Learning structured and stable reduced models from data with operator inference

by

Nihar Sawant

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

May, 2023

Dr. Benjamin Peherstorfer

Dedication

For my mother and father, who taught me to believe in myself, and maintain a positive outlook no matter what I faced.

Acknowledgments

First and foremost, I would like to thank my advisor Dr. Benjamin Peherstorfer, who played a key role in the project's inception, as well as its successful completion. Benjamin's drive for research, innovative ideas, and constant support, were essential in advancing my work. I would like to thank my thesis reading committee members Dr. Michael Overton and Dr. Georg Stadler, whose insightful comments and questions were of great help. I would like to thank Dr. Boris Kramer for his guidance in key parts of the project, which lead to our publication. I would like to thank Pranjal, whose frequent counsel helped me navigate many difficult aspects of my PhD. I would like to thank my sister Pooja Sawant, who believed in me even more than I did. Finally, I would like to thank my parents, Seema and Suhas Sawant, who helped me develop an inquisitive mind at an early age, and gave me emotional support whenever I needed it.

Abstract

Operator inference learns low-dimensional dynamical-system models with polynomial nonlinear terms from trajectories of high-dimensional physical systems (non-intrusive model reduction). This work focuses on the large class of physical systems that can be well described by models with quadratic and cubic nonlinear terms and proposes a regularizer for operator inference that induces a stability bias onto learned models. The proposed regularizer is physics informed in the sense that it penalizes higher-order terms with large norms and so explicitly leverages the polynomial model form that is given by the underlying physics. This means that the proposed approach judiciously learns from data and physical insights combined, rather than from either data or physics alone. A formulation of operator inference is proposed that enforces model constraints for preserving structure such as symmetry and definiteness in linear terms. Additionally, for a system of nonlinear conservation laws, we enforce model constraints that preserve the entropy stability of the dynamical system. Numerical results demonstrate that models learned with operator inference and the proposed regularizer and structure preservation are accurate and stable even in cases where using no regularization and Tikhonov regularization leads to models that are unstable.

Table of contents

List of figures

Chapter 1

Introduction

1.1 Motivation

As simulating high-dimensional dynamical systems is computationally expensive, there is a need for model reduction to derive low-dimensional reduced models that are quicker to simulate [Ant05, RHP07, QRM11, QR14, BGW15, HRS16, ABG21, Peh22]. Classical projection-based model reduction generates reduced models via projecting the full model operators onto the reduced space. Thus, classical model reduction is intrusive in nature and cannot be used when the full model is a gray box model, where full model operators cannot be accessed and we only have access to data and partial structural information. This motivates non-intrusive, data-driven model reduction that learns reduced model operators from trajectories of the full model. Relying on data alone is often insufficient to learn accurate, interpretable, and robust models. For example, for a dynamical system with a dissipation term, the operator corresponding to the dissipation term is symmetric positive definite, and not preserving the structure of this operator can lead to an unstable model. There also typically is inherent structure to the full model operators, resulting from the physics of the underlying model, which often cannot be learned from data alone. This motivates using a combination of data and physical insights to learn the reduced model, which is also the route we go in this thesis.

1.2 Thesis objectives

The thesis objectives are as follows:

1. Combine data and physical insights to learn reduced models in a non-intrusive manner.
2. Increase robustness of non-intrusive model reduction with operator inference via physics-informed regularization.
3. Improve stability of non-intrusive model reduction with operator inference via structure preservation.

4. Preserve entropy conservation and entropy stability for non-intrusive model reduction with operator inference.

Motivated by the Lyapunov stability analysis of dynamical systems, we design a physics-informed regularizer that penalizes the norm of the non-linear operators. To preserve the underlying structure of the dynamical systems, we enforce constraints on the linear operators of the dynamical system and solve a constrained optimization problem to learn the reduced model. To preserve the entropy stability and stricter entropy conservation properties of a system of nonlinear conservation laws, we also enforce constraints on the operators, and solve a constrained optimization problem when learning the reduced model.

1.3 Thesis outline

In chapter 2, we briefly review intrusive and non-intrusive model reduction techniques in time and frequency domain and operator inference. In chapter 3, we discuss the motivation behind the conducted research, propose physics-informed regularization for operator inference, and demonstrate the proposed regularizer with numerical experiments. In chapter 4, we discuss the structure preservation for operator inference. In chapter 5, we discuss non-intrusive entropy conservation and entropy stable model reduction using operator inference and show numerical results. In chapter 6, we conclude with a brief overview of our findings.

Chapter 2

Preliminaries

In this chapter, we review selected literature on intrusive and non-intrusive model reduction in time and frequency domain. Section 2.1 briefly describes the details of the high-dimensional full models that we consider in the following. Section 2.2 reviews selected techniques of intrusive and non-intrusive model reduction in time domain, and Section 2.3 reviews a non-intrusive model reduction method in frequency domain. Section 2.4 reviews operator inference, which is the non-intrusive model reduction approach that we will build on in the following. The concept of re-projection is discussed in Section 2.5. Operator inference with re-projection aims to recover intrusive model operators without accessing full-model operators. We refer to the surveys and textbooks [Ant05, RHP07, QRM11, QR14, BGW15, HRS16, ABG21, Peh22] for more details on model reduction.

2.1 High-dimensional numerical models ("full models")

Consider a high-dimensional dynamical system of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t;\mu) = \boldsymbol{f}(\boldsymbol{x}(t;\mu), \boldsymbol{u}(t;\mu);\mu)\,, \tag{2.1}$$

with the state $\boldsymbol{x}(t;\mu) \in \mathbb{R}^N$ at time $t \in [0,T]$ and for parameter $\mu \in \mathcal{D}$ in a parameter domain $\mathcal{D}$. The system can be excited with the input $\boldsymbol{u}(t;\mu) \in \mathbb{R}^p$. The dynamics are encoded by the potentially nonlinear function $\boldsymbol{f} : \mathbb{R}^N \times \mathbb{R}^p \times \mathcal{D} \to \mathbb{R}^N$. This is the high-dimensional full model, and the goal of model reduction is to learn a reduced model of dimension $n \ll N$ that is within reasonable accuracy of the full model.

In many cases, the nonlinear function $\boldsymbol{f}$ has structure. In this work, we focus on the cases where $\boldsymbol{f}$ has polynomial structure so that we can write

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t;\mu) = \boldsymbol{A}(\mu)\boldsymbol{x}(t;\mu) + \boldsymbol{B}(\mu)\boldsymbol{u}(t;\mu) + \sum_{r=2}^{R} \boldsymbol{F}_r(\mu)\boldsymbol{x}^r(t;\mu)\,, \tag{2.2}$$

with linear operator $\boldsymbol{A}(\mu) \in \mathbb{R}^{N\times N}$, input operator $\boldsymbol{B}(\mu) \in \mathbb{R}^{N\times p}$, and non-linear operators $\boldsymbol{F}_r(\mu)$ for $r = 2,\ldots,R$. For example, we have $\boldsymbol{F}_2(\mu) \in \mathbb{R}^{N\times N(N+1)/2}$ for $r = 2$, and $\boldsymbol{F}_3(\mu) \in \mathbb{R}^{N\times N(N+1)(N+2)/6}$ for $r = 3$. For a quadratic system, the vector $\boldsymbol{x}^2(t;\mu) \in \mathbb{R}^{N(N+1)/2}$ is defined as

$$\boldsymbol{x}^2(t;\mu) = [\boldsymbol{x}_2^{(1)}(t;\mu)^T, \ldots, \boldsymbol{x}_2^{(N)}(t;\mu)^T]^T \tag{2.3}$$

where $\boldsymbol{x}_2^{(i)}(t;\mu) = x_i(t;\mu)[x_1(t;\mu),\ldots,x_i(t;\mu)]^T$ for $i = 1,\ldots,N$. The vector $\boldsymbol{x}^2(t;\mu)$ contains all pairwise products of components of the state vector $\boldsymbol{x}(t;\mu)$ up to duplicates; see, e.g.,

[PW16b]. For a cubic system, the vector $\boldsymbol{x}^3(t;\mu) \in \mathbb{R}^{N(N+1)(N+2)/6}$ is defined as

$$\boldsymbol{x}^3(t;\mu) = [\boldsymbol{x}_3^{(1)}(t;\mu)^T, \ldots, \boldsymbol{x}_3^{(N)}(t;\mu)^T]^T \tag{2.4}$$

where $\boldsymbol{x}_3^{(i)}(t;\mu) = [x_3^{(i,i)}(t;\mu)^T, \ldots, x_3^{(i,N)}(t;\mu)^T]^T$ for $i = 1, \ldots, N$, with

$$x_3^{(i,j)}(t;\mu) = x_i(t;\mu)x_j(t;\mu)[x_i(t;\mu), \ldots, x_j(t;\mu)]^T \quad \text{for } j = i, \ldots, N.$$

A special case is given by linear time-invariant and parameter-free systems so that (2.2) becomes

$$\boldsymbol{E}\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \tag{2.5}$$

where often an output equation is given as well as

$$\boldsymbol{y}(t) = \boldsymbol{C}\boldsymbol{x}(t)\,.$$

The output operator is $\boldsymbol{C} \in \mathbb{R}^{q\times N}$ and output at time t is $\boldsymbol{y}(t) \in \mathbb{R}^q$. Notice that there is a matrix $\boldsymbol{E} \in \mathbb{R}^{N\times N}$ that we assume to have full rank in the following. A system such as (2.5) can also be represented in the frequency domain by taking the Laplace transform. One obtains

$$\boldsymbol{Y}(s) = \boldsymbol{H}(s)\boldsymbol{U}(s), \tag{2.6}$$

with the transfer function

$$\boldsymbol{H}(s) = \boldsymbol{C}(s\boldsymbol{E} - \boldsymbol{A})^{-1}\boldsymbol{B}, \tag{2.7}$$

at frequency s and the input $\boldsymbol{U}(s)$ in the frequency domain.

2.2 Reducing models formulated in the time domain

We now describe model reduction techniques that apply to models that are formulated in the time domain. We start in Section 2.2.1 with intrusive methods. Intrusive means that the full-model operators have to be available either in assembled form or via a routine that provides the action of the operator on a function. In Section 2.2.2, we discuss non-intrusive model reduction techniques.

2.2.1 Intrusive projection-based model reduction methods

Traditionally, model reduction consists of two phases. In the offline phase, a reduced model is constructed. This is a one-time computationally expensive phase. In the online (test/deployment/evaluation) phase, the reduced model is used to provide predictions of the full-model solutions; often with greatly reduced costs. We note, however, that there is a range of methods that break with the classical offline/online splitting, e.g., [PW15a, AZW15, Car15, PW15b, PW16a, ZPW18, Peh20a, CKMP20, HPR22].

We focus on snapshot-based model reduction techniques. For each parameter $\mu \in \{\mu_1, \dots, \mu_M\}$ in a set of $M \in \mathbb{N}$ parameters, consider $M_b \in \mathbb{N}$ input trajectories

$$\boldsymbol{U}^{\mathrm{b}}_1(\mu), \dots, \boldsymbol{U}^{\mathrm{b}}_{M_b}(\mu),$$

and initial conditions $\boldsymbol{x}^{\mathrm{b}}_{1,0}(\mu), \dots, \boldsymbol{x}^{\mathrm{b}}_{M_b,0}(\mu)$. The full model is simulated at these input trajectories and initial conditions to generate the corresponding state trajectories

$$\boldsymbol{X}^{\mathrm{b}}_1(\mu), \dots, \boldsymbol{X}^{\mathrm{b}}_{M_b}(\mu),$$

where

$$\boldsymbol{X}^{\mathrm{b}}_j(\mu) = [\boldsymbol{x}^{\mathrm{b}}_{j,1}(\mu), \cdots, \boldsymbol{x}^{\mathrm{b}}_{j,K}(\mu)] \in \mathbb{R}^{N \times K}, \quad j = 1, \dots, M_b,$$

with time steps $0 = t_0 < t_1 < \cdots < t_K = T$. Thus, state $\boldsymbol{x}^{\mathrm{b}}_{j,i}(\mu)$ corresponds to the j-th trajectory and time step i. The state trajectories for all parameters are concatenated into the snapshot matrix

$$\boldsymbol{X}^{\mathrm{b}} = [\boldsymbol{X}^{\mathrm{b}}_1(\mu_1), \ldots, \boldsymbol{X}^{\mathrm{b}}_{M_b}(\mu_1), \ldots, \boldsymbol{X}^{\mathrm{b}}_1(\mu_M), \ldots, \boldsymbol{X}^{\mathrm{b}}_{M_b}(\mu_M)] \in \mathbb{R}^{N \times KMM_b}. \tag{2.8}$$

Intrusive model reduction based on snapshots proceeds in three steps: The **first step** is snapshot collection to obtain the snapshot matrix (2.8).

The **second step** is constructing a reduced basis matrix $\boldsymbol{V}_n \in \mathbb{R}^{N \times n}$. The columns of the basis matrix $\boldsymbol{V}_n$ span a reduced space $\mathcal{V}_n$. There are many techniques for constructing bases of reduced spaces. We will focus on proper orthogonal decomposition (POD) [Sir87, BHL93, RP03]. For a POD basis of dimension $n \ll N$, the POD space $\mathcal{V}_n$ is the space that minimizes the quantity

$$\sum_{i=1}^{M} \sum_{j=1}^{M_b} \sum_{k=1}^{K} \inf_{\boldsymbol{v}_n \in \mathcal{V}_n} \left\| \boldsymbol{x}^{\mathrm{b}}_{j,k}(\mu_j) - \boldsymbol{v}_n \right\|.$$

The solution for this minimization problem is constructed from the singular value decomposition (SVD) of the snapshot matrix $\boldsymbol{X}^{\mathrm{b}}$. We do not scale and center the snapshots before applying POD. The reduced basis vectors are the first n left singular vectors

$$\boldsymbol{V}_n = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n] \in \mathbb{R}^{N \times n}, \tag{2.9}$$

corresponding to the n largest singular values of $\boldsymbol{X}^{\mathrm{b}}$. The basis matrix $\boldsymbol{V}_n$ is orthogonal in the following. We focus on the orthogonality with respect to the Euclidean distance so that $\boldsymbol{V}_n^T \boldsymbol{V}_n = \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix. There is a wide range of heuristics to select the dimension n. One is observing the decay of the singular values and truncating when 95% or 99% of the energy are retained in the POD modes; see [BGW15].

The **third step** is constructing the reduced model. We focus here on Galerkin projection,

but note that Petrov-Galerkin projection is also often used. The reduced model constructed via Galerkin projection of the full model is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\tilde{\boldsymbol{x}}(t;\mu) = \boldsymbol{V}_n^T \boldsymbol{f}(\boldsymbol{V}_n \tilde{\boldsymbol{x}}(t;\mu); \boldsymbol{u}(t;\mu))\,, \tag{2.10}$$

where the reduced state $\tilde{\boldsymbol{x}}(t;\mu) \in \mathbb{R}^n$ is of dimension n and approximates the full-model state $\boldsymbol{x}(t;\mu)$ as $\boldsymbol{x}(t;\mu) \approx \boldsymbol{V}_n \tilde{\boldsymbol{x}}(t;\mu)$. Notice that $\boldsymbol{f}$ is the nonlinear function of the generic full-model form given in (2.1). Thus, to evaluate the right-hand side of the Galerkin reduced model given in (2.10), one has to take the reduced state $\tilde{\boldsymbol{x}}(t;\mu)$ and represent it in the high-dimensional space $\mathcal{V}$ corresponding to the full model and then evaluate the nonlinear function $\boldsymbol{f}$ at all N components, only to project it back onto the reduced space $\mathcal{V}_n$. This means that evaluating $\tilde{\boldsymbol{x}} \mapsto \boldsymbol{V}_n^T \boldsymbol{f}(\boldsymbol{V}_n \tilde{\boldsymbol{x}})$ incurs computational costs that are at least as high as evaluating $\boldsymbol{f}$ in the full model and thus typically no speedups can be achieved. This is referred to as the "lifting bottleneck" in the model reduction. Various solutions have been proposed. Especially sampling-based techniques have been very successful, which are based on gappy POD [ES95, AWWB08, CBMF11, CFCA13] and empirical interpolation [BNMP04, MM13, MMPY15, CS10, PBWB14, DG16, PDG20, SBK15, MBK17]. In this work, we go a different route. We exploit the polynomial structure of $\boldsymbol{f}$ as given in (2.2), which allows pre-computing reduced operators in the offline phase, which then can be used online with costs that scale independently of the dimension N of the full model [Gu11, BB15, KW19]. This gives us a reduced model of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\tilde{\boldsymbol{x}}(t;\mu) = \boldsymbol{V}_n^T \boldsymbol{A}(\mu) \boldsymbol{V}_n \tilde{\boldsymbol{x}}(t;\mu) + \boldsymbol{V}_n^T \boldsymbol{B}(\mu) \boldsymbol{u}(t;\mu) + \sum_{r=2}^{R} \boldsymbol{V}_n^T \boldsymbol{F}_r(\mu)(\boldsymbol{V}_n \otimes \cdots \otimes \boldsymbol{V}_n)\tilde{\boldsymbol{x}}^r(t;\mu)\,,$$

where the $\boldsymbol{V}_n^T \boldsymbol{F}_r(\mu)(\boldsymbol{V}_n \otimes \cdots \otimes \boldsymbol{V}_n)$ is denoting a projection that is described in detail in, e.g., [KW19, Peh20b]. Thus, for each parameter $\mu \in \{\mu_1, \ , \ldots, \ , \mu_M\}$, one obtains a reduced

model

$$\frac{\mathrm{d}}{\mathrm{d}t}\tilde{\boldsymbol{x}}(t,\mu) = \tilde{\boldsymbol{A}}(\mu)\tilde{\boldsymbol{x}}(t;\mu) + \tilde{\boldsymbol{B}}(\mu)\boldsymbol{u}(t;\mu) + \sum_{r=2}^{R} \tilde{\boldsymbol{F}}_r(\mu)\tilde{\boldsymbol{x}}^r(t;\mu)\,, \tag{2.11}$$

where

$$\tilde{\boldsymbol{A}}(\mu) = \boldsymbol{V}_n^T\boldsymbol{A}(\mu)\boldsymbol{V}_n\,, \quad \tilde{\boldsymbol{B}} = \boldsymbol{V}_n^T\boldsymbol{B}(\mu)\,, \quad \tilde{\boldsymbol{F}}_r = \boldsymbol{V}_n^T\boldsymbol{F}_r(\mu)\underbrace{(\boldsymbol{V}_n \otimes \cdots \otimes \boldsymbol{V}_n)}_{\text{r times}},\ r = 2,\ldots,R. \tag{2.12}$$

The reduced model (2.11) can then be used in the online phase to provide predictions at new initial conditions that have not been used in the training set. Because the reduced system has states of dimension $n \ll N$, it often can be simulated with lower computational costs than the full model with states of dimension N. If one wants to use the reduced model to predict at new parameter $\mu \in \mathcal{D} \setminus \{\mu_1,\ldots,\mu_M\}$ that are outside of the set of training parameters $\{\mu_1,\ldots,\mu_M\}$, then one can interpolate the reduced operators (2.12) at a new parameter $\mu \in \mathcal{D}$. For a test parameter μ, such that

$$\mu_j < \mu < \mu_{j+1},\ j \in \{1,\ldots,M-1\},$$

the reduced model operators $\hat{\boldsymbol{A}}(\mu_i), \hat{\boldsymbol{B}}(\mu_i), \hat{\boldsymbol{F}}_2(\mu_i),\ldots,\hat{\boldsymbol{F}}_R(\mu_i),\ i = i,\ldots,M$ are interpolated to compute reduced model operators $\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu), \hat{\boldsymbol{F}}_2(\mu),\ldots,\hat{\boldsymbol{F}}_R(\mu)$ in the following manner:

$$\begin{aligned}
\hat{\boldsymbol{A}}(\mu) &= \mathcal{I}\left(\mu; \hat{\boldsymbol{A}}(\mu_1),\ldots,\hat{\boldsymbol{A}}(\mu_M)\right),\\
\hat{\boldsymbol{B}}(\mu) &= \mathcal{I}\left(\mu; \hat{\boldsymbol{B}}(\mu_1),\ldots,\hat{\boldsymbol{B}}(\mu_M)\right),\\
\hat{\boldsymbol{F}}_r(\mu) &= \mathcal{I}\left(\mu; \hat{\boldsymbol{F}}_r(\mu_1),\ldots,\hat{\boldsymbol{F}}_r(\mu_M)\right), \quad r = 2,\ldots,R,
\end{aligned}$$

where $\mathcal{I}\left(\mu; \hat{\boldsymbol{A}}(\mu_1), \ldots, \hat{\boldsymbol{A}}(\mu_M)\right)$ is the linear interpolation around μ defined as,

$$\mathcal{I}\left(\mu; \hat{\boldsymbol{A}}(\mu_1), \ldots, \hat{\boldsymbol{A}}(\mu_M)\right) = \hat{\boldsymbol{A}}(\mu_j) + (\mu - \mu_j)\frac{\hat{\boldsymbol{A}}(\mu_{j+1}) - \hat{\boldsymbol{A}}(\mu_i)}{\mu_{j+1} - \mu_j}.$$

Linear interpolation is done in a similar manner for $\hat{\boldsymbol{B}}(\mu), \hat{\boldsymbol{F}}_2(\mu), \ldots, \hat{\boldsymbol{F}}_R(\mu)$.

