A localized reduced-order modeling approach for PDEs with bifurcating solutions, *Comput. Methods Appl. Mech. Engrg* **351**, 379–403.
J. S. Hesthaven and C. Pagliantini (2021), Structure-preserving reduced basis methods for Poisson systems, *Math. Comp.* **90**, 1701–1740.
J. S. Hesthaven, C. Pagliantini and N. Ripamonti (2022*a*), Rank-adaptive structure-preserving model order reduction of Hamiltonian systems, *ESAIM Math. Model. Numer. Anal.* **56**, 617–650.
J. S. Hesthaven, C. Pagliantini and N. Ripamonti (2024), Adaptive symplectic model order reduction of parametric particle-based Vlasov–Poisson equation, *Math. Comp.* **93**, 1153–1202.
J. S. Hesthaven, C. Pagliantini and G. Rozza (2022*b*), Reduced basis methods for time-dependent problems, *Acta Numer.* **31**, 265–345.

J. S. Hesthaven, G. Rozza and B. Stamm (2016), *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*, SpringerBriefs in Mathematics, Springer.
J. S. Hesthaven, B. Stamm and S. Zhang (2014), Efficient greedy algorithms for high-dimensional parameter spaces with applications to empirical interpolation and reduced basis methods, *ESAIM Math. Model. Numer. Anal.* **48**, 259–283.
B. Hillebrecht and B. Unger (2025*a*), Prediction error certification for PINNs: Theory, computation, and application to Stokes flow. Available at arXiv:2508.07994.
B. Hillebrecht and B. Unger (2025*b*), Rigorous *a posteriori* error bounds for PDE-defined PINNs, *IEEE Trans. Neural Netw. Learn. Syst.* **36**, 1583–1593.
M. Hochbruck, M. Neher and S. Schrammer (2023), Rank-adaptive dynamical low-rank integrators for first-order and second-order matrix differential equations, *BIT Numer. Math.* **63**, art. 9.
P. Holmes, J. L. Lumley and G. Berkooz (1996), *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge University Press.
E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang and W. Chen (2022), LoRA: Low-rank adaptation of large language models, in *International Conference on Learning Representations (ICLR 2022)*. Available at https://openreview.net/forum?id=nZeVKeeFYf9.
C. Huang and K. Duraisamy (2023), Predictive reduced order modeling of chaotic multi-scale problems using adaptively sampled projections, *J. Comput. Phys.* **491**, art. 112356.
C. Huang, J. Xu, K. Duraisamy and C. Merkle (2018), Exploration of reduced-order models for rocket combustion applications, in *2018 AIAA Aerospace Sciences Meeting*. Available at doi:10.2514/6.2018-1183.
M. Hund, T. Mitchell, P. Mlinarić and J. Saak (2022), Optimization-based parametric model order reduction via $\mathcal{H}_2 \otimes \mathcal{L}_2$ first-order necessary conditions, *SIAM J. Sci. Comput.* **44**, A1554–A1578.
A. Iollo and D. Lombardi (2014), Advection modes by optimal mass transfer, *Phys. Rev. E* **89**, art. 022923.
A. Iollo, J. Labatut, P. Mounoud and T. Taddei (2026), Mathematical aspects of registration methods in bounded domains. Available at arXiv:2601.03010.
O. Issan and B. Kramer (2023), Predicting solar wind streams from the inner-heliosphere to earth via shifted operator inference, *J. Comput. Phys.* **473**, art. 111689.
S. Jain, P. Tiso, J. B. Rutzmoser and D. J. Rixen (2017), A quadratic manifold for model order reduction of nonlinear structural dynamics, *Comput. Struct.* **188**, 80–94.
K. Kashima (2016), Nonlinear model reduction by deep autoencoder of noise response data, in *55th Conference on Decision and Control (CDC)*, pp. 5750–5755. Available at doi:10.1109/CDC.2016.7799153.
M. Kast and J. S. Hesthaven (2024), Positional embeddings for solving PDEs with evolutional deep neural networks, *J. Comput. Phys.* **508**, art. 112986.
S. Kaulmann and B. Haasdonk (2013), Online greedy reduced basis construction using dictionaries, in *Adaptive Modeling and Simulation 2013 (ADMOS 2013)* (J. P. Moitinho de Almeida *et al.*, eds), International Center for Numerical Methods in Engineering (CIMNE), pp. 365–376.
K. G. Kay (1989), The matrix singularity problem in the time-dependent variational method, *Chem. Phys.* **137**, 165–175.
Y. Kazashi, F. Nobile and F. Zoccolan (2025), Dynamical low-rank approximation for stochastic differential equations, *Math. Comp.* **94**, 1335–1375.