2.2.2 Non-intrusive methods in the time domain

Classical model reduction requires access to the high-dimensional operators, thus it is intrusive in nature as described in Section 2.2.1. This can be seen in the projection step shown in (2.12), where a multiplication with the full-model operator is performed. Non-intrusive model reduction methods learn reduced models from a gray box model without requiring access to the full model operators by either circumventing the projection step (2.12) or replacing it with a non-intrusive calculation.

The are many different methods in literature for non-intrusive model reduction. There is dynamic mode decomposition (DMD) [Sch10, RMB+09, TRL+14, KBBP16] that best-fits linear operators to state trajectories in L_2 norm. Methods based on Koopman operators have been developed to extend DMD to nonlinear systems [Mez05, WKR15, BBPK16]. There are also sparse identification methods such as SINDy [BPK16] and the works [SCHO13, Sch17, TW17]. Motivated by frequency-domain methods that we will be survey below, there is a range of time-domain Loewner approaches such as [PGW17, KGA21].

We build upon operator inference [PW16b] that infers reduced models with polynomial nonlinear terms from snapshots data. After generating the reduced basis $\boldsymbol{V} \in \mathbb{R}^{N\times n}$, operator inference projects the state trajectories onto the reduced basis to generate low-dimensional projected trajectories. Reduced model operators $\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu)$ and $\hat{\boldsymbol{F}}_2(\mu), \ldots, \hat{\boldsymbol{F}}_R(\mu)$ are then fitted to these projected trajectories by solving a least squares problem. We discuss operator inference in detail in section 2.4.

2.3 Non-intrusive model reduction in the frequency domain

We focus on briefly reviewing the Loewner framework [AA86, MA07, ABG21] as a method for non-intrusive model reduction in the frequency domain. Note that there is a range of other non-intrusive model reduction methods that can be applied in the frequency domain such as the AAA algorithm [NST18], eigensystem realization algorithm [Vib95, KG16, Qin06, Rey12], and vector fitting [GS99, DGB15].

Loewner framework was developed for linear time-invariant systems but since than has been extended to parameterized [IA14], switched [GPA18], structured [SUBG18], delayed [SU16], bilinear [AGI16], quadratic [GKA21], quadratic bilinear [GA18], polynomial [BG21], and parameter-varying [GPA21] systems as well as to learning from time-domain data [PGW17, KGA21] and noisy data [LIA10, EI22, DP22]. We describe the Loewner framework in the setting of linear time-invariant systems. Consider the LTI system given in (2.5) with its frequency-domain representation in (2.6). The aim is to construct a reduced model of dimension $n \ll N$ of the form

$$\begin{aligned} \hat{\boldsymbol{E}}\frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{x}}(t) &= \hat{\boldsymbol{A}}\hat{\boldsymbol{x}}(t) + \hat{\boldsymbol{B}}\boldsymbol{u}(t), \\ \hat{\boldsymbol{y}}(t) &= \hat{\boldsymbol{C}}\hat{\boldsymbol{x}}(t), \end{aligned} \tag{2.13}$$

with a transfer function

$$\hat{\boldsymbol{H}}(s) = \hat{\boldsymbol{C}}(s\hat{\boldsymbol{E}} - \hat{\boldsymbol{A}})^{-1}\hat{\boldsymbol{B}}$$

that interpolates the full-model transfer function $\boldsymbol{H}$ at interpolation points $s_1, \ldots, s_{2n}$. The interpolation points are partitioned into two sets

$$[s_1, \ldots, s_{2n}] = [\lambda_1, \ldots, \lambda_n] \cup [\pi_1, \ldots, \pi_n].$$

The values of the full-model transfer function $\boldsymbol{H}$ evaluated at these interpolation points are used to construct the Loewner matrix

$$\mathbb{L} = \begin{bmatrix} \frac{\boldsymbol{H}(\lambda_1)-\boldsymbol{H}(\pi_1)}{\lambda_1-\pi_1} & \cdots & \frac{\boldsymbol{H}(\lambda_1)-\boldsymbol{H}(\pi_n)}{\lambda_1-\pi_n} \\ \vdots & \ddots & \vdots \\ \frac{\boldsymbol{H}(\lambda_n)-\boldsymbol{H}(\pi_1)}{\lambda_n-\pi_1} & \cdots & \frac{\boldsymbol{H}(\lambda_n)-\boldsymbol{H}(\pi_n)}{\lambda_n-\pi_n} \end{bmatrix},$$

and the shifted Loewner matrix

$$\mathbb{L}_s = \begin{bmatrix} \frac{\lambda_1\boldsymbol{H}(\lambda_1)-\pi_1\boldsymbol{H}(\pi_1)}{\lambda_1-\pi_1} & \cdots & \frac{\lambda_1\boldsymbol{H}(\lambda_1)-\pi_n\boldsymbol{H}(\pi_n)}{\lambda_1-\pi_n} \\ \vdots & \ddots & \vdots \\ \frac{\lambda_n\boldsymbol{H}(\lambda_n)-\pi_1\boldsymbol{H}(\pi_1)}{\lambda_n-\pi_1} & \cdots & \frac{\lambda_n\boldsymbol{H}(\lambda_n)-\pi_n\boldsymbol{H}(\pi_n)}{\lambda_n-\pi_n} \end{bmatrix}.$$

The Loewner reduced model matrices of the reduced model (2.13) are

$$\tilde{\boldsymbol{E}} = -\mathbb{L},\ \hat{\boldsymbol{A}} = -\mathbb{L}_s,\ \hat{\boldsymbol{B}} = [\boldsymbol{H}(\lambda_1), \ldots, \boldsymbol{H}(\lambda_n)]^T,\ \hat{\boldsymbol{C}} = [\boldsymbol{H}(\pi_1), \ldots, \boldsymbol{H}(\pi_n)], \tag{2.14}$$

and the corresponding transfer function $\hat{\boldsymbol{H}}$ interpolates the full model transfer function $\boldsymbol{H}(s)$ at $s_1, \ldots, s_{2n}$. The system given by (2.14) can be compressed via the SVD [ABG21]. To summarize, the data that is used to construct a reduced model consists of pairs of frequencies and transfer-function values.

2.4 Operator inference

Operator inference [PW16b] learns reduced models from data and physical insights from the model. A reduced basis matrix $\boldsymbol{V} \in \mathbb{R}^{N\times n}$ is constructed in same way as in classical intrusive model reduction, as shown in Section 2.2. For each training parameter $\mu \in \{\mu_1, \ldots, \mu_M\}$, consider M_t training input trajectories $\boldsymbol{U}_1(\mu), \ldots, \boldsymbol{U}_{M_t}(\mu)$ with

initial conditions $\boldsymbol{x}_{1,0}(\mu), \ldots, \boldsymbol{x}_{M_t,0}(\mu)$ and the corresponding training state trajectories $\boldsymbol{X}_1(\mu), \ldots, \boldsymbol{X}_{M_t}(\mu)$.

2.4.1 Basic Operator Inference

Operator Inference proceeds in three steps: In Step 1, the training trajectories are projected onto the reduced space via

$$\bar{\boldsymbol{X}}_i(\mu) = \boldsymbol{V}^T \boldsymbol{X}_i(\mu)\,, \qquad i = 1, \ldots, M_t\,.$$

The corresponding projected trajectories $\bar{\boldsymbol{X}}_1(\mu), \ldots, \bar{\boldsymbol{X}}_{M_t}$ have columns with dimension n of the reduced space $\mathcal{V}$, that is spanned by the columns of the basis matrix $\boldsymbol{V}$.

In Step 2, the operators $\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu)$ and $\hat{\boldsymbol{F}}_r(\mu), r = 2, \ldots, R$ are fitted via least-squares regression to the projected training trajectories,

$$\min_{\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu), \hat{\boldsymbol{F}}_2(\mu), \ldots, \hat{\boldsymbol{F}}_R(\mu)} J(\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu), \hat{\boldsymbol{F}}_2(\mu), \ldots, \hat{\boldsymbol{F}}_R(\mu))$$

with the objective function

$$\begin{aligned} &J(\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu), \hat{\boldsymbol{F}}_2(\mu), \ldots, \hat{\boldsymbol{F}}_R(\mu)) \\ &= \sum_{i=1}^{M_t} \sum_{k=1}^{K} \left\| \bar{\boldsymbol{x}}'_{i,k}(\mu) - \hat{\boldsymbol{A}}(\mu)\bar{\boldsymbol{x}}_{i,k}(\mu) - \hat{\boldsymbol{B}}(\mu)\boldsymbol{u}_{i,k}(\mu) - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r(\mu)\bar{\boldsymbol{x}}^r_{i,k}(\mu) \right\|_2^2, \end{aligned} \tag{2.15}$$

where $\boldsymbol{u}_{i,k}(\mu)$ is the input at time step k of the ith training trajectory

$$\boldsymbol{U}_i(\mu) = [\boldsymbol{u}_{i,1}(\mu), \ldots, \boldsymbol{u}_{i,K}(\mu)]\,,$$

for $i = 1, \ldots, M_t$. The quantity $\bar{\boldsymbol{x}}'_{i,k} \in \mathbb{R}^n$ denotes a numerical approximation of the time derivative of the projected state at time k of the ith trajectory, such as a first-order finite

difference approximation

$$\bar{\boldsymbol{x}}'_{i,k} = \frac{\bar{\boldsymbol{x}}_{i,k} - \bar{\boldsymbol{x}}_{i,k-1}}{\delta t}\,, \tag{2.16}$$

or a five-point stencil, with time-step size $\delta t > 0$.

In Step 3, the inferred operators $\hat{\boldsymbol{A}}(\mu), \hat{\boldsymbol{B}}(\mu), \hat{\boldsymbol{F}}_2(\mu), \ldots, \hat{\boldsymbol{F}}_R(\mu)$ are then used to assemble a low-dimensional model

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{x}}(t;\mu) = \hat{\boldsymbol{A}}(\mu)\hat{\boldsymbol{x}}_k(t;\mu) + \hat{\boldsymbol{B}}(\mu)\boldsymbol{u}_k(\mu) + \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r(\mu)\hat{\boldsymbol{x}}_k^r(t;\mu). \tag{2.17}$$

For our work, we only consider problems with quadratic and cubic non-linear terms. The learned model (2.17) can then be used to predict at new initial conditions. For predicting at a new parameter, the learned operators can be interpolated as described in Section 2.2.1.

2.4.2 Operator Inference as a building block of Scientific Machine Learning

There is a series of works that build on the basic steps of operator inference outlined above. First, there is work on "lift & learn" [QFW22, Qia21] that first applies a transformation or lifting to the snapshots so that the dynamics can be better approximated with low-order polynomial models. The idea of lifting has a long history; see e.g., [McC76]. In model reduction, lifting to quadratic model form has been used for a long time, starting with the work [Gu11] and follow up works in the model reduction that intersects with the systems and control theory [BB15, BGG18] and snapshot-based, intrusive model reduction with POD [KW19]. Another line of work has extended operator inference to systems without polynomial nonlinear terms by integrating the non-polynomial terms into the objective [BGK+20] and to differential-algebraic equations [KW22]. Additionally, works have built on similar techniques as operator inference for updating reduced models [PW15a, PW16a, KPW17] and context-

aware inference [WP22, WP23, FPN+23, SGP22].

A range of works aim to equip operator-inference models with uncertainty quantification. There is the work [UP21b] that derives probabilistic a posteriori error bounds for operator-inference models of systems with linear dynamics. There is a Bayesian formulation of operator inference [GMW22] that poses the task of learning a reduced model as Bayesian inverse problem with Gaussian prior and likelihood. The resulting posterior distribution characterizes the reduced model operator, thus endowing the reduced model operators with uncertainty. Another line of work in this direction is given by imposing structure on operator-inference models as in [SWK22, SK22] as we discuss later in Section 4.

A major challenge is learning from incomplete, perturbed, and noisy data. Active operator inference [UWWP21] deals with noise in the full model trajectories that can distort the full model dynamics, and affect the reduced model learned via operator inference. The authors introduce a sampling scheme for querying high-dimension systems for data, such that under certain conditions the inferred operators are unbiased estimators of intrusive model operators obtained via projection of the full model operators. The work partial observations [UP21a] looks a state histories to account for the loss of information due to missing components; this is closely related to concepts developed and used in [CHK02, LBCK14, LCV17, PD18, TGDW19, LCV17].

Another important topic is regularization, where especially Tikhonov regularization has been investigated in the context of operator inference in [SKHW20] and [MHW21]. The work [UHP22] develops operator inference in the context of differentiable solvers and takes multiple time steps in the operator-inference objective into account. It is shown that this serves as a regularizer and helps to learn more stable models.

Finally, it is important to note that operator inference is scalable to learning from state with millions of dimensions. There is the work [SKHW20] that applies operator inference to a large-scale application from combustion. Because the learning process of operator inference

relies on standard linear algebra routines, it can be efficiently scaled to high-dimensional state spaces. The work [QFW22] also considers a combustion example and it is shown that an operator-inference model with only about 50 dimensions is sufficient to provide acceptable approximations of a full model with more than 18.5 million degrees of freedom. Similarly, the works [FGMW23, FMW22] scale operator inference to large-scale problems of rotating detonation engines.

2.5 Operator inference with re-projection for recovery guarantees

In this section we review operator inference with re-projection [Peh20b], where the goal is to exactly recover the intrusive reduced model (2.11), without requiring access to the full model operators. Consider the projected state trajectory $\bar{\boldsymbol{X}}(\mu) = \boldsymbol{V}^T\boldsymbol{X}(\mu)$ and the state trajectory $\tilde{\boldsymbol{X}}(\mu)$ generated via iterating the reduced model generated by projecting the full model operators onto the reduced space. The two trajectories are different as there is a non-zero closure error

$$\left\|\bar{\boldsymbol{X}}(\mu) - \tilde{\boldsymbol{X}}(\mu)\right\|_F,$$

thus the non-intrusive reduced model fails to approximate the intrusive reduced model. The goal of re-projection is to generate trajectories $\breve{\boldsymbol{X}}(\mu)$ such that the closure error is zero.

Consider a toy example

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k, \quad k = 0, \ldots, K-1, \tag{2.18}$$

which is a linear full model with $\boldsymbol{B} = \boldsymbol{0}$. The reduced space $\mathcal{V}$ is spanned by the columns of the reduced basis $\boldsymbol{V}$. Consider that the orthogonal component of $\mathcal{V}^\perp$ is spanned by the columns of matrix $\boldsymbol{V}^\perp \in \mathbb{R}^{N\times(N-n)}$, such that $\mathbb{R}^N = \mathcal{V} \oplus \mathcal{V}^\perp$. Define $\boldsymbol{x}_k^{\|} = \boldsymbol{V}^T\boldsymbol{x}_k$, and

$\boldsymbol{x}_k^{\perp} = (\boldsymbol{V}^{\perp})^T \boldsymbol{x}_k$, so that $\boldsymbol{x}_k = \boldsymbol{V}\boldsymbol{x}_k^{\parallel} + \boldsymbol{V}^{\perp}\boldsymbol{x}_k^{\perp}$. Multiplying (2.18) by $\boldsymbol{V}^T$ gives

$$\begin{aligned}\boldsymbol{V}^T\boldsymbol{x}_{k+1} &= \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{x}_k,\\ &= \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}\boldsymbol{x}_k^{\parallel} + \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}^{\perp}\boldsymbol{x}_k^{\perp}.\end{aligned}$$

Similarly, multiplying (2.18) by $(\boldsymbol{V}^{\perp})^T$ gives

$$\begin{aligned}(\boldsymbol{V}^{\perp})^T\boldsymbol{x}_{k+1} &= (\boldsymbol{V}^{\perp})^T\boldsymbol{A}\boldsymbol{x}_k,\\ &= (\boldsymbol{V}^{\perp})^T\boldsymbol{A}\boldsymbol{V}\boldsymbol{x}_k^{\parallel} + (\boldsymbol{V}^{\perp})^T\boldsymbol{A}\boldsymbol{V}^{\perp}\boldsymbol{x}_k^{\perp}.\end{aligned}$$

This leads to the equations

$$\begin{aligned}\boldsymbol{x}_{k+1}^{\parallel} &= \boldsymbol{A}^{\parallel\parallel}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}^{\parallel\perp}\boldsymbol{x}_k^{\perp},\\ \boldsymbol{x}_{k+1}^{\perp} &= \boldsymbol{A}^{\perp\parallel}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}^{\perp\perp}\boldsymbol{x}_k^{\perp},\end{aligned} \tag{2.19}$$

where,

$$\boldsymbol{A}^{\parallel\parallel} = \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V},\ \boldsymbol{A}^{\parallel\perp} = \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}^{\perp},\ \boldsymbol{A}^{\perp\parallel} = (\boldsymbol{V}^{\perp})^T\boldsymbol{A}\boldsymbol{V},\ \boldsymbol{A}^{\perp\perp} = (\boldsymbol{V}^{\perp})^T\boldsymbol{A}\boldsymbol{V}^{\perp}.$$

From (2.19), the projected trajectory can be written as

$$\boldsymbol{V}^T\boldsymbol{x}_{k+1} = \bar{\boldsymbol{x}}_{k+1} = \boldsymbol{x}_{k+1}^{\parallel} = \boldsymbol{A}^{\parallel\parallel}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}^{\parallel\perp}\boldsymbol{x}_k^{\perp}, \quad k = 0, \ldots, K-1.$$

Thus the projected state can be written as

$$\breve{\boldsymbol{x}}_{k+1} = \boldsymbol{x}_{k+1}^{\parallel} = \boldsymbol{A}^{\parallel\parallel}\boldsymbol{x}_k^{\parallel} + \boldsymbol{A}^{\parallel\perp}\sum_{i=0}^{k-1}(\boldsymbol{A}^{\perp\perp})^{k-1-i}\boldsymbol{A}^{\perp\parallel}\boldsymbol{x}_i^{\parallel} + \boldsymbol{A}^{\parallel\perp}(\boldsymbol{A}^{\perp\perp})^k\boldsymbol{x}_0^{\perp}.$$

The projected state $\breve{\boldsymbol{x}}_{k+1}$ at time t_{k+1} depends on the history of the projected states $\boldsymbol{x}_0^{\parallel}, \ldots, \boldsymbol{x}_k^{\parallel}$, instead of only at the last time step $\boldsymbol{x}_k^{\parallel}$. This means that the projected tra-

jectory cannot be well described by a Markovian model.

Re-projection aims to fix this problem. Instead of iterating the full model, and then projecting the generated trajectory onto the reduced space, re-projection iterates the full model for a single time step, projecting the state onto the reduced space and re-projecting it onto the full model space at every time step.

Consider a non-linear full model

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad k = 0, \ldots, K-1, \tag{2.20}$$

where

$$\boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}\boldsymbol{x}_k + \sum_{r=2}^{R} \boldsymbol{F}_r \boldsymbol{x}_k^r + \boldsymbol{B}\boldsymbol{u}_k.$$

Consider an initial condition $\boldsymbol{x}_0 \in \mathcal{V}$, and $\bar{\boldsymbol{x}}_0 = \boldsymbol{V}^T\boldsymbol{x}_0$. Then $\boldsymbol{V}\bar{\boldsymbol{x}}_0 = \boldsymbol{x}_0$, as $\boldsymbol{x}_0 \in \mathcal{V}$. The full model is iterated for a single time step to generate

$$\boldsymbol{x}_{\text{tmp}} = \boldsymbol{f}(\boldsymbol{V}\bar{\boldsymbol{x}}_0, \boldsymbol{u}_0). \tag{2.21}$$

Then $\boldsymbol{x}_{\text{tmp}}$ is projected onto the reduced space to generate $\bar{\boldsymbol{x}}_1 = \boldsymbol{V}^T\boldsymbol{x}_{\text{tmp}}$. Now the full model is iterated a single time step with $\boldsymbol{V}\bar{\boldsymbol{x}}_1$ and $\boldsymbol{u}_1$ to generate $\boldsymbol{f}(\boldsymbol{V}\bar{\boldsymbol{x}}_1, \boldsymbol{u}_1)$ and so on, to generate the re-projected trajectory

$$\bar{\boldsymbol{X}} = [\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_K].$$

The work [Peh20b] offers the following proof that the state trajectory obtained by re-projection is same as the state trajectory generated by iterating the intrusive reduced model.