M. Khamlich, F. Pichi and G. Rozza (2025), Optimal transport–inspired deep learning framework for slow-decaying Kolmogorov n-width problems: Exploiting Sinkhorn loss and Wasserstein kernel, *SIAM J. Sci. Comput.* **47**, C235–C264.
E. Kieri, C. Lubich and H. Walach (2016), Discretized dynamical low-rank approximation in the presence of small singular values, *SIAM J. Numer. Anal.* **54**, 1020–1038.
Y. Kim, Y. Choi, D. Widemann and T. Zohdi (2022), A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder, *J. Comput. Phys.* **451**, art. 110841.
M. Kirby and D. Armbruster (1992), Reconstructing phase space from PDE simulations, *Z. Angew. Math. Phys.* **43**, 999–1022.
H. Kleikamp, M. Ohlberger and S. Rave (2022), Nonlinear model order reduction using diffeomorphic transformations of a space–time domain, in *Recent Advances in Model Reduction and Surrogate Modeling (MATHMOD 2022)*, ARGESIM.
B. Klein and M. Ohlberger (2026), Multi-fidelity learning of reduced order models for parabolic PDE constrained optimization, *Adv. Comput. Math.* **52**, art. 19.
R. Klein, B. Sanderse, P. Costa, R. Pecnik and R. Henkes (2025), Entropy-stable model reduction of one-dimensional hyperbolic systems using rational quadratic manifolds, *J. Comput. Phys.* **528**, art. 113817.
O. Koch and C. Lubich (2007), Dynamical low-rank approximation, *SIAM J. Matrix Anal. Appl.* **29**, 434–454.
O. Koch and C. Lubich (2010), Dynamical tensor approximation, *SIAM J. Matrix Anal. Appl.* **31**, 2360–2375.
A. Kolmogoroff (1936), Über die beste Annäherung von Funktionen einer gegebenen Funktionenklasse, *Ann. of Math. (2)* **37**, 107–110.
S. Kolouri, S. R. Park and G. K. Rohde (2016), The Radon cumulative distribution transform and it application to image classification, *IEEE Trans. Image Process.* **25**, 920–934.
P. Krah, S. Büchholz, M. Häringer and J. Reiss (2023), Front transport reduction for complex moving fronts, *J. Sci. Comput.* **96**, art. 28.
P. Krah, A. Marmin, B. Zorawski, J. Reiss and K. Schneider (2025), A robust shifted proper orthogonal decomposition: Proximal methods for decomposing flows with multiple transports, *SIAM J. Sci. Comput.* **47**, A633–A656.
B. Kramer and K. E. Willcox (2019), Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition, *AIAA J.* **57**, 2297–2307.
B. Kramer, B. Peherstorfer and K. E. Willcox (2024), Learning nonlinear reduced models from data with operator inference, *Annu. Rev. Fluid Mech.* **56**, 521–548.
P. Kramer and M. Saraceno (1981), *Geometry of the Time-Dependent Variational Principle in Quantum Mechanics*, Vol. 140 of Lecture Notes in Physics, Springer.
K. Kunisch and S. Volkwein (2008), Proper orthogonal decomposition for optimality systems, *ESAIM Math. Model. Numer. Anal.* **42**, 1–23.
S. Kvaal, C. Lasser, T. B. Pedersen and L. Adamowicz (2023), No need for a grid: Adaptive fully-flexible gaussians for the time-dependent Schrödinger equation. Available at arXiv:2207.00271.
H. Y. Lam, G. Ceruti and D. Kressner (2025), Randomized low-rank Runge–Kutta methods, *SIAM J. Matrix Anal. Appl.* **46**, 1587–1615.
C. Lasser and C. Lubich (2020), Computing quantum dynamics in the semiclassical regime, *Acta Numer.* **29**, 229–401.
Y. LeCun, Y. Bengio and G. Hinton (2015), Deep learning, *Nature* **521**(7553), 436–444.