Proposition 2.1. *For the nonlinear model* (2.20)*, consider initial condition* $\boldsymbol{x}_0 \in \mathcal{V}$*, and inputs* $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_K$*. If for the initial condition* $\boldsymbol{V}^T\boldsymbol{x}_0$*, the intrusive model is iterated to gen-*

erate trajectory $\tilde{\boldsymbol{X}}$*, and re-projection is used to generate the re-projected trajectory* $\bar{\boldsymbol{X}}$*, then* $\tilde{\boldsymbol{X}} = \bar{\boldsymbol{X}}$ *holds.*

Proof. By adding zero columns to the matrices $\boldsymbol{F}_2, \ldots, \boldsymbol{F}_R$ one can get matrices $\boldsymbol{H}_2, \ldots, \boldsymbol{H}_R$ such that

$$\boldsymbol{F}_r \boldsymbol{x}^r = \boldsymbol{H}_r \underbrace{(\boldsymbol{x} \otimes \cdots \otimes \boldsymbol{x})}_{\text{r times}}.$$

Thus, the full model (2.20) can be written as

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \sum_{r=2}^{R} \boldsymbol{H}_r \underbrace{(\boldsymbol{x} \otimes \cdots \otimes \boldsymbol{x})}_{\text{r times}} + \boldsymbol{B}\boldsymbol{u}_k.$$

The first time step of the re-projection iteration (2.21) can be written as

$$\begin{aligned}
\boldsymbol{x}_{\text{tmp}} &= \boldsymbol{A}\boldsymbol{x}_0 + \sum_{r=2}^{R} \boldsymbol{F}_r \boldsymbol{x}_0^r + \boldsymbol{B}\boldsymbol{u}_0, \\
&= \boldsymbol{A}\boldsymbol{x}_0 + \sum_{r=2}^{R} \boldsymbol{H}_r (\boldsymbol{x}_0 \otimes \cdots \otimes \boldsymbol{x}_0) + \boldsymbol{B}\boldsymbol{u}_0, \\
&= \boldsymbol{A}\boldsymbol{V}\tilde{\boldsymbol{x}}_0 + \sum_{r=2}^{R} \boldsymbol{H}_r (\boldsymbol{V}\tilde{\boldsymbol{x}}_0 \otimes \cdots \otimes \boldsymbol{V}\tilde{\boldsymbol{x}}_0) + \boldsymbol{B}\boldsymbol{u}_0, \\
&= \boldsymbol{A}\boldsymbol{V}\tilde{\boldsymbol{x}}_0 + \sum_{r=2}^{R} \boldsymbol{H}_r (\boldsymbol{V} \otimes \cdots \otimes \boldsymbol{V})(\tilde{\boldsymbol{x}}_0 \otimes \cdots \otimes \tilde{\boldsymbol{x}}_0) + \boldsymbol{B}\boldsymbol{u}_0.
\end{aligned}$$

As $\bar{\boldsymbol{x}}_1 = \boldsymbol{V}^T \boldsymbol{x}_{\text{tmp}}$, it can be written as

$$\begin{aligned}
\bar{\boldsymbol{x}}_1 &= \boldsymbol{V}^T\boldsymbol{A}\boldsymbol{V}\tilde{\boldsymbol{x}}_0 + \sum_{r=2}^{R} \boldsymbol{V}^T\boldsymbol{H}_r (\boldsymbol{V} \otimes \cdots \otimes \boldsymbol{V})(\tilde{\boldsymbol{x}}_0 \otimes \cdots \otimes \tilde{\boldsymbol{x}}_0) + \boldsymbol{V}^T\boldsymbol{B}\boldsymbol{u}_0, \\
&= \tilde{\boldsymbol{A}}\tilde{\boldsymbol{x}}_0 + \sum_{r=2}^{R} \tilde{\boldsymbol{H}}_r (\tilde{\boldsymbol{x}}_0 \otimes \cdots \otimes \tilde{\boldsymbol{x}}_0) + \tilde{\boldsymbol{B}}\boldsymbol{u}_0, \\
&= \tilde{\boldsymbol{x}}_1.
\end{aligned}$$

Thus, $\tilde{\boldsymbol{x}}_1 = \bar{\boldsymbol{x}}_1$. With an inductive argument, it can be shown that $\tilde{\boldsymbol{x}}_k = \bar{\boldsymbol{x}}_k,\ k = 2, \ldots, K$ holds. Thus, $\bar{\boldsymbol{X}} = \tilde{\boldsymbol{X}}$ holds. $\square$

Chapter 3

Operator inference with physics-informed regularization

We first discuss literature on regularization of operator inference in Section 3.1 and then, in Section 3.2, briefly describe learning low-dimensional models with operator inference and motivates this work with a synthetic example. Section 3.3 proposes the physics-informed regularizer based on new upper bounds of stability radii of polynomial models for operator inference. The computational procedure is discussed in Section 3.4. Numerical results in Section 3.5 demonstrate that operator inference with the proposed regularizer learns stable models even when Tikhonov regularization and models learned without regularization are unstable.

3.1 Introduction

Our goal is to learn low-dimensional nonlinear dynamical-system models and to penalize unstable models. We build on operator inference [PW16b] that infers reduced models with polynomial nonlinear terms from snapshots data. Operator inference comes with recovery guarantees under certain assumptions [Peh20b, UP21b, UP21a] and it is a building block of more general learning methods that go far beyond polynomial nonlinear terms and exploit additional physical insights [QKMW19, QKPW20, BGK+20, KW22, SWK22, UWWP21, SK22, BGHD22]. In [SKHW20], operator inference is used together with a physics-informed lifting approach to learn a model of a large-scale combustion system, where it has been shown that regularization is important for obtaining stable models. A Tikhonov regularizer is proposed in [SKHW20], which has been further investigated in, e.g., [MHW21, Qia21]. In contrast, we propose a regularizer that goes beyond Tikhonov regularization and that is explicitly motivated by the nature of the polynomial—in particular, quadratic and cubic polynomials—model form, which in turn is given by the underlying physics. Building on the insights from [TVG94, Che07, Kra21], we penalize quadratic and cubic terms with large norms, which critically and provably influences the stability radius of the learned models and which is also in agreement with the findings in, e.g., [BB15, GT89]. To formulate the corresponding regularizers for operator inference, we derive novel upper bounds for the stability radii of cubic and quadratic-cubic models. We present numerical results that demonstrate improved stability of models learned with the proposed regularization compared to no regularization and Tikhonov regularization. We expect that similar regularization approaches can be used when learning reduced models using different non-intrusive model reduction techniques for the same type of polynomial nonlinear systems.

3.2 Motivation

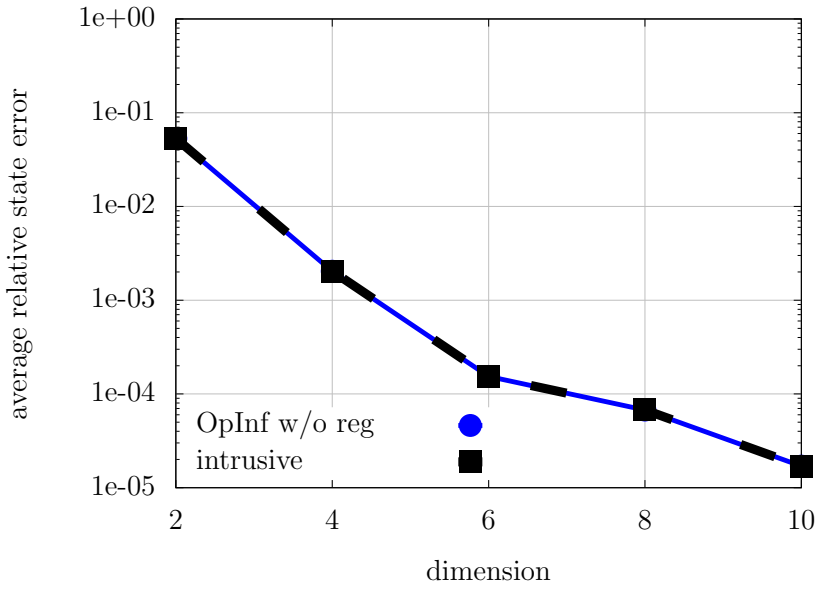
We demonstrate operator inference on a toy example. Consider the quadratic dynamical system

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \boldsymbol{A}(\mu)\boldsymbol{x}(t) + \boldsymbol{B}u(t) + \boldsymbol{F}\boldsymbol{x}^2(t)\,, \tag{3.1}$$

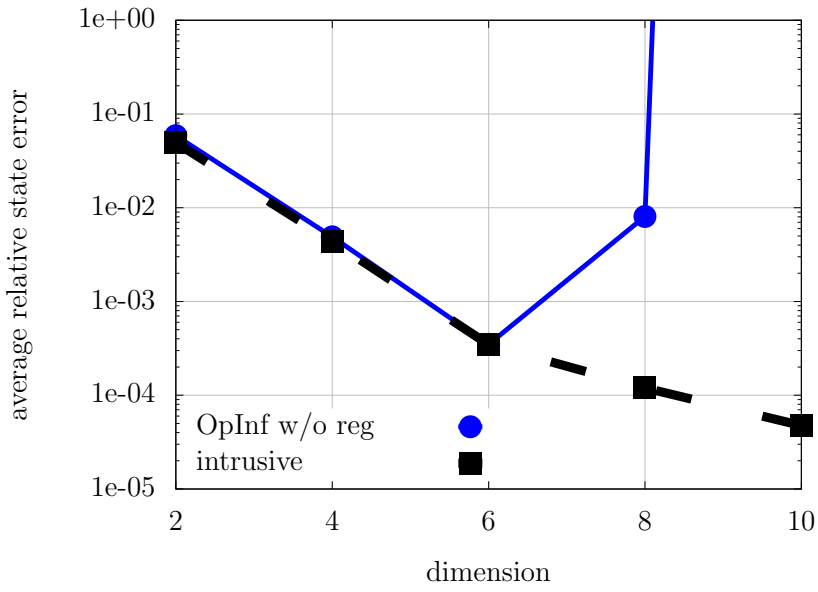
where $\boldsymbol{B} \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{F} \in \mathbb{R}^{N\times N(N+1)/2}$ have entries that are realizations of the uniform distribution in $[0, 1]$. The dimension is $N = 128$. The linear operator in (3.1) is $\boldsymbol{A}(\mu) = -\mu(\boldsymbol{A}_s + \boldsymbol{A}_s^T + 2N\boldsymbol{I})$, where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{A}_s \in \mathbb{R}^{N\times N}$ is a matrix that has as entries realizations of the uniform distribution in $[0, 1]$. The matrix $\boldsymbol{A}(\mu)$ is symmetric negative definite with probability 1. The parameter domain is $\mathcal{D} = [0.1, 1]$ and end time is $T = 1$. We discretize (3.1) with time-step size $\delta t = 10^{-3}$ and explicit Euler. For each training parameter $\mu \in \{0.1,\ \ldots,\ 1.0\}$, we generate a single ($M_b = 1$) input trajectory $\boldsymbol{U}_1^{\mathrm{b}}(\mu)$, whose entries are random with a uniform distribution in $[0, 2]$, and an initial condition $\boldsymbol{x}_{1,0}^{\mathrm{b}}$, whose entries follow a uniform distribution in $[0, 1]$. The corresponding state trajectories are $\boldsymbol{X}_1^{\mathrm{b}}(\mu_1), \ldots, \boldsymbol{X}_1^{\mathrm{b}}(\mu_M)$. A basis matrix $\boldsymbol{V} \in \mathbb{R}^{N\times n}$ is then constructed from the corresponding snapshots as described in Section 2.1. The reduced basis is generated for dimension $n = 2,\ \ldots,\ 10$. For parameter $\mu = 0.7$, we then construct $M_t = 3$ training inputs $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{M_t}$ with training initial conditions $\boldsymbol{x}_{1,0}, \ldots, \boldsymbol{x}_{M_t,0}$, which are sampled from the same distributions as the inputs and initial conditions for the basis construction. The corresponding training state trajectories are $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{M_t}$, to which we apply operator inference as described in Section 2.4. We use a first-order forward difference scheme to approximate the time derivative as in (2.16). Additionally, for benchmarking purposes, we also construct a reduced model (2.11) via the intrusive process described in Section 2.2.1.

Figure 3.1a shows the training error of the operator-inference model

$$e_{\text{train}} = \sum_{i=1}^{M_t} \frac{\|\boldsymbol{V}\hat{\boldsymbol{X}}_i - \boldsymbol{X}_i\|_F}{\|\boldsymbol{X}_i\|_F}\,, \tag{3.2}$$



(a) training error (3.2)



(b) test error (3.3)

Figure 3.1: Synthetic example: (a) The training error for the model learned via operator inference (OpInf) matches the error of the model obtained with intrusive model reduction. (b) When tested for a different input and initial condition at the same parameter, the model learned via operator inference (without regularization) is inaccurate and unstable in this example.

which indicates that the operator-inference model achieves a comparable error decay as the reduced model obtained from intrusive model reduction. However, if we simulate the operator-inference model at a test input trajectory $\boldsymbol{U}^{\text{test}}$, whose entries are sampled uniformly in $[0, 10]$, and test initial condition $\boldsymbol{x}_0^{\text{test}}$, with entries sampled uniformly in $[0, 1]$, and plot the error

$$e_{\text{test}} = \frac{\|\boldsymbol{V}\hat{\boldsymbol{X}}^{\text{test}} - \boldsymbol{X}^{\text{test}}\|_F}{\|\boldsymbol{X}^{\text{test}}\|_F}, \tag{3.3}$$

in Figure 3.1b, then an instability can be observed, compared to the reduced model from intrusive model reduction. The results indicate that operator inference is prone to overfitting, which can result in unstable behavior at test inputs as in this motivating example.

3.3 A physics-informed regularizer for operator inference

We propose a physics-informed regularizer for operator inference that penalizes unstable dynamical-system models. In Sections 3.3.1-3.3.3, we recapitulate the definition of the stability radius of quadratic models and derive new bounds for stability radii of cubic and quadratic-cubic dynamical-system models with Lyapunov stability criteria. We then propose a regularizer that penalizes models with small stability radii in Section 3.3.4. In the following, for each parameter $\mu \in \{\mu_1, \ldots, \mu_M\}$ in the training set, an operator-inference model is learned separately as discussed in Section 2.4. Thus, in this section, the parameter dependence of quantities is not explicitly denoted.

3.3.1 Stability radius of quadratic models of dynamical systems

We closely follow [TVG94, Kra21] to define the concept of stability radius for quadratic models of dynamical systems. In fact, the work [Kra21] is the motivation for the proposed

physics-informed regularizer.

Consider the autonomous quadratic reduced model

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{x}}(t) = \hat{\boldsymbol{A}}\hat{\boldsymbol{x}}(t) + \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t)) \tag{3.4}$$

where $\otimes$ denotes the Kronecker product. Notice that the quadratic term in (3.4) is denoted with $\hat{\boldsymbol{H}}$ instead of $\hat{\boldsymbol{F}}$ as in the reduced model (2.11) and the operator-inference model (2.17). The term $\hat{\boldsymbol{H}}$ is of dimension $n \times n^2$ and acts on the Kronecker product $\hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t)$, whereas the quadratic operator $\hat{\boldsymbol{F}}$ is of dimension $n \times n(n+1)/2$ and acts on $\hat{\boldsymbol{x}}^2(t)$ without duplicates due to the commutativity of the multiplication operators, see Section 2.4. However, one can transform between the two different representations. Thus, the model given in (3.4) is a different representation of an autonomous version of the reduced model (2.11) and the operator-inference model (2.17) without a cubic term; cf. Section 3.3.2. See Section 3.3.3 for models with cubic terms. Similarly, we can represent a full model with only a quadratic term given in the form of (2.2) as (3.4).

Let now without loss of generality $\hat{\boldsymbol{x}}_e = \boldsymbol{0}$ be an equilibrium point of (3.4), i.e., $\hat{\boldsymbol{A}}\hat{\boldsymbol{x}}_e + \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}}_e \otimes \hat{\boldsymbol{x}}_e) = \boldsymbol{0}$. The domain of attraction $\mathcal{A}(\hat{\boldsymbol{x}}_e)$ of the equilibrium $\hat{\boldsymbol{x}}_e$ is then defined as the set of initial conditions that lead to the equilibrium point $\hat{\boldsymbol{x}}_e$ as a steady state, i.e.,

$$\mathcal{A}(\hat{\boldsymbol{x}}_e) = \{\hat{\boldsymbol{x}}_0 : \lim_{t\to\infty} \hat{\boldsymbol{x}}(t) = \hat{\boldsymbol{x}}_e\},$$

where $\hat{\boldsymbol{x}}(t)$ is the state at time t of (3.4) with initial condition $\hat{\boldsymbol{x}}_0$. Directly working with the stability domain is challenging from an analytic and computational point of view and thus one typically resorts to deriving subsets $D \subseteq \mathcal{A}(\hat{\boldsymbol{x}}_e)$. To measure a subset D, we build on the Lyapunov theory to derive a stability radius. We remark that Lyapunov introduced the first precise notion of stability of dynamical systems. A short history of the Lyapunov theory can be found in [HP05, Section 3.1.7]. Since its inception, the Lyapunov theory has

become an important concept for stability analysis of dynamical systems. In the following, we build on Lyapunov theory as used in [TVG94, Che07, Kra21].

If there exists a Lyapunov function $\nu : \mathbb{R}^N \to \mathbb{R}^+$ that is continuously differentiable and that satisfies

$$\nu(\hat{\boldsymbol{x}}) > 0,\ \dot{\nu}(\hat{\boldsymbol{x}}) < 0, \quad \forall \hat{\boldsymbol{x}} \in \mathcal{A}(\hat{\boldsymbol{x}}_e)\,,$$

then model (3.4) is locally asymptotically stable about $\hat{\boldsymbol{x}}_e$. Here, $\dot{\nu}(\hat{\boldsymbol{x}})$ means $\dot{\nu}(\hat{\boldsymbol{x}}) = \frac{\mathrm{d}\nu}{\mathrm{d}\hat{\boldsymbol{x}}}\hat{\boldsymbol{f}}(\hat{\boldsymbol{x}})$, where $\hat{\boldsymbol{f}}(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{A}}\hat{\boldsymbol{x}} + \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})$ is the right-hand side function of the corresponding dynamical system. As shown in [TVG94, Che07, Kra21], given a Lyapunov function ν, an estimate $D(\rho) \subseteq \mathcal{A}(\hat{\boldsymbol{x}}_e)$ of the domain of attraction $\mathcal{A}(\hat{\boldsymbol{x}}_e)$ is given by

$$D(\rho) = \{\hat{\boldsymbol{x}} : \nu(\hat{\boldsymbol{x}}) \leq \rho^2,\ \dot{\nu}(\hat{\boldsymbol{x}}) < 0\}\,,$$

where we refer to ρ as the stability radius.

Consider an autonomous quadratic model (3.4) with Lyapunov function $\nu(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}}^T \boldsymbol{P} \hat{\boldsymbol{x}}$, where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix that satisfies

$$\boldsymbol{L}\boldsymbol{L}^T = -\hat{\boldsymbol{A}}^T \boldsymbol{P} - \boldsymbol{P}\hat{\boldsymbol{A}}\,, \tag{3.5}$$

for an arbitrary matrix $\boldsymbol{L} \in \mathbb{R}^{n \times n}$, and for the linear operator $\hat{\boldsymbol{A}}$, that is Hurwitz. As a remark, we add that the textbook [HP05] discusses the stability of a linear model with respect to perturbations in $\hat{\boldsymbol{A}}$ and then derives a stability radius for the same. The notion of stability radius in [HP05] is different to what we derive here, because we are interested in perturbations in the initial conditions whereas [HP05] considers perturbations in the matrix $\hat{\boldsymbol{A}}$.

The derivative of the Lyapunov function along a trajectory is

$$\dot{\nu}(\hat{\boldsymbol{x}}) = \dot{\hat{\boldsymbol{x}}}^T \boldsymbol{P}\hat{\boldsymbol{x}} + \hat{\boldsymbol{x}}^T \boldsymbol{P}\dot{\hat{\boldsymbol{x}}}\,.$$

Building on [Kra21, Proposition 3.1], we obtain the radius

$$\hat{\rho} = \frac{\sigma_{\min}^2(\boldsymbol{L})}{2\sqrt{\|\boldsymbol{P}\|_F}\|\hat{\boldsymbol{H}}\|_F} \tag{3.6}$$

and that $D(\hat{\rho}) \subseteq \mathcal{A}(\hat{\boldsymbol{x}}_e)$ is a subset of $\mathcal{A}(\hat{\boldsymbol{x}}_e)$, if $\hat{\boldsymbol{A}}$ is Hurwitz, i.e., the real parts of all eigenvalues of $\hat{\boldsymbol{A}}$ are negative. Notice that in contrast to the 2-norm $\|\cdot\|_2$ used in [Kra21, Proposition 3.1], we state the radius (3.6) with respect to the Frobenius norm $\|\cdot\|_F$, which leads to an operator inference problem that can be solved more efficiently than when working with the $\|\cdot\|_2$ norm.

We comment on the bound $\hat{\rho}$ defined in (3.6) of the stability radius ρ. The same comments apply to the bounds that are derived in the following sections. We will use the bound $\hat{\rho}$ to formulate a regularization terms. Thus, what is important for our approach is that penalizing a small $\hat{\rho}$ encourages stabler models that have a larger stability radius ρ, and our numerical experiments provide empirical evidence of this. This means that for our approach, it is less critical how close $\hat{\rho}$ is to ρ in absolute terms and more important that $\hat{\rho}$ and ρ follow the same trend in the sense that penalizing the bound $\hat{\rho}$ leads to a good penalization of ρ.