K. Lee and K. T. Carlberg (2020), Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, *J. Comput. Phys.* **404**, art. 108973.
K. Lee and K. T. Carlberg (2021), Deep conservation: A latent-dynamics model for exact satisfaction of physical conservation laws, *Proc. AAAI Conf. Artif. Intell.* **35**, 277–285.
M. Lindsey (2025), MNE: Overparametrized neural evolution with applications to diffusion processes and sampling. Available at arXiv:2502.03645.
T. Long, R. Barnett, R. Jefferson-Loveday, G. Stabile and M. Icardi (2025), A reduced-order model for advection-dominated problems based on the Radon Cumulative Distribution Transform, *Adv. Comput. Math.* **51**, art. 5.
H. Lu and D. M. Tartakovsky (2020), Lagrangian dynamic mode decomposition for construction of reduced-order models of advection-dominated phenomena, *J. Comput. Phys.* **407**, art. 109229.
H. Lu and D. M. Tartakovsky (2021), Dynamic mode decomposition for construction of reduced-order models of hyperbolic problems with shocks, *J. Mach. Learn. Model. Comput.* **2**, 1–29.
C. Lubich (2005), On variational approximations in quantum molecular dynamics, *Math. Comp.* **74**, 765–779.
C. Lubich (2008), *From Quantum to Classical Molecular Dynamics: Reduced Models and Numerical Analysis*, Vol. 12 of Zurich Lectures in Advanced Mathematics, European Mathematical Society.
C. Lubich and J. Nick (2025), Regularized dynamical parametric approximation of stiff evolution problems. Available at arXiv:2501.12118.
C. Lubich and I. V. Oseledets (2014), A projector-splitting integrator for dynamical low-rank approximation, *BIT Numer. Math.* **54**, 171–188.
C. Lubich, I. V. Oseledets and B. Vandereycken (2015), Time integration of tensor trains, *SIAM J. Numer. Anal.* **53**, 917–941.
C. Lubich, T. Rohwedder, R. Schneider and B. Vandereycken (2013), Dynamical approximation by hierarchical Tucker and tensor-train tensors, *SIAM J. Matrix Anal. Appl.* **34**, 470–494.
Y. Maday and E. M. Rønquist (2002), A reduced-basis element method, *J. Sci. Comput.* **17**, 447–459.
Y. Maday, A. Manzoni and A. Quarteroni (2016), An online intrinsic stabilization strategy for the reduced basis approximation of parametrized advection-dominated problems, *C.R. Math. Acad. Sci. Paris* **354**, 1188–1194.
Y. Maday, A. T. Patera and G. Turinici (2002), Global *a priori* convergence theory for reduced-basis approximations of single-parameter symmetric coercive elliptic partial differential equations, *C.R. Math. Acad. Sci. Paris* **335**, 289–294.
R. Maulik, B. Lusch and P. Balaprakash (2021), Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders, *Phys. Fluids* **33**, art. 037106.
A. D. McLachlan (1964), A variational solution of the time-dependent Schrodinger equation, *Mol. Phys.* **8**, 39–44.
V. Mehrmann and B. Unger (2023), Control of port-Hamiltonian differential-algebraic systems and applications, *Acta Numer.* **32**, 395–515.
A. Mendible, S. L. Brunton, A. Y. Aravkin, W. Lowrie and J. N. Kutz (2020), Dimensionality reduction and reduced-order modeling for traveling wave physics, *Theor. Comput. Fluid Dyn.* **34**, 385–400.

H.-D. Meyer, U. Manthe and L. Cederbaum (1990), The multi-configurational time-dependent Hartree approach, *Chem. Phys. Lett.* **165**, 73–78.

K. Miller and R. N. Miller (1981), Moving finite elements I, *SIAM J. Numer. Anal.* **18**, 1019–1032.

A. Mohaghegh and C. Huang (2026), Feature-guided sampling strategy for adaptive model order reduction of convection-dominated problems, *J. Comput. Phys.* **545**, art. 114468.

R. Mojgani and M. Balajewicz (2017), Lagrangian basis method for dimensionality reduction of convection dominated nonlinear flows. Available at arXiv:1701.04343.

B. Moore (1981), Principal component analysis in linear systems: Controllability, observability, and model reduction, *IEEE Trans. Automat. Control* **26**, 17–32.

S. Mowlavi and T. P. Sapsis (2018), Model order reduction for stochastic dynamical systems with continuous symmetries, *SIAM J. Sci. Comput.* **40**, A1669–A1695.

C. Mullis and R. Roberts (1976), Synthesis of minimum roundoff noise fixed point digital filters, *IEEE Trans. Circuits Syst.* **23**, 551–562.

E. Musharbash and F. Nobile (2018), Dual dynamically orthogonal approximation of incompressible Navier Stokes equations with random boundary conditions, *J. Comput. Phys.* **354**, 135–162.

E. Musharbash, F. Nobile and E. Vidličková (2020), Symplectic dynamical low rank approximation of wave equations with random parameters, *BIT Numer. Math.* **60**, 1153–1201.