3.3.2 Stability radius of cubic models of dynamical systems

Consider the autonomous cubic reduced model

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{x}}(t) = \hat{\boldsymbol{A}}\hat{\boldsymbol{x}}(t) + \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t)) \tag{3.7}$$

which is an equivalent representation of an autonomous version of the reduced model (2.11) and the operator-inference model (2.17) with no input and no quadratic term.

Proposition 3.1. *Let $\nu(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}}^T \boldsymbol{P} \hat{\boldsymbol{x}}$ be a Lyapunov function, where $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix that satisfies (3.5). Then an estimate of the domain of attraction $\mathcal{A}(\hat{\boldsymbol{x}}_e)$, of the equilibrium point $\hat{\boldsymbol{x}}_e$ of (3.7), is given by $D(\hat{\rho}) \subseteq \mathcal{A}(\hat{\boldsymbol{x}}_e)$, where*

$$\hat{\rho} = \frac{\sigma_{min}(\boldsymbol{L})}{\sqrt{2\|\hat{\boldsymbol{K}}\|_F}}\,. \tag{3.8}$$

Proof. A bound of the derivative for the Lyapunov function $\nu(\hat{\boldsymbol{x}})$ along a trajectory is given as

$$\begin{aligned}
\dot{\nu}(\hat{\boldsymbol{x}}) &= \dot{\hat{\boldsymbol{x}}}^T \boldsymbol{P} \hat{\boldsymbol{x}} + \hat{\boldsymbol{x}}^T \boldsymbol{P} \dot{\hat{\boldsymbol{x}}} \\
&= \big[\hat{\boldsymbol{A}}\hat{\boldsymbol{x}} + \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})\big]^T \boldsymbol{P} \hat{\boldsymbol{x}} + \hat{\boldsymbol{x}}^T \boldsymbol{P} \big[\hat{\boldsymbol{A}}\hat{\boldsymbol{x}} + \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})\big] \\
&= \hat{\boldsymbol{x}}^T [\hat{\boldsymbol{A}}^T \boldsymbol{P} + \boldsymbol{P}\hat{\boldsymbol{A}}]\hat{\boldsymbol{x}} + (\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})^T \hat{\boldsymbol{K}}^T \boldsymbol{P} \hat{\boldsymbol{x}} + \hat{\boldsymbol{x}}^T \boldsymbol{P} \hat{\boldsymbol{K}} (\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) \\
&= -\hat{\boldsymbol{x}}^T \boldsymbol{L}\boldsymbol{L}^T \hat{\boldsymbol{x}} + 2\hat{\boldsymbol{x}}^T \boldsymbol{P} \hat{\boldsymbol{K}} (\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) \\
&\leq -\sigma_{\min}^2(\boldsymbol{L}) \|\hat{\boldsymbol{x}}\|_F^2 + 2\|\hat{\boldsymbol{x}}\|_F^4 \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F\,.
\end{aligned}$$

We consider the region where the Lyapunov function gradient is negative, which gives us

$$\|\hat{\boldsymbol{x}}\|_F^2 < \frac{\sigma_{\min}^2(\boldsymbol{L})}{2\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F} \implies \dot{\nu}(\hat{\boldsymbol{x}}) < 0\,.$$

Thus, for the Lyapunov function, the inequality

$$\nu(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}}^T \boldsymbol{P} \hat{\boldsymbol{x}} \leq \|\hat{\boldsymbol{x}}\|_F^2 \|\boldsymbol{P}\|_F < \frac{\sigma_{\min}^2(\boldsymbol{L})}{2\|\hat{\boldsymbol{K}}\|_F} = \hat{\rho}^2$$

holds, and thus the stability radius of the cubic system is

$$\hat{\rho} = \frac{\sigma_{\min}(\boldsymbol{L})}{\sqrt{2\|\hat{\boldsymbol{K}}\|_F}}.$$

□

3.3.3 Stability radius of quadratic-cubic models of dynamical systems

Consider the autonomous quadratic-cubic reduced model

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{\boldsymbol{x}}(t) = \hat{\boldsymbol{A}}\hat{\boldsymbol{x}}(t) + \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t)) + \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t)) \tag{3.9}$$

which is a different representation of an autonomous version of the reduced model (2.11) and the operator-inference model (2.17).

Proposition 3.2. *Let $\nu(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}}^T\boldsymbol{P}\hat{\boldsymbol{x}}$ be a Lyapunov function, where $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix that satisfies (3.5). Then an estimate of the domain of attraction $\mathcal{A}(\hat{\boldsymbol{x}}_e)$, of the equilibrium point $\hat{\boldsymbol{x}}_e$ of* (3.9)*, is given by $D(\hat{\rho}) \subseteq \mathcal{A}(\hat{\boldsymbol{x}}_e)$, where*

$$\hat{\rho} = \frac{\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F}}{2\|\hat{\boldsymbol{K}}\|_F} - \frac{\sqrt{\|\boldsymbol{P}\|_F}\|\hat{\boldsymbol{H}}\|_F}{2\|\hat{\boldsymbol{K}}\|_F}. \tag{3.10}$$

Proof. A bound of the derivative of the Lyapunov function along a trajectory is given as

$$\begin{aligned}
\dot{\nu}(\hat{\boldsymbol{x}}) &= \dot{\hat{\boldsymbol{x}}}^T \boldsymbol{P}\hat{\boldsymbol{x}} + \hat{\boldsymbol{x}}^T \boldsymbol{P}\dot{\hat{\boldsymbol{x}}} \\
&= \big[\hat{\boldsymbol{A}}\hat{\boldsymbol{x}} + \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) + \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})\big]^T \boldsymbol{P}\hat{\boldsymbol{x}} + \\
&\quad \hat{\boldsymbol{x}}^T \boldsymbol{P}\big[\hat{\boldsymbol{A}}\hat{\boldsymbol{x}} + \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) + \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})\big] \\
&= \hat{\boldsymbol{x}}^T [\hat{\boldsymbol{A}}^T \boldsymbol{P} + \boldsymbol{P}\hat{\boldsymbol{A}}]\hat{\boldsymbol{x}} + (\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})^T \hat{\boldsymbol{H}}^T \boldsymbol{P}\hat{\boldsymbol{x}} + (\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})^T \hat{\boldsymbol{K}}^T \boldsymbol{P}\hat{\boldsymbol{x}} + \\
&\quad \hat{\boldsymbol{x}}^T \boldsymbol{P}\hat{\boldsymbol{H}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) + \hat{\boldsymbol{x}}^T \boldsymbol{P}\hat{\boldsymbol{K}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) \\
&= -\hat{\boldsymbol{x}}^T \boldsymbol{L}\boldsymbol{L}^T \hat{\boldsymbol{x}} + 2\hat{\boldsymbol{x}}^T \boldsymbol{P}\hat{\boldsymbol{H}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) + 2\hat{\boldsymbol{x}}^T \boldsymbol{P}\hat{\boldsymbol{K}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}}) \\
&\leq -\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{x}}\|_F^2 + 2\|\hat{\boldsymbol{x}}\|_F^3 \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F + 2\|\hat{\boldsymbol{x}}\|_F^4 \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F \,,
\end{aligned}$$

where the matrix $\boldsymbol{L}$ is as defined in (3.5). We consider the region where the Lyapunov function gradient satisfies $\dot{\nu}(\hat{\boldsymbol{x}}) < 0$, which gives us,

$$-\sigma_{\min}^2(\boldsymbol{L}) + 2\|\hat{\boldsymbol{x}}\|_F \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F + 2\|\hat{\boldsymbol{x}}\|_F^2 \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F < 0 \,. \tag{3.11}$$

Roots of the quadratic equation in (3.11) in $\|\hat{\boldsymbol{x}}\|_F$ are

$$\begin{aligned}
R_1 &= \frac{-\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F - \sqrt{\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L})\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F}}{2\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F} \,, \\
R_2 &= \frac{-\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F + \sqrt{\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L})\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F}}{2\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F} \,.
\end{aligned}$$

Both R_1 and R_2 are real as $\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L})\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F \geq 0$.

For $\hat{\boldsymbol{x}}$ with $R_1 < \|\hat{\boldsymbol{x}}\|_F < R_2$, we get

$$\begin{aligned}
&(\|\hat{\boldsymbol{x}}\|_F - R_1)(\|\hat{\boldsymbol{x}}\|_F - R_2) < 0 \\
&\implies -\sigma_{\min}^2(\boldsymbol{L}) + 2\|\hat{\boldsymbol{x}}\|_F\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F + 2\|\hat{\boldsymbol{x}}\|_F^2\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{K}}\|_F < 0 \\
&\implies \dot{\nu}(\hat{\boldsymbol{x}}) < 0.
\end{aligned}$$

Because $\|\hat{\boldsymbol{x}}\|_F \geq 0$ and $R_1 \leq 0$, we get,

$$0 \leq \|\hat{\boldsymbol{x}}\|_F < R_2$$

as the set of $\hat{\boldsymbol{x}}$ for which the condition $\dot{\nu}(\hat{\boldsymbol{x}}) < 0$ is satisfied. Thus, for the Lyapunov function, the inequality

$$\nu(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}}^T\boldsymbol{P}\hat{\boldsymbol{x}} \leq \|\hat{\boldsymbol{x}}\|_F^2\|\boldsymbol{P}\|_F < R_2^2\|\boldsymbol{P}\|_F = \hat{\rho}^2$$

holds, and thus the stability radius of the quadratic-cubic system is

$$\hat{\rho} = \sqrt{R_2^2\|\boldsymbol{P}\|_F} = \frac{\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F}}{2\|\hat{\boldsymbol{K}}\|_F} - \frac{\sqrt{\|\boldsymbol{P}\|_F}\|\hat{\boldsymbol{H}}\|_F}{2\|\hat{\boldsymbol{K}}\|_F}.$$

□

3.3.4 Operator inference with physics-informed regularizer

We now formulate a physics-informed regularizer for learning quadratic, cubic, and quadratic-cubic models with operator inference.

3.3.4.1 Physics-informed regularizer for quadratic models

For a quadratic nonlinear model, the stability radius $\hat{\rho}$, which is derived in (3.6), grows inversely proportional to the norm $\|\hat{\boldsymbol{H}}\|_F$ of the quadratic term $\hat{\boldsymbol{H}} \in \mathbb{R}^{n\times n^2}$ in (3.4). Notice

that other results on stability analysis for quadratic systems, e.g., [BB15, GT89], also show that a small norm of the quadratic term can increase the stability radius.

The models that we infer with operator inference have the form (2.17) and thus the quadratic term $\hat{\boldsymbol{F}}$ is of dimension $n \times n(n+1)/2$. However, models of the form (3.4) with $\hat{\boldsymbol{H}}$ can be transformed into models with the quadratic operator $\hat{\boldsymbol{F}}$ such that $\hat{\boldsymbol{F}}\hat{\boldsymbol{x}}^2 = \hat{\boldsymbol{H}}(\hat{\boldsymbol{x}} \otimes \hat{\boldsymbol{x}})$ holds for any $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ and $\|\hat{\boldsymbol{H}}\|_F \leq \|\hat{\boldsymbol{F}}\|_F$ holds as well. We can construct $\hat{\boldsymbol{H}}$ such that $\|\hat{\boldsymbol{H}}\|_F = \|\hat{\boldsymbol{F}}\|_F$, by filling the additional columns of $\hat{\boldsymbol{H}}$ with zeros. Thus, we obtain that if we regularize the norm $\|\hat{\boldsymbol{F}}\|_F$, we also regularize the norm $\|\hat{\boldsymbol{H}}\|_F$ of a corresponding $\hat{\boldsymbol{H}}$, which in turn means that the denominator of the radius $\hat{\rho}$ is regularized. This leads to the optimization problem for inferring model (2.17) with operator inference and the proposed physics-informed regularizer (PIR-OpInf)

$$\min_{\hat{\boldsymbol{A}},\hat{\boldsymbol{B}},\hat{\boldsymbol{F}}} J(\hat{\boldsymbol{A}}, \hat{\boldsymbol{B}}, \hat{\boldsymbol{F}}, \boldsymbol{0}, \lambda) + \lambda \|\hat{\boldsymbol{F}}\|_F^2 \,, \tag{3.12}$$

with J defined in (2.15), with $\hat{\boldsymbol{G}}$ set to the zero matrix because we only infer a quadratic model, and $\lambda > 0$ being a regularization parameter. Notice that increasing λ means more severely penalizing the norm $\|\hat{\boldsymbol{F}}\|_F$, which in turn leads to a potential increase of the radius $\hat{\rho}$ and thus a more stable inferred model in the sense of Lyapunov. We solve the optimization problem for PIR-OpInf using the SDPT3 solver of the CVX toolbox in MATLAB [GB14, GB08].

The PIR-OpInf problem (3.12) imposes no constraints on the linear operator $\hat{\boldsymbol{A}}$. In particular, there is no guarantee that the inferred $\hat{\boldsymbol{A}}$ is Hurwitz and thus there can exist eigenvalues with non-negative real parts. To ensure a linear operator that is Hurwitz, we apply an eigenvalue reflection as a post-processing step. Let $\hat{\boldsymbol{A}} = \boldsymbol{Q}_A \boldsymbol{\Sigma}_A \boldsymbol{Q}_A^{-1}$ be the eigendecomposition of $\hat{\boldsymbol{A}}$. If $\hat{\boldsymbol{A}}$ is not diagonalizable, we reduce the dimension n until a matrix $\hat{\boldsymbol{A}}$ is inferred that is diagonalizable. Notice that this process stops in a finite number of

steps because $\hat{\boldsymbol{A}}$ is diagonalizable if $n = 1$ and $\hat{\boldsymbol{A}}$ is non-zero. Without loss of generality, let $\sigma_1, \ldots, \sigma_r$ be eigenvalues with non-negative real parts and let $\sigma_{r+1}, \ldots, \sigma_n$ be all other eigenvalues. Denote with $\Re(\sigma)$ and $\Im(\sigma)$ the real and imaginary part, respectively, for a complex number $\sigma \in \mathbb{C}$. Then, we replace $\hat{\boldsymbol{A}}$ with the matrix

$$\boldsymbol{Q}_A \operatorname{diag}(-\epsilon + \Im(\sigma_1), \ldots, -\epsilon + \Im(\sigma_r), \sigma_{r+1}, \ldots, \sigma_n) \boldsymbol{Q}_A^{-1},$$

which replaces the positive real parts of the eigenvalues with a negative real number given by the small positive threshold $\epsilon > 0$. Note that the post-processing also needs to be applied after interpolating at a new parameter $\mu \in \mathcal{D}$ outside of the training set; cf. Section 2.2.1.

In our numerical experiments, when eigenvalues have positive real parts, then they typically have small magnitude because the underlying systems from which data are sampled are stable and thus the instability in the learned model is due to, e.g., insufficient data and other shortcomings of the learning. Because we encounter unstable eigenvalues with small magnitude only, the reflection described in the previous paragraph is sufficient in our experiments. This is in contrast to other settings where the underlying systems are unstable and then eigenvalues of learned system matrices have real parts with large magnitudes. In these cases, other post-processing strategies to obtain Hurwitz linear operators are warranted and we refer to, e.g., [KvAB14, GA16, Kö14] for additional details.

3.3.4.2 Physics-informed regularizer for cubic models

For a cubic nonlinear model, the stability radius $\hat{\rho}$, which is derived in (3.8), grows inversely proportional to $\|\hat{\boldsymbol{K}}\|_F^{1/2}$ of the cubic term $\hat{\boldsymbol{K}} \in \mathbb{R}^{n \times n^3}$ in (3.7). The models that we infer with operator inference have the form (2.17) and thus the cubic term $\hat{\boldsymbol{G}}$ is of dimension $n \times n(n+1)(n+2)/6$. Similar to the quadratic model, (3.7) with $\hat{\boldsymbol{K}}$ can be transformed into models with cubic operator $\hat{\boldsymbol{G}}$, such that $\hat{\boldsymbol{G}}\hat{\boldsymbol{x}}^3 = \hat{\boldsymbol{K}}(\hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t) \otimes \hat{\boldsymbol{x}}(t))$ holds for any

$\hat{\boldsymbol{x}} \in \mathbb{R}^n$ and $\|\hat{\boldsymbol{K}}\|_F \leq \|\hat{\boldsymbol{G}}\|_F$ holds as well. We can construct $\hat{\boldsymbol{K}}$ such that $\|\hat{\boldsymbol{K}}\|_F = \|\hat{\boldsymbol{G}}\|_F$ by filling the additional columns of $\hat{\boldsymbol{K}}$ with zeros. Thus, we obtain that if we regularize the norm $\|\hat{\boldsymbol{G}}\|_F$, we also regularize the norm $\|\hat{\boldsymbol{K}}\|_F$ of a corresponding $\hat{\boldsymbol{K}}$, which in turn means that the denominator of the radius $\hat{\rho}$ is regularized. For the cubic model, this leads to the optimization problem for inferring model (2.17) with PIR-OpInf

$$\min_{\hat{\boldsymbol{A}},\hat{\boldsymbol{B}},\hat{\boldsymbol{G}}} J(\hat{\boldsymbol{A}}, \hat{\boldsymbol{B}}, \boldsymbol{0}, \hat{\boldsymbol{G}}, \lambda) + \lambda \|\hat{\boldsymbol{G}}\|_F \,, \tag{3.13}$$

with $\hat{\boldsymbol{F}}$ set to the zero matrix because there is no quadratic term. The same post-processing as for quadratic models in Section 3.3.4.1 is applied to ensure an $\hat{\boldsymbol{A}}$ that is Hurwitz.

3.3.4.3 Physics-informed regularizer for quadratic-cubic models

For a quadratic-cubic nonlinear model, by taking derivatives of the stability radius $\hat{\rho}$, derived in (3.10), with respect to $\|\hat{\boldsymbol{H}}\|_F$ and $\|\hat{\boldsymbol{K}}\|_F$ we get,

$$\frac{\partial \hat{\rho}}{\partial \|\hat{\boldsymbol{H}}\|_F} = \frac{\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F - \sqrt{\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F}}{2\|\hat{\boldsymbol{K}}\|_F \sqrt{\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\hat{\boldsymbol{K}}\|_F}} \,,$$

$$\frac{\partial \hat{\rho}}{\partial \|\hat{\boldsymbol{K}}\|_F} = \frac{\sqrt{\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F^2} \sqrt{\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\hat{\boldsymbol{K}}\|_F} - \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F^2 - \sigma_{\min}^2(\boldsymbol{L}) \|\hat{\boldsymbol{K}}\|_F}{2\|\hat{\boldsymbol{K}}\|_F^2 \sqrt{\|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\hat{\boldsymbol{K}}\|_F}} .$$

For $\partial \hat{\rho} / \partial \|\hat{\boldsymbol{H}}\|_F$ we get

$$\begin{aligned}
&\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F > \|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 \\
&\implies \sqrt{\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F} > \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F \\
&\implies \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{H}}\|_F - \sqrt{\|\boldsymbol{P}\|_F^2 \|\hat{\boldsymbol{H}}\|_F^2 + 2\sigma_{\min}^2(\boldsymbol{L}) \|\boldsymbol{P}\|_F \|\hat{\boldsymbol{K}}\|_F} < 0 \\
&\implies \frac{\partial \hat{\rho}}{\partial \|\hat{\boldsymbol{H}}\|_F} < 0 \,.
\end{aligned}$$

For $\partial\hat{\rho}/\partial\|\hat{\boldsymbol{K}}\|_F$, squaring the two terms in the numerator gives us

$$\begin{aligned}&\left(\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2}\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F}\right)^2\\&=\|\boldsymbol{P}\|_F^2\|\hat{\boldsymbol{H}}\|_F^4+2\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F\,,\end{aligned}$$

and

$$\begin{aligned}&\left(\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F\right)^2\\&=\|\boldsymbol{P}\|_F^2\|\hat{\boldsymbol{H}}\|_F^4+2\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F+\sigma_{\min}^4(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F^2\,.\end{aligned}$$

Comparing the two terms, and using the fact that $\sigma_{\min}^4(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F^2$ and all other terms are non-negative, leads to the inequality

$$\begin{aligned}&\left(\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F\right)^2>\left(\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2}\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F}\right)^2\\&\implies\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F>\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2}\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F}\\&\implies\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2}\sqrt{\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2+2\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F}-\|\boldsymbol{P}\|_F\|\hat{\boldsymbol{H}}\|_F^2-\sigma_{\min}^2(\boldsymbol{L})\|\hat{\boldsymbol{K}}\|_F<0\\&\implies\frac{\partial\hat{\rho}}{\partial\|\hat{\boldsymbol{K}}\|_F}<0\,.\end{aligned}$$

That the partial derivatives are negative, $\partial\hat{\rho}/\partial\|\hat{\boldsymbol{H}}\|_F<0$ and $\partial\hat{\rho}/\partial\|\hat{\boldsymbol{K}}\|_F<0$, motivates that decreasing $\|\hat{\boldsymbol{H}}\|_F$ and $\|\hat{\boldsymbol{K}}\|_F$ can increase $\hat{\rho}$, which in turn motivates regularizing with respect to both $\|\hat{\boldsymbol{H}}\|_F$ and $\|\hat{\boldsymbol{K}}\|_F$. As described in Sections 3.3.4.1 and 3.3.4.2, by regularizing $\|\hat{\boldsymbol{F}}\|_F$ and $\|\hat{\boldsymbol{G}}\|_F$, we also regularize $\|\hat{\boldsymbol{H}}\|_F$ and $\|\hat{\boldsymbol{K}}\|_F$ respectively. For the quadratic-cubic model, this leads to the optimization problem for inferring model (2.17) with PIR-OpInf

$$\min_{\hat{\boldsymbol{A}},\hat{\boldsymbol{B}},\hat{\boldsymbol{F}},\hat{\boldsymbol{G}}}J(\hat{\boldsymbol{A}},\hat{\boldsymbol{B}},\hat{\boldsymbol{F}},\hat{\boldsymbol{G}},\lambda)+\lambda(\|\hat{\boldsymbol{F}}\|_F^2+\|\hat{\boldsymbol{G}}\|_F)\,.\tag{3.14}$$

The same post-processing as for quadratic models in Section 3.3.4.1 is applied to ensure an $\hat{\boldsymbol{A}}$ that is Hurwitz.