E. Musharbash, F. Nobile and T. Zhou (2015), Error analysis of the dynamically orthogonal approximation of time dependent random PDEs, *SIAM J. Sci. Comput.* **37**, A776–A810.

M. A. Nabian, R. J. Gladstone and H. Meidani (2021), Efficient training of physics-informed neural networks via importance sampling, *Comput.-Aided Civ. Engrg* **36**, 962–977.

M. H. Naderi and H. Babaee (2023), Adaptive sparse interpolation for accelerating nonlinear stochastic reduced-order modeling with time-dependent bases, *Comput. Methods Appl. Mech. Engrg* **405**, art. 115813.

J. L. Nicolini and F. L. Teixeira (2021), Reduced-order modeling of advection-dominated kinetic plasma problems by shifted proper orthogonal decomposition, *IEEE Trans. Plasma Sci.* **49**, 3689–3699.

F. Nobile and T. T. Trindade (2025*a*), Error estimates for SUPG-stabilised dynamical low rank approximations, in *Numerical Mathematics and Advanced Applications (ENUMATH 2023)* (A. Sequeira *et al.*, eds), Vol. 2, Springer, pp. 438–447.

F. Nobile and T. T. Trindade (2025*b*), Petrov–Galerkin dynamical low rank approximation: SUPG stabilisation of advection-dominated problems, *Comput. Methods Appl. Mech. Engrg* **433**, art. 117495.

M. Nonino, F. Ballarin, G. Rozza and Y. Maday (2023), A reduced basis method by means of transport maps for a fluid–structure interaction problem with slowly decaying Kolmogorov n-width, *Adv. Comput. Sci. Engrg* **1**, 36–58.

A. Nouy (2010), *A priori* model reduction through proper generalized decomposition for solving time-dependent partial differential equations, *Comput. Methods Appl. Mech. Engrg* **199**, 1603–1626.

A. Nouy and A. Pasco (2024), Dictionary-based model reduction for state estimation, *Adv. Comput. Math.* **50**, art. 32.

M. Ohlberger and S. Rave (2013), Nonlinear reduced basis approximation of parameterized evolution equations via the method of freezing, *C.R. Math. Acad. Sci. Paris* **351**, 901–906.

M. Ohlberger and S. Rave (2016), Reduced basis methods: Success, limitations and future challenges, in *Proceedings of the Conference Algoritmy*. Available at http://www.iam.fmph.uniba.sk/amuc/ojs/index.php/algoritmy/article/view/389.

M. Ohlberger and F. Schindler (2015), Error control for the localized reduced basis multiscale method with adaptive on-line enrichment, *SIAM J. Sci. Comput.* **37**, A2865–A2895.

M. Ohlberger and F. Schindler (2017), Non-conforming localized model reduction with online enrichment: Towards optimal complexity in PDE constrained optimization, in *Finite Volumes for Complex Applications VIII: Hyperbolic, Elliptic and Parabolic Problems* (C. Cancès and P. Omnes, eds), Springer, pp. 357–365.

S. E. Otto, G. R. Macchio and C. W. Rowley (2023), Learning nonlinear projections for reduced-order modeling of dynamical systems using constrained autoencoders, *Chaos* **33**, art. 113130.

S. E. Otto, A. Padovan and C. W. Rowley (2022), Optimizing oblique projections for nonlinear systems using trajectories, *SIAM J. Sci. Comput.* **44**, A1681–A1702.

P. Pacciarini and G. Rozza (2014), Stabilized reduced basis method for parametrized advection–diffusion PDEs, *Comput. Methods Appl. Mech. Engrg* **274**, 1–18.

C. Pagliantini (2021), Dynamical reduced basis methods for Hamiltonian systems, *Numer. Math.* **148**, 409–448.

C. Pagliantini and F. Vismara (2025), Fully adaptive structure-preserving hyper-reduction of parametric Hamiltonian systems, *SIAM J. Sci. Comput.* **47**, A124–A152.

C. Pagliantini and F. Vismara (2026), Adaptive hyper-reduction of non-sparse operators: Application to parametric particle-based kinetic plasma models, *J. Comput. Phys.* **549**, art. 114609.

S. Pan, S. L. Brunton and J. N. Kutz (2023), Neural implicit flow: A mesh-agnostic dimensionality reduction paradigm of spatio-temporal data, *J. Mach. Learn. Res.* **24**, 1–60.

D. Papapicco, N. Demo, M. Girfoglio, G. Stabile and G. Rozza (2022), The neural network shifted-proper orthogonal decomposition: A machine learning approach for non-linear reduction of hyperbolic equations, *Comput. Methods Appl. Mech. Engrg* **392**, art. 114687.

E. J. Parish and K. T. Carlberg (2021), Windowed least-squares model reduction for dynamical systems, *J. Comput. Phys.* **426**, art. 109939.

A. Pazy (1983), *Semigroups of Linear Operators and Applications to Partial Differential Equations*, Applied Mathematical Sciences, Springer.

B. Peherstorfer (2020), Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling, *SIAM J. Sci. Comput.* **42**, A2803–A2836.

B. Peherstorfer (2022), Breaking the Kolmogorov barrier with nonlinear model reduction, *Notices Amer. Math. Soc.* **69**, 725–733.

B. Peherstorfer and K. Willcox (2015*a*), Dynamic data-driven reduced-order models, *Comput. Methods Appl. Mech. Engrg* **291**, 21–41.

B. Peherstorfer and K. Willcox (2015*b*), Online adaptive model reduction for nonlinear systems via low-rank updates, *SIAM J. Sci. Comput.* **37**, A2123–A2150.

B. Peherstorfer, D. Butnaru, K. Willcox and H.-J. Bungartz (2014), Localized discrete empirical interpolation method, *SIAM J. Sci. Comput.* **36**, A168–A192.

B. Peherstorfer, Z. Drmač and S. Gugercin (2020), Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points, *SIAM J. Sci. Comput.* **42**, A2837–A2864.

B. Peherstorfer, K. Willcox and M. Gunzburger (2018), Survey of multifidelity methods in uncertainty propagation, inference, and optimization, *SIAM Rev.* **60**, 550–591.

E. Perez, F. Strub, H. de Vries, V. Dumoulin and A. Courville (2018), FiLM: Visual reasoning with a general conditioning layer, in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18, AAAI Press.

D. B. Phuong Huynh, D. J. Knezevic and A. T. Patera (2013), A static condensation reduced basis element method: Approximation and *a posteriori* error estimation, *ESAIM Math. Model. Numer. Anal.* **47**, 213–251.

A. Pinkus (1985), *n-Widths in Approximation Theory*, Ergebnisse der Mathematik und ihrer Grenzgebiete, Springer.

A. Pinkus (1999), Approximation theory of the MLP model in neural networks, *Acta Numer.* **8**, 143–195.

C. Prud'homme, D. V. Rovas, K. Veroy, L. Machiels, Y. Maday, A. T. Patera and G. Turinici (2001), Reliable real-time solution of parametrized partial differential equations: Reduced-basis output bound methods, *J. Fluids Engrg* **124**, 70–80.

E. Qian, M. Grepl, K. Veroy and K. Willcox (2017), A certified trust region reduced basis approach to PDE-constrained optimization, *SIAM J. Sci. Comput.* **39**, S434–S460.

E. Qian, B. Kramer, B. Peherstorfer and K. Willcox (2020), Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems, *Phys. D* **406**, art. 132401.

A. Quarteroni and G. Rozza, eds (2014), *Reduced Order Methods for Modeling and Computational Reduction*, Modeling, Simulation and Applications, Springer.

A. Quarteroni, A. Manzoni and F. Negri (2016), *Reduced Basis Methods for Partial Differential Equations: An Introduction*, UNITEXT, Springer.

M. Raissi, P. Perdikaris and G. Karniadakis (2019), Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *J. Comput. Phys.* **378**, 686–707.

D. Ramezanian, A. G. Nouri and H. Babaee (2021), On-the-fly reduced order modeling of passive and reactive species via time-dependent manifolds, *Comput. Methods Appl. Mech. Engrg* **382**, art. 113882.

M.-L. Rapún and J. M. Vega (2010), Reduced order models based on local POD plus Galerkin projection, *J. Comput. Phys.* **229**, 3046–3063.

A. H. Razavi and M. Yano (2025), Registration-based nonlinear model reduction of parametrized aerodynamics problems with applications to transonic Euler and RANS flows, *J. Comput. Phys.* **521**, art. 113576.

F. Regazzoni, S. Pagani, M. Salvador, L. Dede' and A. Quarteroni (2024), Learning the intrinsic dynamics of spatio-temporal processes through latent dynamics networks, *Nat. Commun.* **15**, art. 1834.

J. Reiss (2021), Optimization-based modal decomposition for systems with multiple transports, *SIAM J. Sci. Comput.* **43**, A2079–A2101.

J. Reiss, P. Schulze, J. Sesterhenn and V. Mehrmann (2018), The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena, *SIAM J. Sci. Comput.* **40**, A1322–A1344.

J. Ren, W. R. Wolf and X. Mao (2021), Model reduction of traveling-wave problems via Radon cumulative distribution transform, *Phys. Rev. Fluids* **6**, art. L082501.