A similar approach as described in sections 3.3.1-3.3.3 can be used for models with higher order nonlinear terms, to derive the expression for the estimated stability radius and to use it for regularization.

3.4 Computational procedure of physics-informed operator inference

To select a regularization parameter for PIR-OpInf (3.12)–(3.14), we propose a parameter-selection scheme in Section 3.4.1 and Section 3.4.2. Section 3.4.3 presents Algorithm 1 that summarizes the computational procedure for operator inference with physics-informed regularization.

3.4.1 A regularization parameter-selection scheme for PIR-OpInf

Let $\mu_1, \ldots, \mu_M$ be the training parameters and recall that $\boldsymbol{X}_1(\mu_i), \ldots, \boldsymbol{X}_{M_t}(\mu_i)$ are the training trajectories with input trajectories $\boldsymbol{U}_1(\mu_i), \ldots, \boldsymbol{U}_{M_t}(\mu_i)$, respectively, for $i = 1, \ldots, M$; cf. Section 2.1. Define the minimum $\lambda_{\min}$ and maximum $\lambda_{\max}$ of the regularization parameter and discretize the interval $[\lambda_{\min}, \lambda_{\max}] \subset \mathbb{R}$ with m points

$$\lambda_{\min} = \lambda_1 < \cdots < \lambda_m = \lambda_{\max}. \tag{3.15}$$

For each λ_i, we learn a model $\hat{\Sigma}_{ij}$ with PIR-OpInf (3.12) for μ_j, with $i = 1, \ldots, m$ and $j = 1, \ldots, M$. In our case, when $\hat{\Sigma}$ refers to a model, then we mean the corresponding matrices that define the model. For example, in the case of a cubic model as in (2.17), the notation $\hat{\Sigma}$ refers to the matrices $\hat{\boldsymbol{A}}, \hat{\boldsymbol{B}}, \hat{\boldsymbol{F}}$ and $\hat{\boldsymbol{G}}$, which is different from the way the symbol

Σ is typically used in system and control literature where it refers to the system rather than a realization (model) of the system. Then, for $j = 2, \ldots, M-1$ and for $i = 1, \ldots, m$, we derive $\hat{\Pi}_{ij}$ by interpolating between models

$$\hat{\Sigma}_{i,1}, \ldots, \hat{\Sigma}_{i,j-1}, \hat{\Sigma}_{i,j+1}, \ldots, \hat{\Sigma}_{i,M} \tag{3.16}$$

corresponding to parameters $\mu_1, \ldots, \mu_{j-1}, \mu_{j+1}, \ldots, \mu_M$, i.e., the parameter μ_j corresponding to model $\hat{\Sigma}_{ij}$ is left out from the interpolation process. Notice that all models in (3.16) are trained with the same regularization parameter λ_i. The interpolated model $\hat{\Pi}_{ij}$ is integrated in time with the input trajectories $\boldsymbol{U}_1(\mu_j), \ldots, \boldsymbol{U}_{M_t}(\mu_j)$ corresponding to parameter μ_j to obtain the trajectories $\hat{\boldsymbol{X}}_1^{(i)}(\mu_j), \ldots, \hat{\boldsymbol{X}}_{M_t}^{(i)}(\mu_j)$ and the error

$$e_{ij}^{\text{val}} = \sum_{\ell=1}^{M_t} \frac{\|\boldsymbol{V}\hat{\boldsymbol{X}}_\ell^{(i)}(\mu_j) - \boldsymbol{X}_\ell(\mu_j)\|_F}{\|\boldsymbol{X}_\ell(\mu_j)\|_F} \tag{3.17}$$

is assigned to the pair of regularization parameter λ_i and parameter μ_j, where $\boldsymbol{V}$ is the basis matrix. We then pick λ^* by solving

$$\underset{i=1,\ldots,m}{\arg\min} \quad \frac{1}{M-2} \sum_{j=2}^{M-1} e_{ij}^{\text{val}}. \tag{3.18}$$

Notice that error due to interpolating between models enters the validation error (3.17) and thus the selection criterion (3.18) for λ^*. This is in contrast to other parameter-selection schemes for operator inference that are either formulated in parameter-independent settings or ignore the parameter dependency in the selection process [SKHW20, MHW21].

3.4.2 Selecting regularization parameters based on initial conditions

In case the operators are independent of a parameter μ, we select the regularization parameter based on trajectories generated with multiple initial conditions. For initial conditions $\boldsymbol{x}_0^1, \ldots, \boldsymbol{x}_0^{M_t}$, and inputs $\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{M_t}$, the training trajectories are $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{M_t}$. Define the minimum $\lambda_{\min}$ and maximum $\lambda_{\max}$ of the regularization parameter and discretize the interval $[\lambda_{\min}, \lambda_{\max}] \subset \mathbb{R}$ with m points

$$\lambda_{\min} = \lambda_1 < \cdots < \lambda_m = \lambda_{\max}.$$

We split the training data into L different folds, with the data in each fold being

$$\boldsymbol{X}_l^f = \begin{cases} [\boldsymbol{X}_{(l-1)f_l+1} \ \ldots \ \boldsymbol{X}_{lf_l}], & \text{for } l = 1, \ldots, L-1 \\ [\boldsymbol{X}_{(l-1)f_l+1} \ \ldots \ \boldsymbol{X}_{M_t}], & \text{for } l = L \end{cases} \tag{3.19}$$

with the rounded value $f_l = \left\lfloor \frac{M_t}{L} \right\rceil$. For fold l, define the validation data as

$$\bar{\boldsymbol{X}}_l^f = \begin{cases} [\boldsymbol{X}_{f_l+1} \ \ldots \ \boldsymbol{X}_{M_t}], & \text{for } l = 1 \\ [\boldsymbol{X}_1 \ \ldots \ \boldsymbol{X}_{(l-1)f_l} \ \boldsymbol{X}_{lf_l+1} \ \ldots \ \boldsymbol{X}_{M_t}], & \text{for } l = 2, \ldots, L-1 \\ [\boldsymbol{X}_1 \ \ldots \ \boldsymbol{X}_{(l-1)f_l}], & \text{for } l = L\,. \end{cases}$$

For each λ_i, we learn the model $\hat{\Sigma}_{il}$ with PIR-OpInf (3.12), using the data $\bar{\boldsymbol{X}}_l^f$, with $i = 1, \ldots, m$ and $l = 1, \ldots, L$. The learned model $\hat{\Sigma}_{il}$ is integrated in time with input trajectories and initial conditions corresponding to $\boldsymbol{X}_l^f$, to obtain trajectories $\hat{\boldsymbol{X}}_l^f(\lambda_i)$, and

the error

$$e_{il}^{\text{val}} = \frac{\|\boldsymbol{V}\hat{\boldsymbol{X}}_l^f(\lambda_i) - \boldsymbol{X}_l^f\|_F}{\|\boldsymbol{X}_l^f\|_F} \tag{3.20}$$

is assigned to the pair of regularization parameter λ_i and fold l, where $\boldsymbol{V}$ is the basis matrix. We then pick λ^* by solving

$$\underset{i=1,\ldots,m}{\arg\min} \quad \frac{1}{L}\sum_{l=1}^{L} e_{il}^{\text{val}}\,. \tag{3.21}$$

3.4.3 Algorithm of operator inference with physics-informed regularizer

Algorithm 1 summarizes the computational procedure of the proposed approach. Inputs are the basis matrix $\boldsymbol{V}$, which is constructed from trajectories as described in Section 2.2.1, and the training trajectories $\boldsymbol{X}_1(\mu_i), \ldots, \boldsymbol{X}_{M_t}(\mu_i)$ and inputs $\boldsymbol{U}_1(\mu_i), \ldots, \boldsymbol{U}_{M_t}(\mu_i)$ for the training parameter μ_i with $i = 1, \ldots, M$. In the nested for loop, models are generated with PIR-OpInf (3.12) for all pairwise combinations of regularization parameters defined in (3.15) and training parameters $\mu_1, \ldots, \mu_M$. Then, the validation error (3.17) is computed and the index i^* of the regularization parameter λ_{i^*} that minimizes the validation error is determined. The corresponding inferred models are returned.

3.5 Numerical experiments

In this section, we compare operator inference with the proposed physics-informed regularizer (PIR-OpInf) to Tikhonov regularization and operator inference without regularization. Section 3.5.1 revisits the synthetic example from Section 3.2. Sections 3.5.2–3.5.5 show experiments with the Burgers' equation, a reaction-diffusion problem in a pipe, phase separation described by an Allen-Cahn model, and a FitzHugh-Nagumo model. The proposed approach depends on a small, positive threshold $\epsilon > 0$, e.g., for the post-processing in PIR-

Algorithm 1: Operator inference with physics-informed regularizer

Input: basis $\boldsymbol{V}$, inputs $\boldsymbol{U}_1(\mu_j), \ldots, \boldsymbol{U}_{M_t}(\mu_j)$ and trajectories $\boldsymbol{X}_1(\mu_j), \ldots, \boldsymbol{X}_{M_t}(\mu_j)$ for $j = 1, \ldots, M$

Output: inferred operators $\hat{\boldsymbol{A}}(\mu_j), \hat{\boldsymbol{B}}(\mu_j), \hat{\boldsymbol{F}}(\mu_j), \hat{\boldsymbol{G}}(\mu_j)$ for $j = 1, \ldots, M$

for $i = 1, \ldots, m$ **do**
  for $j = 1, \ldots, M$ **do**
    Infer operators $\hat{\boldsymbol{A}}^{(i)}(\mu_j), \hat{\boldsymbol{B}}^{(i)}(\mu_j), \hat{\boldsymbol{F}}^{(i)}(\mu_j), \hat{\boldsymbol{G}}^{(i)}(\mu_j)$ with PIR-OpInf (3.12) and regularization parameter λ_i defined in (3.15) and training parameter μ_j

Compute validation error (3.17) for $i = 1, \ldots, m$ and $j = 2, \ldots, M-1$

Pick $\lambda^* = \lambda_{i^*}$ with index i^* as in (3.18) that minimizes validation error

Set $\hat{\boldsymbol{A}}(\mu_j) = \hat{\boldsymbol{A}}^{(i^*)}(\mu_j), \hat{\boldsymbol{B}}(\mu_j) = \hat{\boldsymbol{B}}^{(i^*)}(\mu_j), \hat{\boldsymbol{F}}(\mu_j) = \hat{\boldsymbol{F}}^{(i^*)}(\mu_j), \hat{\boldsymbol{G}}(\mu_j) = \hat{\boldsymbol{G}}^{(i^*)}(\mu_j)$ for $j = 1, \ldots, M$

return $\hat{\boldsymbol{A}}(\mu_1), \ldots, \hat{\boldsymbol{A}}(\mu_M), \hat{\boldsymbol{B}}(\mu_1), \ldots, \hat{\boldsymbol{B}}(\mu_M), \hat{\boldsymbol{F}}(\mu_1), \ldots, \hat{\boldsymbol{F}}(\mu_M), \hat{\boldsymbol{G}}(\mu_1), \ldots, \hat{\boldsymbol{G}}(\mu_M)$

OpInf (cf. Section 3.3.4), which we set to $\epsilon = 10^{-10}$ in all of the following experiments.

3.5.1 Synthetic example

Consider again the synthetic example introduced in Section 3.2. We now apply PIR-OpInf with the parameter selection procedure discussed in Section 3.4.1. For each dimension $n \in \{2, 4, 6, 8, 10\}$, we sweep over $m = 51$ regularization parameters that are log-uniformly distributed in the interval $[10^{-15}, 10^5]$. The selected regularization parameters are $\lambda^* = 10^{-10}, 1.58 \times 10^{-7}, 10^{-8}, 3.98 \times 10^{-9}, 1.58 \times 10^{-9}$ for dimensions $n = 2, 4, 6, 8, 10$, respectively. We choose the test parameter set $\{\mu_1^{\text{test}}, \ldots, \mu_{M_{\text{test}}}^{\text{test}}\}$ of $M_{\text{test}} = 7$ test parameters that are equidistantly chosen in $\mathcal{D}$, where for each test parameter a test input trajectory is constructed with entries sampled uniformly in $[0, 10]$ and a test initial condition with entries sampled uniformly in $[0, 1]$, cf. Section 3.2.

Figure 3.2a shows the test error

$$e_{\text{test}} = \sum_{i=1}^{M_{\text{test}}} \frac{\|\boldsymbol{V}\bar{\boldsymbol{X}}^{\text{test}}(\mu_i^{\text{test}}) - \boldsymbol{X}^{\text{test}}(\mu_i^{\text{test}})\|_F}{\|\boldsymbol{X}^{\text{test}}(\mu_i^{\text{test}})\|_F}, \tag{3.22}$$

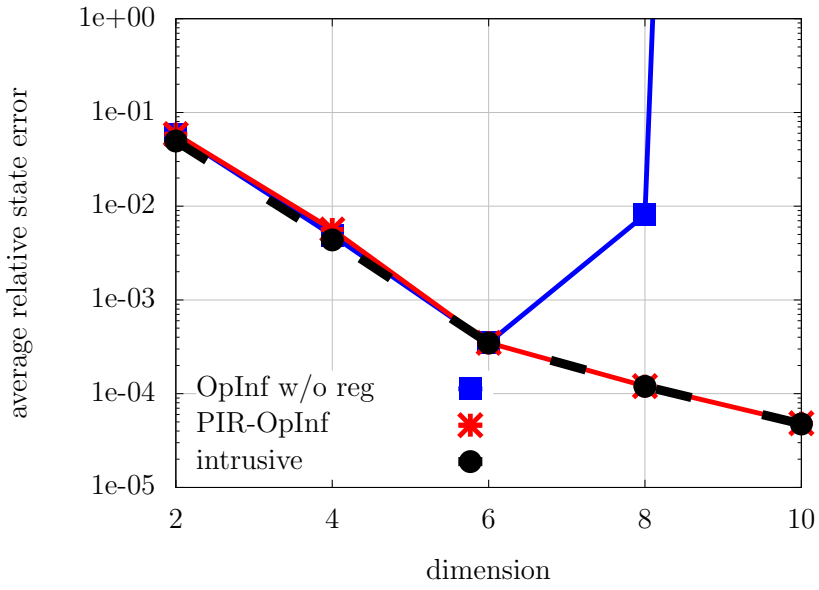
where $\bar{\boldsymbol{X}}^{\text{test}}(\mu_i^{\text{test}})$ is the trajectory obtained at test parameter μ_i^{test} with the corresponding test input trajectory and test initial condition with either PIR-OpInf, OpInf without regularization, or intrusive model reduction. In contrast to OpInf without regularization, PIR-OpInf shows stable behavior and yields accurate predictions even for dimensions $n > 6$ in this example. Figure 3.2b shows the stability radius $\hat{\rho}$ defined in (3.6) for PIR-OpInf, OpInf without regularization, and intrusive model reduction. To compute the bound (3.6) of the stability radius, we draw matrices $\boldsymbol{L}$ with entries uniformly distributed in $[0, 1]$ and repeat the calculation 100 times. We then show the mean and the 25% and 75% quantile in Figure 3.2b. The mean of the stability radius of the model obtained with PIR-OpInf is larger than the mean of the stability radius of OpInf without regularization, which numerically demonstrates that the proposed physics-informed regularizer indeed induces a stability bias.

3.5.2 Burgers' equation

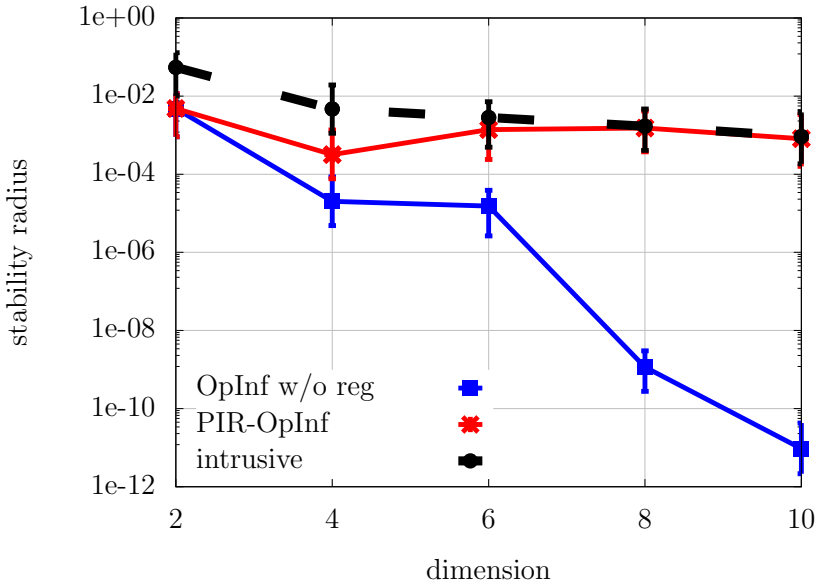
We consider the parameterized Burgers' equation

$$\frac{\partial x}{\partial t}(\omega, t; \mu) = \mu \frac{\partial^2 x}{\partial^2 \omega}(\omega, t; \mu) - x(\omega, t; \mu) \frac{\partial x}{\partial \omega}(\omega, t; \mu)$$

with spatial coordinate $\omega \in (0, 1)$, time $t \in [0, 1]$, and viscosity $\mu \in [10, 100]$. Dirichlet boundary conditions $x(0, t; \mu) = u(t)$, $x(1, t; \mu) = 0$ are imposed, with input $u : [0, 1] \to \mathbb{R}$. The equation is discretized in space with finite differences on an equidistant grid in $[0, 1]$ with $N = 128$ grid points. Time is discretized with the explicit Euler method with time-step size $\delta t = 10^{-4}$.



(a) test error (3.22)



(b) estimated stability radius

Figure 3.2: Synthetic example: The model obtained with the proposed PIR-OpInf shows stable behavior, in contrast to OpInf without regularization, and achieves a comparable test error as intrusive model reduction. The estimated stability radius (3.6) of the PIR-OpInf model is orders of magnitude larger than the estimated stability radius of the OpInf model without regularization, which is in agreement with the aim of the proposed regularizer to penalize models with low stability radii.

3.5.2.1 Problem setup

For each of the $M = 10$ training parameters $\mu = \{10, 20, 30, \ldots, 100\}$, we derive a single input trajectory $\boldsymbol{U}_1^{\mathrm{b}}(\mu)$, with entries uniformly sampled in $[0, 2]$, and an initial condition $\boldsymbol{x}_1(\mu) = \boldsymbol{0}$. Thus, $M_b = 1$. The corresponding state trajectories are $\boldsymbol{X}_1^{\mathrm{b}}(\mu_1), \ldots, \boldsymbol{X}_1^{\mathrm{b}}(\mu_M)$. A basis matrix $\boldsymbol{V} \in \mathbb{R}^{N \times n}$ is then constructed from the corresponding snapshots as described in Section 2.1. Furthermore, we sample $M_t = 10$ training inputs $\boldsymbol{U}_1(\mu), \ldots, \boldsymbol{U}_{M_t}(\mu)$ for each training parameter $\mu \in \{10, \ldots, 100\}$. To generate the initial conditions $\boldsymbol{x}_{1,0}(\mu), \ldots, \boldsymbol{x}_{M_t,0}(\mu)$ for each training parameter $\mu \in \{10, \ldots, 100\}$, we sample n-dimensional random vectors $\boldsymbol{r}_1(\mu), \ldots, \boldsymbol{r}_{M_t}(\mu)$ with independent entries uniformly distributed in $[0, 1]$ and set $\boldsymbol{x}_{i,0}(\mu) = \boldsymbol{V}\boldsymbol{r}_i(\mu)$ for $i = 1, \ldots, M_t$. We apply parameter selection as in Section 3.4.1 to find regularization parameters for each dimension $n \in \{2, \ldots, 10\}$. Furthermore, we construct an operator-inference model obtained without regularization and a reduced model with intrusive model reduction.

The process of learning the reduced model via PIR-OpInf requires multiple hours, and was done on NYU High Performance Computing clusters. The reduced model results in a considerable reduction in the evaluation time. The full model when iterated 20 times for a single training parameter, initial condition and input trajectory, results in the run time of 260 seconds, thus on average 13 seconds run time for generating a single state trajectory. The reduced model of dimension 10 when iterated 20 times for a single training parameter, projected initial condition and input trajectory, results in the run time of 4 seconds, thus on average 0.2 seconds run time for generating a single state trajectory. Overall, the reduced model of dimension 10 achieves ≈ 65 times speed up compared to the full model.

For comparison purposes, we also construct from the same training data an operator-inference model with the regularization proposed in [MHW21, SKHW20], which is Tikhonov regularization that regularizes the Frobenius norms of linear, quadratic, and input operators together, rather than only the norm of the quadratic operator as in the proposed PIR-OpInf.

We refer to this approach as T-OpInf in the following. The same regularization parameter-selection procedure is applied as for PIR-OpInf.

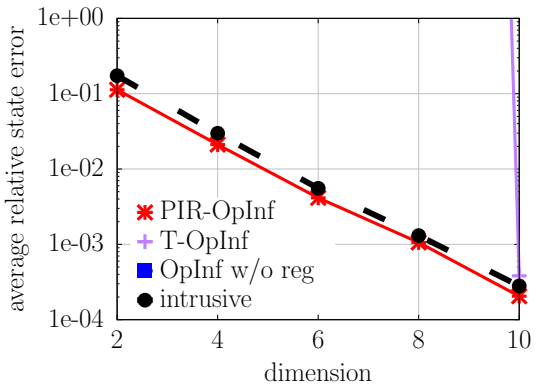
3.5.2.2 Results for PIR-OpInf

We test the models at $M_{\text{test}} = 7$ test parameters that are equidistantly distributed in the parameter domain $\mathcal{D}$. For each test parameter $\mu \in \{\mu_1^{\text{test}}, \dots, \mu_{M_{\text{test}}}^{\text{test}}\}$, we generate $M'_{\text{test}} = 5$ input trajectories $\boldsymbol{U}_1^{\text{test}}(\mu), \ \dots, \ \boldsymbol{U}_{M'_{\text{test}}}^{\text{test}}(\mu)$ and the corresponding test state trajectories $\boldsymbol{X}_1^{\text{test}}(\mu), \dots, \boldsymbol{X}_{M'_{\text{test}}}^{\text{test}}(\mu)$. The test initial conditions are the same as the training initial conditions, i.e, $\boldsymbol{x}_{i,0}^{\text{test}}(\mu) = \boldsymbol{x}_{i,0}(\mu)$, for $i = 1, \dots, M'_{\text{test}}$. The test error is then given by

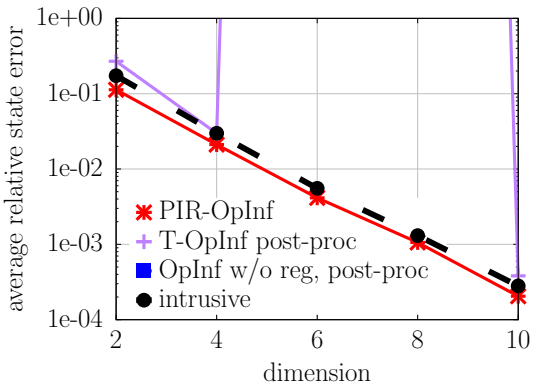
$$e_{\text{test}} = \sum_{i=1}^{M_{\text{test}}} \sum_{j=1}^{M'_{\text{test}}} \frac{\|\boldsymbol{V}\bar{\boldsymbol{X}}_j^{\text{test}}(\mu_i^{\text{test}}) - \boldsymbol{X}_j^{\text{test}}(\mu_i^{\text{test}})\|_F}{\|\boldsymbol{X}_j^{\text{test}}(\mu_i^{\text{test}})\|_F}, \tag{3.23}$$

where the trajectories $\bar{\boldsymbol{X}}_j^{\text{test}}(\mu_i^{\text{test}})$ for $i = 1, \dots, M_{\text{test}}$ and $j = 1, \dots, M'_{\text{test}}$ are obtained from either operator inference without regularization, PIR-OpInf, T-OpInf, or intrusive model reduction.

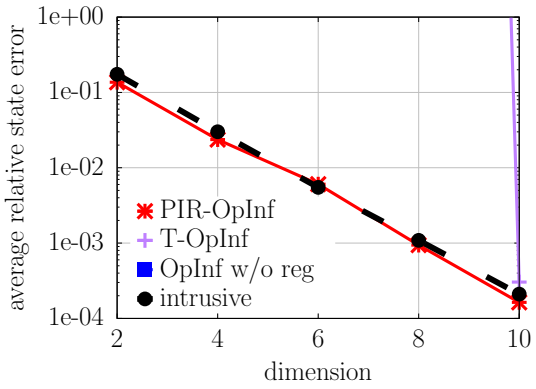
Figure 3.3a, Figure 3.3c, and Figure 3.4a show the test error (3.23) for test inputs with entries sampled uniform from the domains $[0, 2]$, $[0, 3]$, and $[0, 4]$, respectively. In all cases, PIR-OpInf shows stable behavior, whereas OpInf without regularization leads to numerical instabilities. Even T-OpInf with Tikhonov regularization shows unstable behavior for many dimensions n. To separate the effect of the regularization from the effect of the post-processing (cf. Section 3.3.4), we apply the same post-processing as for PIR-OpInf to T-OpInf. The corresponding results in Figure 3.3b, Figure 3.3d, and Figure 3.4b show that post-processing helps to stabilize T-OpInf as well; however, as the range of the inputs increases, a similarly unstable behavior as in the case without post-processing is obtained. Thus, the results indicate that penalizing the quadratic term via the proposed regularizer is responsible for achieving stabler models, rather than the post processing or penalizing



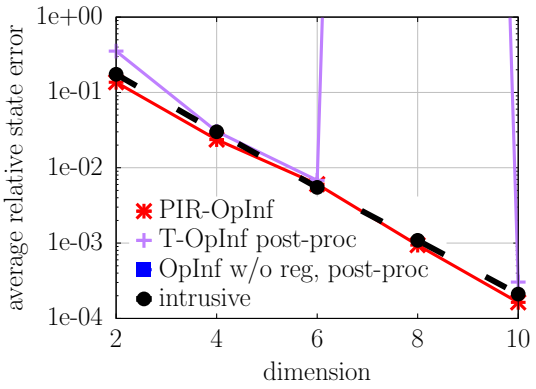
(a) input domain $(0, 2)$



(b) input domain $(0, 2)$, T-OpInf with post-processing

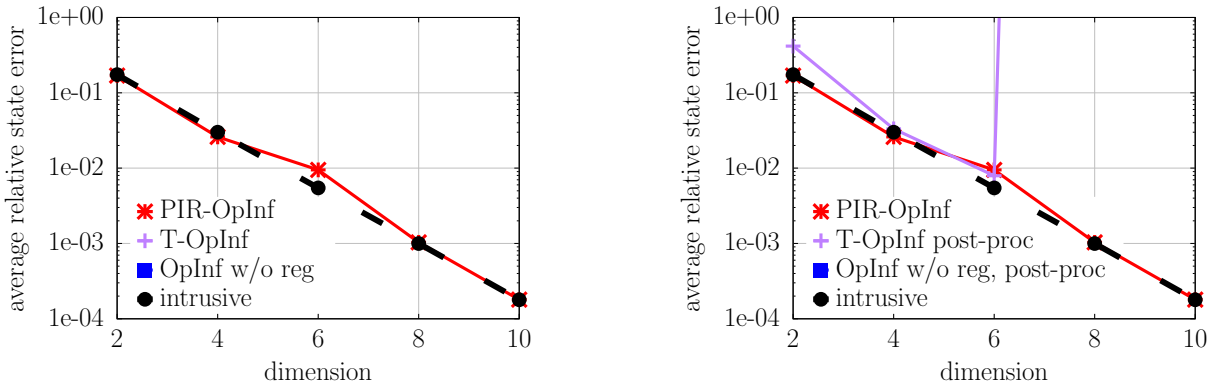


(c) input domain $(0, 3)$



(d) input domain $(0, 3)$, T-OpInf with post-processing

Figure 3.3: Burgers' equation: The proposed PIR-OpInf leads to models that are stable for a large range of inputs, which is in contrast to OpInf without regularization and OpInf with Tikhonov regularization (T-OpInf). The results also show that applying the same post-processing as for PIR-OpInf (cf. Section 3.3.4) to T-OpInf has little effect on the stability of the learned models, which indicates that indeed the proposed regularizer in PIR-OpInf is responsible for obtaining more stable models.



(a) input domain $(0, 4)$

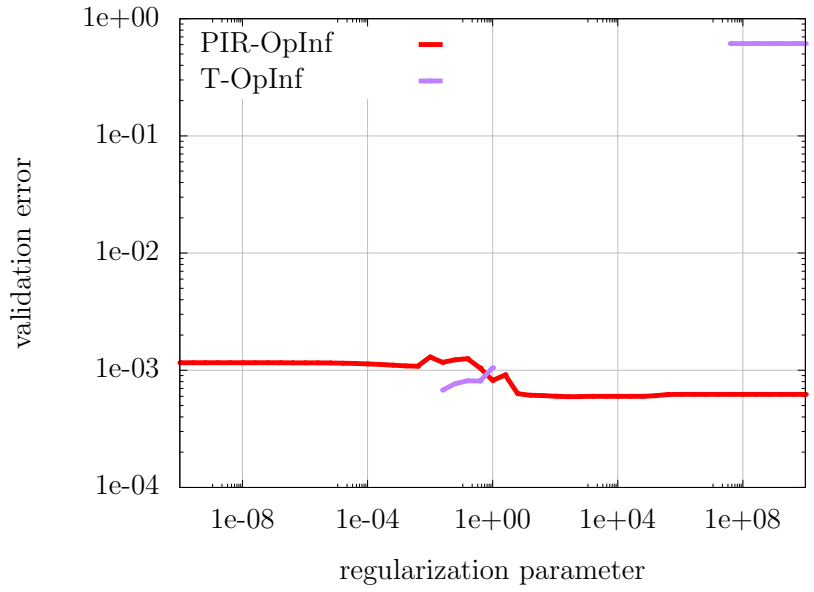
(b) input domain $(0, 4)$, T-OpInf with post-processing

Figure 3.4: Burgers' equation: Similar results observed for input domain $(0, 4)$, as for input domains $(0, 2)$ and $(0, 3)$, as seen in Figure 3.3

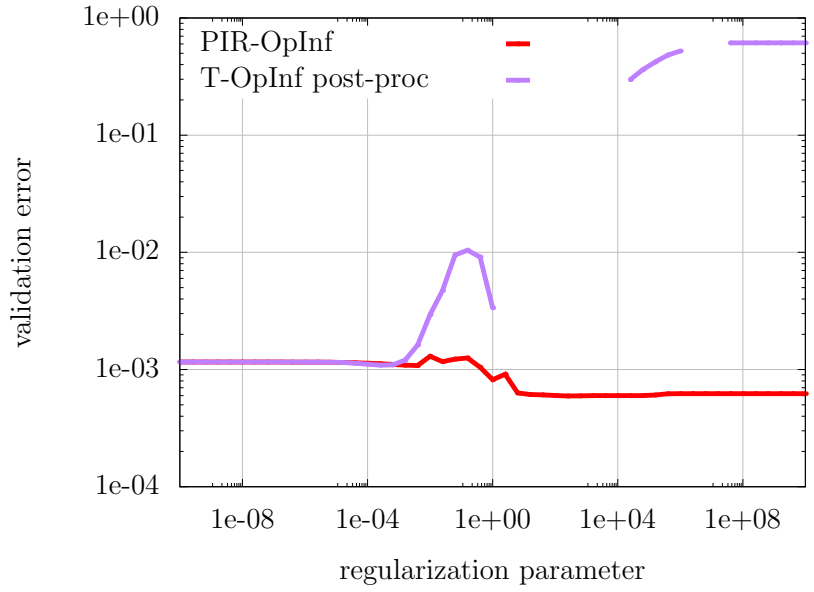
both the linear and the quadratic term together as in Tikhonov regularization, which is in agreement with the theoretical motivation outlined in Section 3.3.1.

Consider now Figure 3.5 that shows the validation error, i.e., the objective of (3.18), of the parameter-selection procedure versus the regularization parameter for dimension $n = 8$ for PIR-OpInf and T-OpInf. Independent of whether post-processing is applied to T-OpInf (Figure 3.5b) or not (Figure 3.5a), the error of T-OpInf grows quickly as the regularization parameter is increased. Thus, if a small regularization parameter is chosen, the validation error of T-OpInf is small but it also means that no regularization is induced. If instead the regularization parameter is large, then there might be a stability bias but at the same time it leads to a distinct increase of the model error. In contrast, the curves corresponding to PIR-OpInf show that the validation error is small for moderately sized regularization parameters, where a stability bias is induced without leading to a deterioration of the model accuracy.

Figure 3.6a shows the estimated stability radius (3.6) for various models. The results indicate that PIR-OpInf achieves a larger stability radius than the models obtained with T-OpInf and OpInf without regularization. At dimension $n = 2$, the stability radius of PIR-OpInf is large because only large regularization parameters lead to stable behavior; see

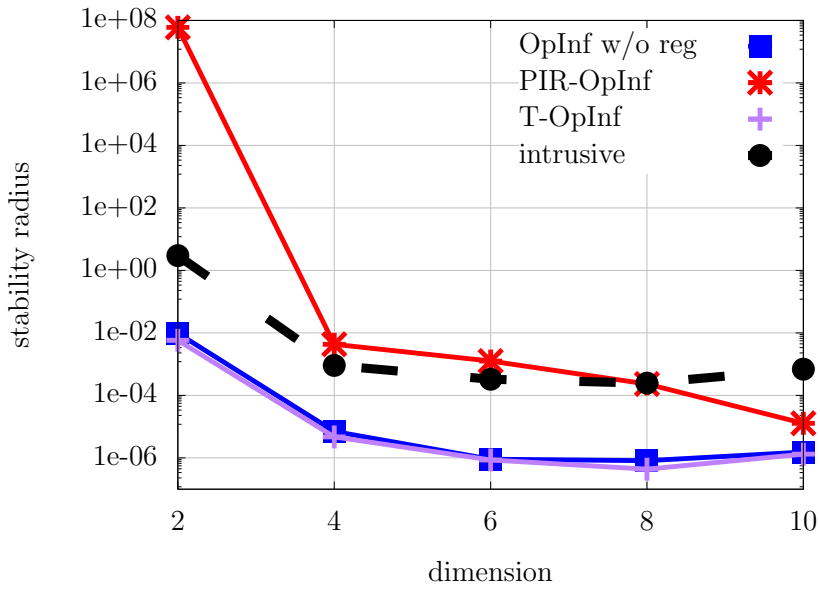


(a) T-OpInf without post-processing

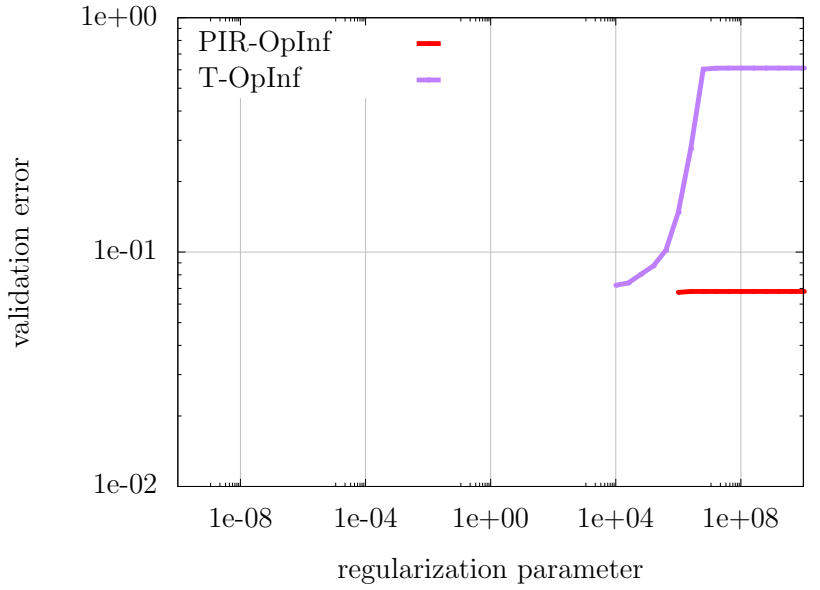


(b) T-OpInf with post-processing

Figure 3.5: Burgers' equation: The validation error (objective of (3.18)) of T-OpInf grows quickly with the regularization parameter, which means that the regularization has a negative effect on the model accuracy. In contrast, the validation error corresponding to the proposed PIR-OpInf is less sensitive to the regularization parameter, which means that large regularization parameters can be chosen—imposing a stronger stability bias—without deteriorating the model accuracy.



(a) estimated stability radius



(b) validation error for $n = 2$

Figure 3.6: Burgers' equation: The models learned with the proposed PIR-OpInf have larger estimated stability radii than models obtained without regularization and with Tikhonov regularization (T-OpInf). The stability radius for $n = 2$ is high for PIR-OpInf models because only large regularization parameters $\lambda^* = 10^6$ lead to stable behavior (see (b)), which forces the norm of the quadratic term to be close to zero and thus increases the stability radius by a large amount. Notice that the PIR-OpInf model at $n = 2$ achieves a similar accuracy as intrusive model reduction; cf. Figure 3.3.

Figure 3.6b. The regularization parameter is chosen so large that the norm of the quadratic term is close to zero, which explains the high estimated stability radius.

3.5.3 Reaction-diffusion problem

Consider the parameterized reaction-diffusion equation

$$\frac{\partial}{\partial t}x(\boldsymbol{\xi}, t; \mu) = \Delta x(\boldsymbol{\xi}, t; \mu) + s(\boldsymbol{\xi})u(t) + g(x(\boldsymbol{\xi}, t; \mu)), \tag{3.24}$$

with the spatial coordinate $\boldsymbol{\xi} = [\xi_1 \ \xi_2]^T \in [0, 1]^2$. We impose homogeneous Neumann boundary conditions. The parameter domain is $\mathcal{D} = [1, 1.5]$ and end time is $T = 20$. The source is $s(\boldsymbol{\xi}) = 10^{-1} \sin(2\pi\xi_1) \sin(2\pi\xi_2)$. The nonlinear term is

$$g(x(\boldsymbol{\xi}, t; \mu)) = -(a \sin(\mu) + 2) \exp(-\mu^2 b)\left(1 + (\mu c)x + \frac{(\mu c)^2}{2!}x^2\right)$$

which is the second-order Taylor approximation of the source term used in [Peh20b], with $a = 0.1, b = 2.7$ and $c = 1.8$. We discretize in space with a mesh width of $h = 1/12$ and finite difference and in time with explicit Euler with time-step size $\delta t = 10^{-2}$. The dimension of the high-dimensional model is $N = 144$.

The training parameter set contains $M = 10$ equidistant points in the parameter domain $\mathcal{D}$. To construct the reduced space, we take a single $M_b = 1$ input trajectory for each training parameter, where the inputs are sampled uniformly in $[0, 1]$. The initial condition is zero. The corresponding trajectories $\boldsymbol{X}_1^{\mathrm{b}}(\mu_1), \ldots, \boldsymbol{X}_1^{\mathrm{b}}(\mu_M)$ are used to construct a POD basis. For each of the M training parameters, we sample $M_t = 10$ input trajectories with entries uniformly in $[0, 1]$. The regularization parameters are selected via our selection procedure described in Section 3.4.1 by sweeping over the 51 logarithmically equidistant points in $[10^{-10}, 10^{10}]$. We test models for $M_{\mathrm{test}} = 7$ test parameters that are equidistantly distributed

in the parameter domain $\mathcal{D}$. For every test parameter $\mu_1^{\text{test}}, \ldots, \mu_{M_{\text{test}}}^{\text{test}}$, we generate a single input trajectory $\boldsymbol{U}^{\text{test}}$, whose entries randomly selected via a uniform distribution in $[0, 1]$, and initial condition $\boldsymbol{x}_0^{\text{test}} = \mathbf{0}$. The test error is then

$$e_{\text{test}} = \sum_{i=1}^{M_{\text{test}}} \frac{\|\boldsymbol{V}\bar{\boldsymbol{X}}^{\text{test}}(\mu_i^{\text{test}}) - \boldsymbol{X}^{\text{test}}(\mu_i^{\text{test}})\|_F}{\|\boldsymbol{X}^{\text{test}}(\mu_i^{\text{test}})\|_F}\,, \tag{3.25}$$

where $\bar{\boldsymbol{X}}^{\text{test}}(\mu_i^{\text{test}})$ is the predicted trajectory at parameter μ_i^{test} by either the PIR-OpInf, T-OpInf, OpInf without regularization, or intrusive model reduction.

Figure 3.7a shows the test error (3.25). The results indicate that OpInf without any regularization becomes unstable quickly. In contrast, T-OpInf and PIR-OpInf provide stable approximations. However, whereas the Tikhonov regularization in T-OpInf leads to a loss of accuracy at higher dimensions, the proposed physics-informed regularizer used by PIR-OpInf achieves errors that are comparable to intrusive model reduction. This is in agreement with the estimated stability radius shown in Figure 3.7b, where PIR-OpInf achieves an orders of magnitude larger stability radius at higher dimensions n than T-OpInf and OpInf without regularization.