D. Rim (2018), Dimensional splitting of hyperbolic partial differential equations using the Radon transform, *SIAM J. Sci. Comput.* **40**, A4184–A4207.
D. Rim and G. Welper (2026), A low-rank neural representation of entropy solutions, *Adv. Comput. Math.* **52**, art. 2.
D. Rim, S. Moe and R. J. LeVeque (2018), Transport reversal for model reduction of hyperbolic partial differential equations, *SIAM/ASA J. Uncertain. Quantif.* **6**, 118–150.
D. Rim, B. Peherstorfer and K. T. Mandli (2023), Manifold approximations via transported subspaces: Model reduction for transport-dominated problems, *SIAM J. Sci. Comput.* **45**, A170–A199.
F. Romor, G. Stabile and G. Rozza (2023), Non-linear manifold reduced-order models with convolutional autoencoders and reduced over-collocation method, *J. Sci. Comput.* **94**, 74.
E. Rothe (1930), Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben, *Math. Ann.* **102**, 650–670.
G. M. Rotskoff, A. R. Mitchell and E. Vanden-Eijnden (2022), Active importance sampling for variational objectives dominated by rare events: Consequences for optimization and generalization, in *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference* (J. Bruna *et al.*, eds), Vol. 145 of Proceedings of Machine Learning Research, PMLR, pp. 757–780.
K. Rowan, L. Schatzki, T. Zaklama, Y. Suzuki, K. Watanabe and K. Varga (2020), Simulation of a hydrogen atom in a laser field using the time-dependent variational principle, *Phys. Rev. E* **101**, art. 023313.
C. W. Rowley and J. E. Marsden (2000), Reconstruction equations and the Karhunen–Loève expansion for systems with symmetry, *Phys. D* **142**, 1–19.
C. W. Rowley, I. G. Kevrekidis, J. E. Marsden and K. Lust (2003), Reduction and reconstruction for self-similar dynamical systems, *Nonlinearity* **16**, art. 1257.
G. Rozza, D. B. P. Huynh and A. T. Patera (2008), Reduced basis approximation and *a posteriori* error estimation for affinely parametrized elliptic coercive partial differential equations, *Arch. Comput. Methods Engrg* **15**, 229–275.
D. E. Rumelhart, G. E. Hinton and R. J. Williams (1986), Learning representations by back-propagating errors, *Nature* **323**(6088), 533–536.
J. B. Rutzmoser, D. J. Rixen, P. Tiso and S. Jain (2017), Generalization of quadratic manifolds for reduced order modeling of nonlinear structural dynamics, *Comput. Struct.* **192**, 196–209.
T. P. Sapsis and P. F. J. Lermusiaux (2009), Dynamically orthogonal field equations for continuous stochastic dynamical systems, *Phys. D* **238**, 2347–2360.
N. Sarna and P. Benner (2021), Data-driven model order reduction for problems with parameter-dependent jump-discontinuities, *Comput. Methods Appl. Mech. Engrg* **387**, art. 114168.
N. Sarna, J. Giesselmann and P. Benner (2024), Data-driven snapshot calibration via monotonic feature matching, *Finite Elem. Anal. Des* **230**, art. 104065.
S. Sawada, R. Heather, B. Jackson and H. Metiu (1985), A strategy for time dependent quantum mechanical calculations using a Gaussian wave packet representation of the wave function, *J. Chem. Phys.* **83**, 3009–3027.
M. Schlegel and B. R. Noack (2015), On long-term boundedness of Galerkin models, *J. Fluid Mech.* **765**, 325–352.
B. Schölkopf and A. J. Smola (2001), *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press.

P. Schulze (2023), Energy-based model reduction of transport-dominated phenomena. Dissertation, Technische Universität Berlin.

P. Schulze, J. Reiss and V. Mehrmann (2019), Model reduction for a pulsed detonation combuster via shifted proper orthogonal decomposition, in *Active Flow and Combustion Control 2018* (R. King, ed.), Springer, pp. 271–286.

P. Schulze, B. Unger, C. Beattie and S. Gugercin (2018), Data-driven structured realization, *Linear Algebra Appl.* **537**, 250–286.

P. Schwerdtner and B. Peherstorfer (2024), Greedy construction of quadratic manifolds for nonlinear dimensionality reduction and nonlinear model reduction. Available at arXiv:2403.06732.

P. Schwerdtner, S. Gugercin and B. Peherstorfer (2025*a*), Empirical sparse regression on quadratic manifolds, *SIAM J. Sci. Comput.* **47**, A3085–A3107.