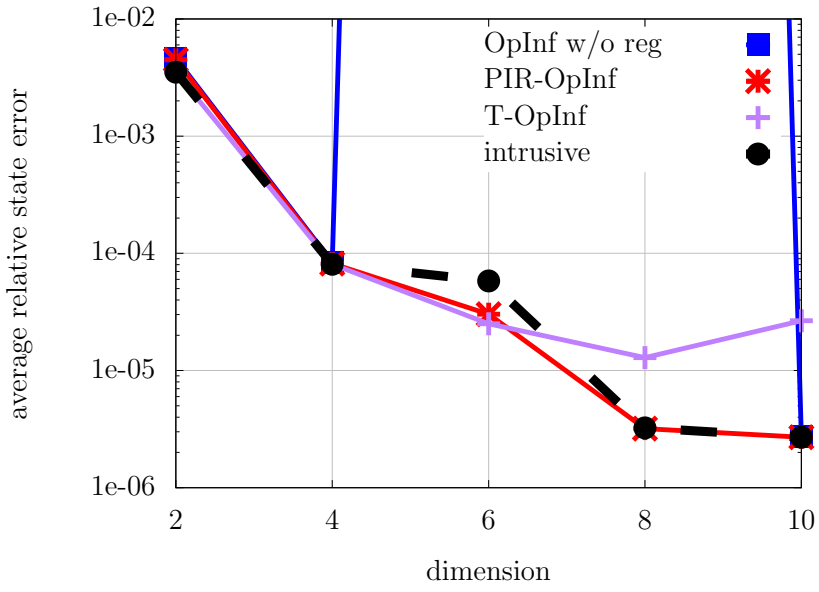
3.5.4 Phase separation described by Allen-Cahn equation

Consider the cubic nonlinear Allen-Cahn model that is used, for example, for describing phase transitions:

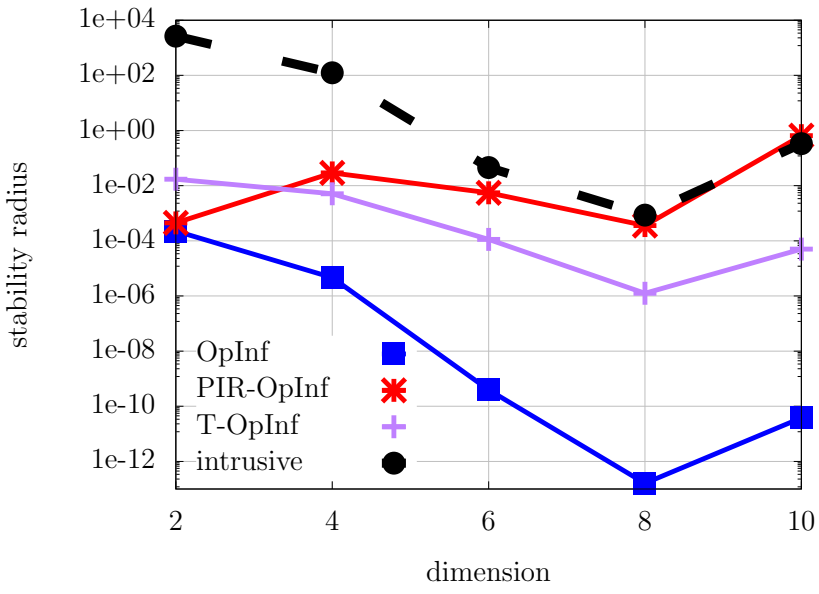
$$\frac{\partial}{\partial t}x(\omega, t) = \frac{\partial^2}{\partial^2 \omega}x(\omega, t) - x(\omega, t)^3 + u(t)\,, \tag{3.26}$$

with spatial coordinate $\omega \in (0, 1)$ and time $t \in [0, 0.1]$. The boundary conditions are

$$x(0, t) = u(t), \quad \frac{\partial x}{\partial t}(1, t) = 0\,,$$



(a) test error (3.25)



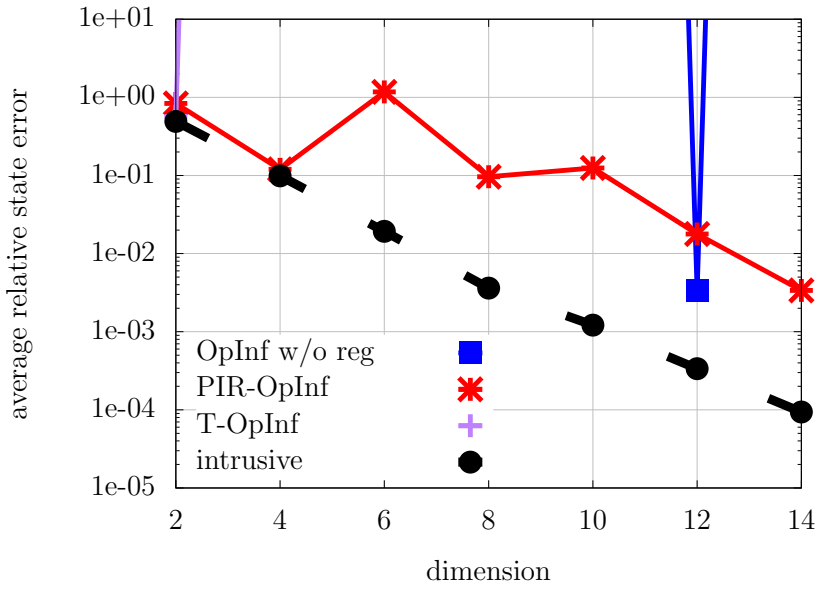
(b) estimated stability radius (3.6)

Figure 3.7: Reaction-diffusion problem: The PIR-OpInf model shows stable behavior in this experiment. In contrast to Tikhonov regularization (T-OpInf), the PIR-OpInf model achieves an accuracy close to intrusive model reduction even for larger $n > 6$ dimensions. The estimated stability radius of the PIR-OpInf model is orders of magnitude higher than the stability radius of the T-OpInf model in this experiment.

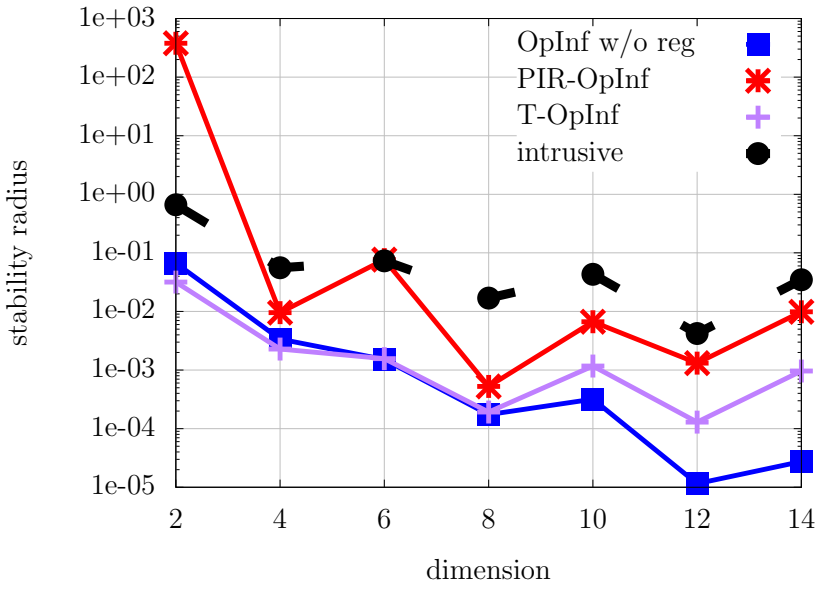
with input $u(t) : [0, 1] \to \mathbb{R}$. The equation is discretized in space with finite differences on an equidistant grid in $[0, 1]$ with $N = 128$ grid points. Time is discretized with the explicit Euler method with time-step size $\delta t = 10^{-5}$.

To construct the basis of a reduced space, we take a single $M_b = 1$ input trajectory, where the inputs are sampled uniformly in $[0, 10]$. The initial condition is zero. The corresponding trajectory $\boldsymbol{X}^b$ is used to construct a POD basis. For training the model, we use $M_t = 10$ input trajectories, with entries uniformly sampled in [0, 10]. The regularization parameters are selected via our selection procedure described in Section 3.4.2 by sweeping over the 51 logarithmically equidistant points in $[10^{-10}, 10^{10}]$, where the validation data consists of the training trajectories and the trajectory obtained with input $\boldsymbol{U}^{\text{val}} = 10(\sin(\pi t)+1), t \in [0, 0.1]$. We test models for the test input trajectory $\boldsymbol{U}^{\text{test}} = 25(\sin(\pi t) + 1), t \in [0, 0.1]$, which is motivated by the work [BB15]. The reduced model is simulated for the test input and the generated state trajectory is compared against the corresponding full model trajectory.

Figure 3.8a shows the test error (3.25). The results indicate that OpInf without any regularization is unstable for all dimensions. T-OpInf is unstable for all dimensions except dimension 12, whereas PIR-OpInf provides a stable solution for all dimensions, except the smallest dimension 2. Our parameter selection scheme for PIR-OpInf selects the regularization parameters $\lambda^* = 10^{12}, 1.58 \times 10^9, 1.58 \times 10^{10}, 3.98 \times 10^5, 3.98 \times 10^6, 3.98 \times 10^6, 2.51 \times 10^5$ for dimensions $n = 2, 4, 6, 8, 10, 12, 14$, respectively. The selected regularization parameters show that a strong regularization is necessary in this example. At the same time, the large values of the selected regularization parameter results in PIR-OpInf result in a higher test error compared to the intrusive model. As seen in Figure 3.8b, PIR-OpInf achieves a larger stability radius at all dimensions n than both T-OpInf and OpInf without regularization.



(a) test error (3.25)



(b) estimated stability radius (3.6)

Figure 3.8: Allen-Cahn model: The PIR-OpInf model shows stable behavior in this experiment, whereas Tikhonov regularization is unstable for all dimensions except 2. The estimated stability radius of the PIR-OpInf model is higher than the stability radius of the T-OpInf model.

3.5.5 FitzHugh-Nagumo equation

Consider the FitzHugh-Nagumo equation with quadratic and cubic nonlinear terms:

$$\begin{aligned}\epsilon\frac{\partial v}{\partial t}(\omega,t) &= \epsilon^2\nabla^2 v(\omega,t) - v(\omega,t)(v(\omega,t)-R)(1-v(\omega,t)) - w(\omega,t) + c,\\ \frac{\partial w}{\partial t}(\omega,t) &= bv(\omega,t) - \gamma w(\omega,t) + c\end{aligned} \tag{3.27}$$

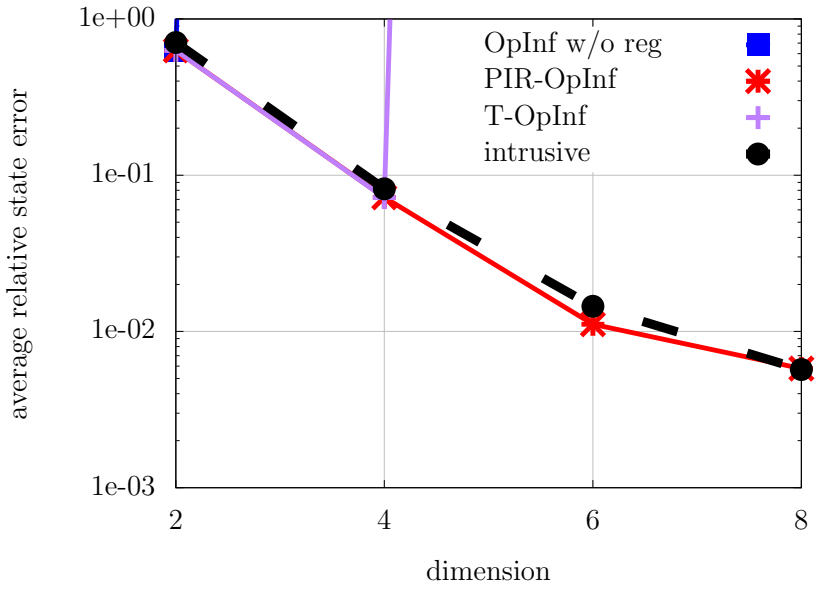
with spatial coordinate $\omega \in (0,1)$, time $t \in [0,8]$. The equation is discretized in space with finite differences on an equidistant grid with $N = 128$ grid points for both $v(\omega,t)$ and $w(\omega,t)$. In time, we use as discretization the the semi-implicit Euler method with time-step size $\delta t = 10^{-3}$. The parameter values for FitzHugh-Nagumo equation are $\epsilon = 0.05, R = 1, c = 0.05, b = 5$ and $\gamma = 20$. Parameter values are not varied when learning the model but different initial conditions (see below) and different inputs are used that enter via the boundary conditions:

$$v_\omega(0,t) = -i_0 t^3 \exp(-i_1 t),\ v_\omega(1,t) = 0, \qquad t \geq 0\,,$$

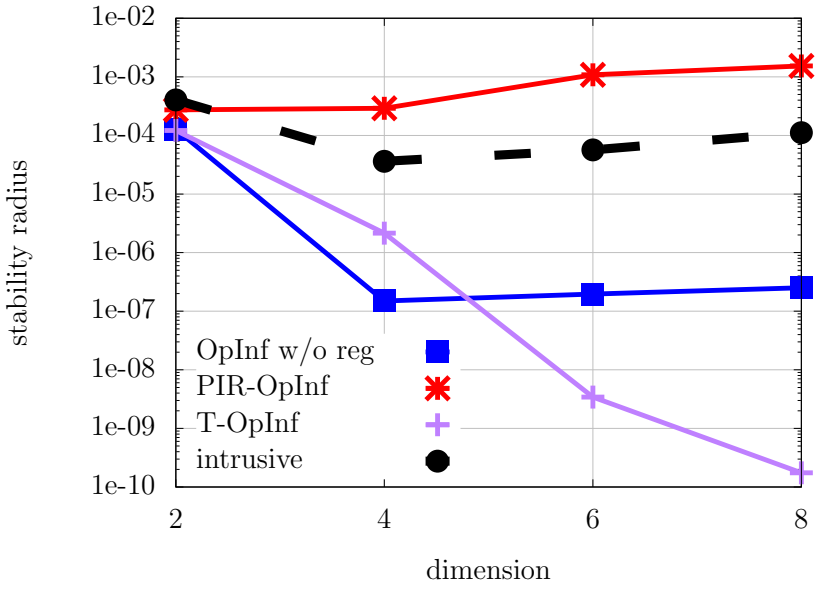
where i_0 is uniformly sampled in $[49500, 50500]$, and i_1 is uniformly sampled in $[14,16]$, when generating trajectories for basis generation and training, while constant values $i_0 = 50000, i_1 = 15$ is used for testing.

To construct the reduced space, we take $M_b = 5$ input trajectories, and zero initial condition. The corresponding trajectory $\boldsymbol{X}^b$ is used to construct a POD basis. For training the model, we use $M_t = 10$ input trajectories, and initial conditions with entries uniformly sampled in $[-0.1, 0.1]$. The regularization parameters are selected via our selection procedure described in Section 3.4.2 by sweeping over the 51 logarithmically equidistant points in $[10^{-10}, 10^{10}]$. We test models for the test input trajectory given above and zero initial condition, which is different from the training initial conditions. The reduced model is simulated

for the test input and the generated state trajectory is compared against the corresponding full model trajectory. Figure 3.9a shows the test error (3.25). The results indicate that OpInf without any regularization is unstable for all dimensions > 2. For PIR-OpInf, the solution is stable for all dimensions, and the test error is close to the error of the intrusive model. T-OpInf is unstable for dimensions > 4. Figure 3.9b shows that among all of the inferred models, PIR-OpInf gives the model with the largest stability radius. For dimensions 6 and 8, our parameter selection scheme selects regularization parameters $\lambda^* = 10^2, 10$ for PIR-OpInf and $\lambda^* = 10^8, 1.58 \times 10^9$ for T-OpInf. T-OpInf regularizes all parameters with a large regularization parameter, which results in a model with a small $\|\hat{\boldsymbol{A}}\|_F$. As the equation (3.5) computes the Lyapunov matrix $\boldsymbol{P}$, a model with a small $\|\hat{\boldsymbol{A}}\|_F$ leads to a model with a large $\|\boldsymbol{P}\|_F$. A large $\|\boldsymbol{P}\|_F$ results in a smaller $\hat{\rho}$ (3.10), and consequently, a less stable model, which is in agreement with the numerical results shown in Figure 3.9b.



(a) test error (3.25)



(b) estimated stability radius (3.6)

Figure 3.9: FitzHugh-Nagumo: PIR-OpInf learns stable models, while OpInf without regularization gives unstable models for dimensions > 2, and T-OpInf is unstable for dimensions > 4. Of all the inferred models, PIR-OpInf learns the reduced model with the largest estimated stability radius ($\hat{\rho}$), and for dimensions 6 and 8, $\hat{\rho}$ for PIR-OpInf is considerably larger compared to the models learned via T-OpInf.

Chapter 4

Operator inference with structure preservation

We add constraints to the optimization problem solved by operator inference with physics-informed regularizer (PIR-OpInf), to preserve the underlying structure of full model in the reduced model. Section 4.1 proposes the structure preservation constraints for operator inference. The computational procedure is discussed in Section 4.2. Numerical results in Section 4.3 demonstrate that operator inference with the proposed regularizer and structure preservations learns stable models even when Tikhonov regularization and models learned without regularization are unstable.

4.1 Operator inference with structure preservation

Structure can be imposed on the linear operator by adding hard constraints to the operator inference problem. We focus on problems that lead to symmetric negative definite linear operators, which are present in, e.g., Hamiltonian systems [SWK22]. We consider the constrained problem

$$\begin{aligned} \min_{\hat{\boldsymbol{A}},\hat{\boldsymbol{B}},\hat{\boldsymbol{F}},\hat{\boldsymbol{G}}} \quad & J(\hat{\boldsymbol{A}},\hat{\boldsymbol{B}},\hat{\boldsymbol{F}},\hat{\boldsymbol{G}}) + \lambda(\|\hat{\boldsymbol{F}}\|_F^2 + \|\hat{\boldsymbol{G}}\|_F)\,, \\ \text{such that} \quad & \hat{\boldsymbol{A}} + \epsilon\boldsymbol{I} \preceq 0\,, \end{aligned} \tag{4.1}$$

where $\hat{\boldsymbol{A}} + \epsilon\boldsymbol{I} \preceq 0$ means that $\hat{\boldsymbol{A}} + \epsilon\boldsymbol{I}$ is symmetric negative semi-definite. The matrix $\boldsymbol{I}$ is the identity and $\epsilon > 0$ is a margin that guarantees that $\hat{\boldsymbol{A}}$ is definite, rather than semi-definite. We refer to (4.1) as the SPIR-OpInf problem, where the S stands for "structure." Problem (4.1) is in the class of semi-definite programs, which typically are computationally more expensive to solve than linear least-squares problems as (3.12) with the same number of unknowns. However, efficient numerical algorithms and software exist for solving semi-definite programs [Tod01]. Additionally, we are seeking low-dimensional models that have few degrees of freedom, which means that we focus mostly on optimization problems with a manageable number of unknowns. We solve the optimization problem for PIR-OpInf using the SDPT3 solver of the CVX toolbox [GB14, GB08] in MATLAB, with the $\hat{\boldsymbol{A}} + \epsilon\boldsymbol{I} \preceq 0$ constraint added to the CVX optimization problem.

Instead of imposing a margin ϵ to guarantee definiteness of $\hat{\boldsymbol{A}}$ in (4.1), one can solve problem (4.1) with the constraint $\hat{\boldsymbol{A}} \preceq 0$ and subsequently apply an analogous post-processing step as in Section 3.3.4. Because symmetry is enforced by $\hat{\boldsymbol{A}} \preceq 0$, it is guaranteed that $\hat{\boldsymbol{A}}$ is diagonalizable. The post-processing described in Section 3.3.4 preserves symmetry and thus the result is a symmetric negative definite matrix after the post-processing; see also [Hig88]. However, in the following, we will impose a margin ϵ and therefore do not need a

post-processing step.

Other structures in the linear operator can be preserved in an analogous way. For example, another common structure is skew-symmetry of $\hat{\boldsymbol{A}}$ which can be formulated as a linear constraint; we leave such other constraints to future work. Recall that it is required to interpolate between inferred operators when a model at a parameter μ outside of the training set is required; cf. Section 2.2.1. In case of structure-preserving operator inference, the corresponding operator interpolation schemes have to preserve the operator structure. We discuss such an interpolation scheme for symmetric negative definite matrices in Section 4.2.

4.2 Computational procedure of physics-informed operator inference

In this section, we briefly recapitulate an interpolation scheme that preserves symmetric definiteness of matrices, which is critical for constructing operators at new parameters outside of the training set in SPIR-OpInf.