P. Schwerdtner, P. Mohan, A. Pachalieva, J. Bessac, D. O'Malley and B. Peherstorfer (2025*b*), Online learning of quadratic manifolds from streaming data for nonlinear dimensionality reduction and nonlinear model reduction, *Proc. R. Soc. A* **481**(2314), art. 20240670.

P. Schwerdtner, P. Schulze, J. Berman and B. Peherstorfer (2024), Nonlinear embeddings for conserving Hamiltonians and other quantities with Neural Galerkin schemes, *SIAM J. Sci. Comput.* **46**, C583–C607.

J. Sesterhenn and A. Shahirpour (2019), A characteristic dynamic mode decomposition, *Theor. Comput. Fluid Dyn.* **33**, 281–305.

H. Sharma, H. Mu, P. Buchfink, R. Geelen, S. Glas and B. Kramer (2023), Symplectic model reduction of Hamiltonian systems using data-driven quadratic manifolds, *Comput. Methods Appl. Mech. Engrg* **417**, art. 116402.

Y. S. Shimizu and E. J. Parish (2021), Windowed space–time least-squares Petrov–Galerkin model order reduction for nonlinear dynamical systems, *Comput. Methods Appl. Mech. Engrg* **386**, art. 114050.

V. Simoncini (2016), Computational methods for linear matrix equations, *SIAM Rev.* **58**, 377–441.

R. Singh, W. I. T. Uy and B. Peherstorfer (2023), Lookahead data-gathering strategies for online adaptive model reduction of transport-dominated problems, *Chaos* **33**, art. 113112.

L. Sirovich (1987), Turbulence and the dynamics of coherent structures 1: Coherent structures, *Q. Appl. Math.* **45**, 561–571.

V. Sitzmann, J. Martel, A. Bergman, D. Lindell and G. Wetzstein (2020), Implicit neural representations with periodic activation functions, in *Advances in Neural Information Processing Systems 33* (H. Larochelle *et al.*, eds), Curran Associates, pp. 7462–7473.

K. Smetana and A. T. Patera (2016), Optimal local approximation spaces for component-based static condensation procedures, *SIAM J. Sci. Comput.* **38**, A3318–A3356.

K. Smetana and T. Taddei (2023), Localized model reduction for nonlinear elliptic partial differential equations: Localized training, partition of unity, and adaptive enrichment, *SIAM J. Sci. Comput.* **45**, A1300–A1331.

T. Taddei (2020), A registration method for model order reduction: Data compression and geometry reduction, *SIAM J. Sci. Comput.* **42**, A997–A1027.

T. Taddei and L. Zhang (2021), Space–time registration-based model reduction of parameterized one-dimensional hyperbolic PDEs, *ESAIM Math. Model. Numer. Anal.* **55**, 99–130.

T. Taddei, S. Perotto and A. Quarteroni (2015), Reduced basis techniques for nonlinear conservation laws, *ESAIM Math. Model. Numer. Anal.* **49**, 787–814.
T. Taddei, I. Tifouti and N. Barral (2025), Local registration-based model reduction of parameterized PDEs with spatio-parameter adaptivity, in *AIAA SCITECH 2025 Forum*. Available at doi:10.2514/6.2025-1571.
M. Telgarsky (2016), benefits of depth in neural networks, in *29th Annual Conference on Learning Theory* (V. Feldman *et al.*, eds), Vol. 49 of Proceedings of Machine Learning Research, PMLR, pp. 1517–1539.
M. Tomasetto, J. P. Williams, F. Braghin, A. Manzoni and J. N. Kutz (2025), Reduced order modeling with shallow recurrent decoder networks, *Nat. Commun.* **16**, art. 10260.
D. Torlo (2025), Model reduction for advection dominated hyperbolic problems in an ALE framework: Offline, online phases and error estimator, in *Numerical Mathematics and Advanced Applications (ENUMATH 2023)* (A. Sequeira *et al.*, eds), Vol. 2, pp. 409–419.
L. N. Trefethen (2019), *Approximation Theory and Approximation Practice*, SIAM.
B. Unger and S. Gugercin (2019), Kolmogorov n-widths for linear dynamical systems, *Adv. Comput. Math.* **45**, 2273–2286.
K. Urban and A. T. Patera (2012), A new error bound for reduced basis approximation of parabolic partial differential equations, *C.R. Math. Acad. Sci. Paris* **350**, 203–207.
W. I. T. Uy, C. R. Wentland, C. Huang and B. Peherstorfer (2024), Reduced models with nonlinear approximations of latent dynamics for model premixed flame problems, in *Reduction, Approximation, Machine Learning, Surrogates, Emulators and Simulators: RAMSES* (G. Rozza *et al.*, eds), Springer, pp. 241–259.
T. Van Gastelen, W. Edeling and B. Sanderse (2026), Modeling advection-dominated flows with space-local reduced-order models, *Comput. Fluids* **305**, art. 106911.
V. Vapnik (1991), Principles of risk minimization for learning theory, in *Advances in Neural Information Processing Systems 4* (J. Moody *et al.*, eds), Morgan Kaufmann.
K. Veroy and A. Patera (2005), Certified real-time solution of the parametrized steady incompressible Navier–Stokes equations: Rigorous reduced-basis *a posteriori* error bounds, *Internat. J. Numer. Methods Fluids* **47**, 773–788.
K. Washabaugh, D. Amsallem, M. Zahr and C. Farhat (2012), Nonlinear model reduction for CFD problems using local reduced-order bases, in *42nd AIAA Fluid Dynamics Conference and Exhibit*. Available at doi:10.2514/6.2012-2686.
P. Weder, P. Schwerdtner and B. Peherstorfer (2025), Nonlinear model reduction with Neural Galerkin schemes on quadratic manifolds, *J. Comput. Phys.* **539**, art. 114249.
G. Welper (2017), Interpolation of functions with parameter dependent jumps by transformed snapshots, *SIAM J. Sci. Comput.* **39**, A1225–A1250.
G. Welper (2020), Transformed snapshot interpolation with high resolution transforms, *SIAM J. Sci. Comput.* **42**, A2037–A2061.
Y. Wen, E. Vanden-Eijnden and B. Peherstorfer (2024), Coupling parameter and particle dynamics for adaptive sampling in neural Galerkin schemes, *Phys. D* **462**, art. 134129.
C. R. Wentland, K. Duraisamy and C. Huang (2023), Scalable projection-based reduced-order models for large multiscale fluid systems, *AIAA J.* **61**, 4499–4523.
B. Wieland (2015), Implicit partitioning methods for unknown parameter sets, *Adv. Comput. Math.* **41**, 1159–1186.
P. Wittmuess, C. Tarin, A. Keck, E. Arnold and O. Sawodny (2016), Parametric model order reduction via balanced truncation with Taylor series representation, *IEEE Trans. Automat. Control* **61**, 3438–3451.

C. Wu, M. Zhu, Q. Tan, Y. Kartha and L. Lu (2023), A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks, *Comput. Methods Appl. Mech. Engrg* **403**, art. 115671.

J. Xu and K. Duraisamy (2020), Multi-level convolutional autoencoder networks for parametric prediction of spatio-temporal dynamics, *Comput. Methods Appl. Mech. Engrg* **372**, art. 113379.

S. Yıldız, P. Goyal, T. Bendokat and P. Benner (2024), Data-driven identification of quadratic representations for nonlinear Hamiltonian systems using weakly symplectic liftings, *J. Mach. Learn. Model. Comput.* **5**, 45–71.

M. J. Zahr and C. Farhat (2015), Progressive construction of a parametric reduced-order model for PDE-constrained optimization, *Internat. J. Numer. Methods Engrg* **102**, 1111–1135.

H. Zhang, Y. Chen, E. Vanden-Eijnden and B. Peherstorfer (2024), Sequential-in-time training of nonlinear parametrizations for solving time-dependent partial differential equations. Available at arXiv:2404.01145. To appear in *SIAM Rev*.

T. Zhao, C. Sun, A. Cohen, J. Stokes and S. Veerapaneni (2024), Quantum-inspired variational algorithms for partial differential equations: Application to financial derivative pricing, *Quant. Finance* **24**, 1–11.

R. Zimmermann and K. Willcox (2016), An accelerated greedy missing point estimation procedure, *SIAM J. Sci. Comput.* **38**, A2827–A2850.

R. Zimmermann, B. Peherstorfer and K. Willcox (2018), Geometric subspace updates with applications to online adaptive nonlinear model reduction, *SIAM J. Matrix Anal. Appl.* **39**, 234–261.

V. Zucatti and M. J. Zahr (2024), An adaptive, training-free reduced-order model for convection-dominated problems based on hybrid snapshots, *Internat. J. Numer. Methods Fluids* **96**, 189–208.

A. Zuyev, L. Feng and P. Benner (2024), Estimates of the Kolmogorov n-width for nonlinear transformations with application to distributed-parameter control systems, *IEEE Control Systems Lett.* **8**, 1877–1882.