In SPIR-OpInf introduced in Section 4.1, the definiteness and symmetry constraints in the optimization problem (4.1) ensure that for each training parameter $\mu_1, \ldots, \mu_M$ a model is obtained with a linear operator that is symmetric negative definite. When we interpolate the trained models at a new parameter $\mu \in \mathcal{D} \setminus \{\mu_1, \ldots, \mu_M\}$ outside of the training set, however, we have to ensure that the interpolated linear operator is symmetric negative definite as well. There are various interpolation schemes in model reduction that preserve such structure, see, e.g., [DVW10, AF08]. We build on the Log-Cholesky averaging method presented in [Lin19].

Given are M symmetric negative definite matrices $\hat{\boldsymbol{A}}(\mu_1), \ldots, \hat{\boldsymbol{A}}(\mu_M)$ at parameters $\mu_1, \ldots, \mu_M$. We compute the Cholesky factors $\hat{\boldsymbol{L}}(\mu_i)$ such that $\hat{\boldsymbol{A}}(\mu_i) = -\hat{\boldsymbol{L}}(\mu_i)\hat{\boldsymbol{L}}(\mu_i)^T$

for $i = 1, \ldots, M$. The Cholesky factors $\hat{\boldsymbol{L}}(\mu_i)$ are then split into

$$\hat{\boldsymbol{L}}(\mu_i) = \lfloor \hat{\boldsymbol{L}}(\mu_i) \rfloor + \operatorname{diag}(\hat{\boldsymbol{L}}(\mu_i)), \qquad i = 1, \ldots, M,$$

where $\operatorname{diag}(\hat{\boldsymbol{L}}(\mu_i))$ is the diagonal matrix with the same the diagonal as $\hat{\boldsymbol{L}}(\mu_i)$ and $\lfloor \hat{\boldsymbol{L}}(\mu_i) \rfloor$ is its remaining strictly lower triangular part. The interpolated matrix $\hat{\boldsymbol{A}}(\mu)$ at a new parameter μ is

$$\hat{\boldsymbol{A}}(\mu) = -\hat{\boldsymbol{L}}(\mu)\hat{\boldsymbol{L}}(\mu)^T,$$

where the Cholesky factor $\hat{\boldsymbol{L}}(\mu)$ is obtained as

$$\begin{aligned}\hat{\boldsymbol{L}}(\mu) =& \mathcal{I}(\mu; \lfloor \hat{\boldsymbol{L}}(\mu_1) \rfloor, \ldots, \lfloor \hat{\boldsymbol{L}}(\mu_M) \rfloor) + \\ & \exp\left(\mathcal{I}\left(\mu; \log\left(\operatorname{diag}\left(\hat{\boldsymbol{L}}(\mu_1)\right)\right), \ldots, \log\left(\operatorname{diag}\left(\hat{\boldsymbol{L}}(\mu_M)\right)\right)\right)\right).\end{aligned}$$

The operator $\mathcal{I}$ denotes linear interpolation of the matrix entries at μ and $\exp(\cdot)$ and $\log(\cdot)$ are the matrix exponential and logarithm, respectively.

The optimal regularization parameter for SPIR-OpInf is selected in the same way as PIR-OpInf as discussed in sections 3.4.1 and 3.4.2.

4.3 Numerical experiments

In this section, we compare operator inference with the proposed physics-informed regularizer and structure preservation (SPIR-OpInf) to operator inference with the proposed physics-informed regularizer without structure preservation (PIR-OpInf) and Tikhonov regularization. Sections 4.3.1 and 4.3.2 revisit the Burgers' equation and Reaction-diffusion problem from Sections 3.5.2 and 3.5.3, respectively. The proposed approach depends on a small, positive threshold $\epsilon > 0$, e.g., for the margin in SPIR-OpInf (4.1), which we set to

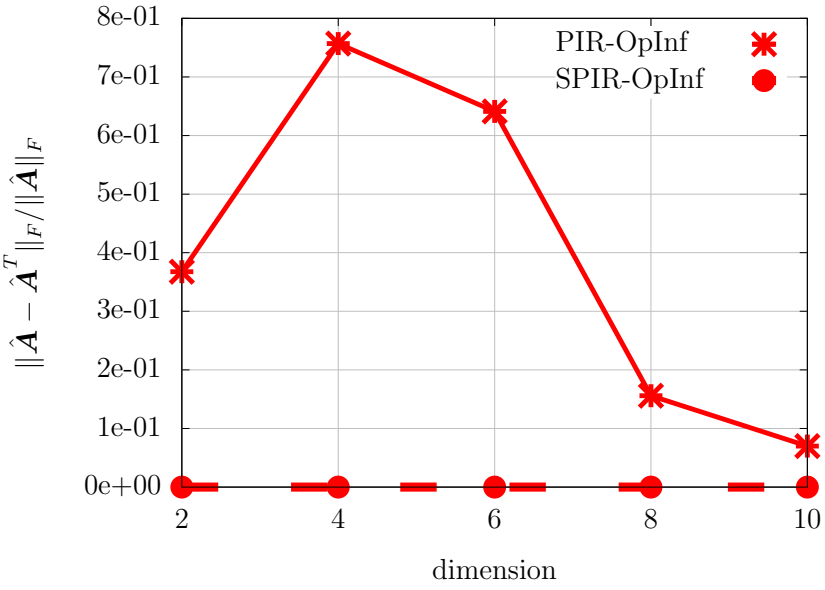
$\epsilon = 10^{-10}$ in all of the following experiments.

4.3.1 Burgers' equation

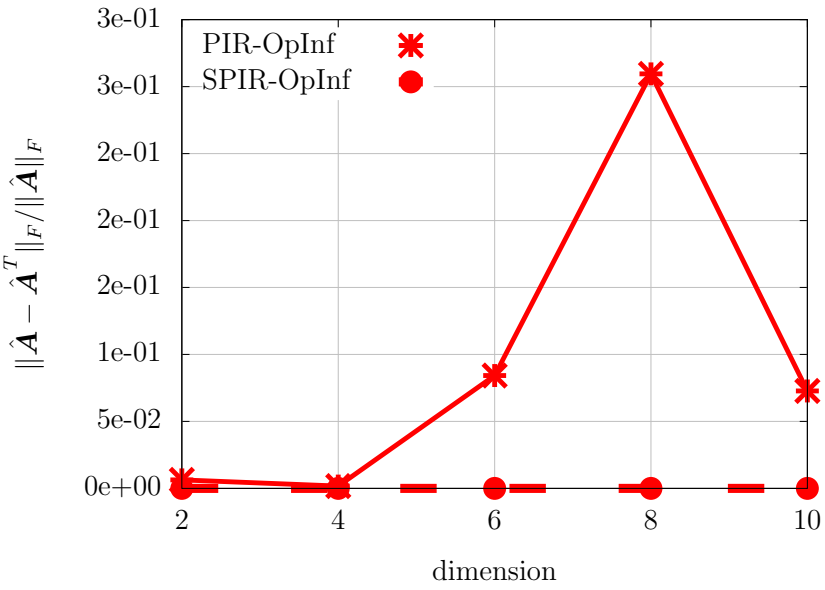
We consider again Burgers' equation with same problem setup as in Section 3.5.2. We now apply SPIR-OpInf to learn a symmetric and (positive or negative) definite linear operator, which is in contrast to PIR-OpInf that does not impose any structure on the linear operator; see Figure 4.1a.

Figure 4.2 shows the test error (3.23) corresponding to SPIR-OpInf. Stable behavior is obtained in all cases; however, a leveling off of the error as the dimension increases indicates that restricting to symmetric operators in this example is limiting the accuracy. Operator inference, independent of the used regularizer, is fitting operators to projected trajectories of the high-dimensional systems. Projected trajectories are different from reduced trajectories that are generated with the corresponding reduced model obtained with intrusive model reduction; see [Peh20b]. The difference between the projected and reduced trajectories is the closure error. This means that operator inference aims to find operators that well predict the projected trajectories, and the best operators (in the sense of the objective of operator inference) do not necessarily have the same structure as the intrusive operators. Thus, allowing operator inference to break structure can help in terms of accuracy, as can be seen by comparing Figure 3.3 and 3.4 with Figure 4.2. Additional constraints such as symmetry that are imposed by SPIR-OpInf on the linear operator mean that the optimization search space is smaller and thus SPIR-OpInf can lead to models that have a larger error than having no structure as in PIR-OpInf. However, breaking structure also means that predictions can become unphysical. Preventing such unphysical behavior is the motivation for structure preservation with SPIR-OpInf.

The estimated stability radii of the SPIR-OpInf models are compared to the stability radii of T-OpInf models in Figure 4.3a. For large dimensions $n > 4$, the estimated stability



(a) Burgers' equation



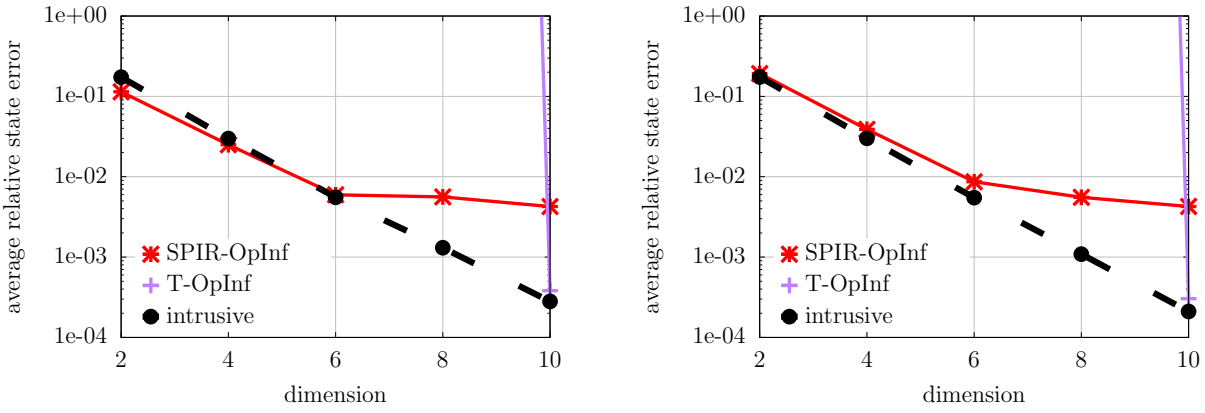
(b) reaction-diffusion problem

Figure 4.1: Structure such as symmetry in the matrix $\hat{\boldsymbol{A}}$ can be imposed via SPIR-OpInf.

radii of the SPIR-OpInf models is larger than the stability radii of the T-OpInf models. For small dimensions $n \leq 4$, the stability radii of SPIR-OpInf and T-OpInf models is large. This is reflected by the small regularization parameter selected by the proposed parameter selection procedure, which selects $\lambda^* \approx 3 \times 10^{-7}$ for $n = 2$.

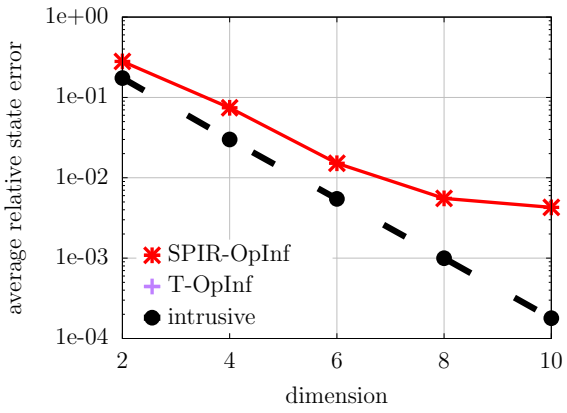
4.3.2 Reaction-diffusion equation

We consider again Reaction-diffusion problem as in Section 3.5.3. We now apply SPIR-OpInf to learn a symmetric and (positive or negative) definite linear operator, which is in contrast to PIR-OpInf that does not impose any structure on the linear operator; see Figure 4.1b. Figure 4.4a shows the test error (3.25). While T-OpInf and SPIR-OpInf both provide stable approximations, Tikhonov regularization in T-OpInf leads to a loss of accuracy at higher dimensions, whereas SPIR-OpInf achieves errors that are comparable to intrusive model reduction. As seen in figure 4.4b, SPIR-OpInf achieves an orders of magnitude larger stability radius at higher dimensions n than T-OpInf.



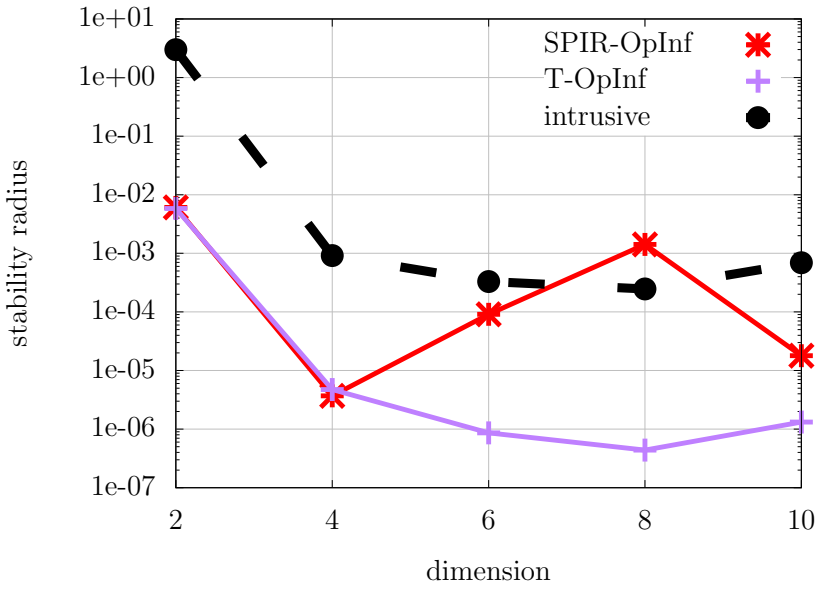
(a) input domain $(0, 2)$

(b) input domain $(0, 3)$

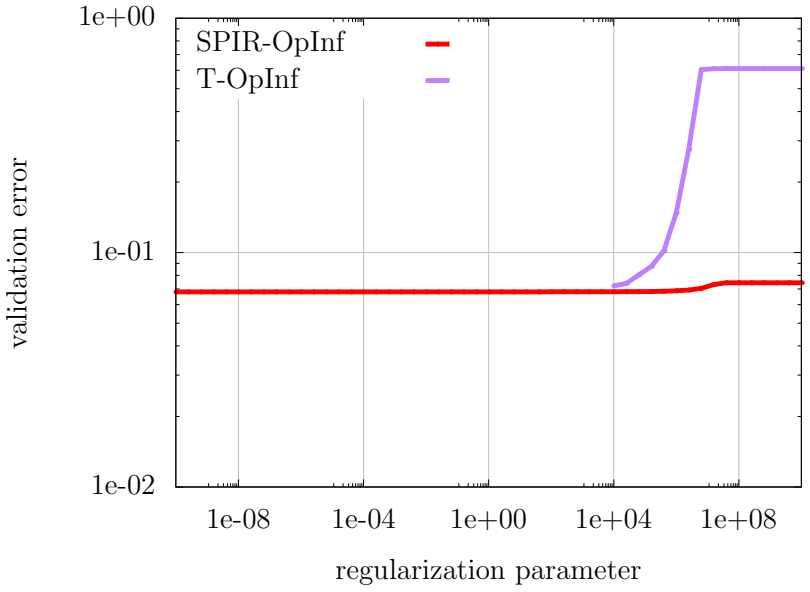


(c) input domain $(0, 4)$

Figure 4.2: Burgers' equation: Imposing symmetry and definiteness onto the linear operator with the proposed SPIR-OpInf (4.1) leads to stable models in this experiment; however, the additional constraints lead to a lower accuracy than PIR-OpInf that include the proposed regularizer but no constraints on the linear operator (cf. Figure 3.3).

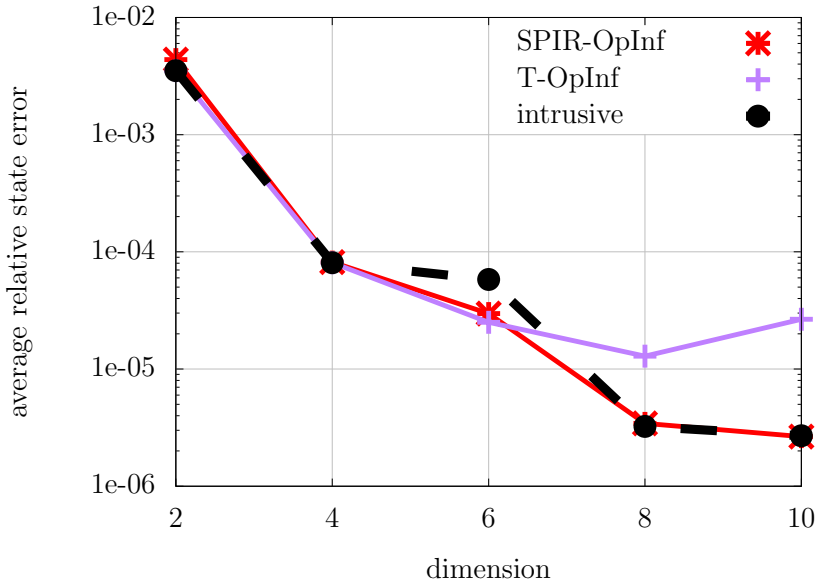


(a) estimated stability radius

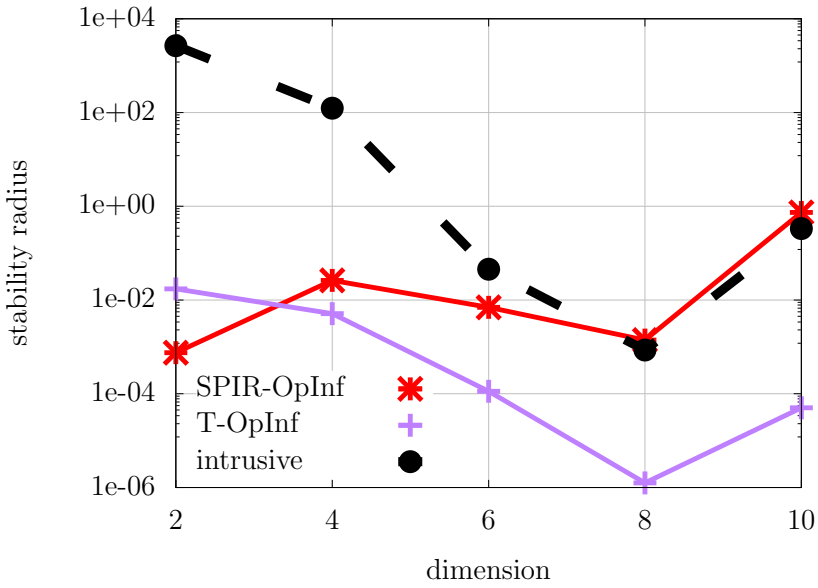


(b) validation error for dimension $n = 2$

Figure 4.3: Burgers' equation: Models learned with SPIR-OpInf have a larger estimated stability radius than models learned with Tikhonov regularization for dimensions $n > 4$ in this example. The stability radius of the SPIR-OpInf model is similar to the stability radius of the T-OpInf model for $n = 2$ because a small regularization parameter is chosen as shown in plot (b).



(a) test error (3.25)



(b) estimated stability radius (3.6)

Figure 4.4: Reaction-diffusion problem: Constraining the linear inferred operator to be symmetric and definite with SPIR-OpInf leads to models with comparable accuracy and stability radius as PIR-OpInf in this example.

Chapter 5

Non-intrusive entropy stable reduced order modeling

In this chapter, we discuss learning an entropy stable reduced model of nonlinear conservation laws, in a non-intrusive manner. Section 5.1 provides a literature overview. We closely base Section 5.2 on [Cha20] to describe details of the full model, which is a system of nonlinear conservation laws, the entropy conservation and entropy stability properties of the full model solutions, and the numerical schemes that preserve these properties in the discretized full model. Section 5.3 proposes constrained operator inference methods to learn entropy conserving and entropy stable reduced models in a non-intrusive manner from full model trajectories. The computational procedure is described in Section 5.4. Numerical results in Section 5.5 demonstrate that the proposed constrained operator inference learns entropy conserving and entropy stable models, while models learned without the constraints are unstable.

5.1 Introduction

Operator inference was developed for polynomial models and it preserves the polynomial structure of the full model, if present. However, full models can have a range of other types of structure and it is often beneficial to preserve these structures: There are methods that preserve Lagrangian [LKM03, CTB15] and Hamiltonian structure [BB12, GPBVDS12, PM16, CBG16, GWW17, AH17, MAH19] by formulating the projection step accordingly. There also is a line of work that aims to preserve energy by reducing nonlinear full models with sampling-based reduction techniques [FACC14, FCA15]. The construction of structure-preserving reduced models for nonlinear conservation laws however is less well researched. Some methods that improve stability of the reduced models are Petrov-Galerkin reduced models that use alternative test bases [MPR01, RV07, SLGB12, AF12, RHM13, BMQR15, CBA17] and other stabilization techniques [WABI12, KvAB14, CIJS14, BTD16, BKST09, KB10, KBAvBW14]. Another alternative is enforcing physical conditions such as kinetic energy preservation [ARWH20], which requires the full model in such a way that nonlinear terms only involve polynomial nonlinearities.

Our approach follows closely the work [Cha20] that approximates nonlinear terms using hyper-reduction techniques [AKJ08, HCF17]. This approach discretizes the full model using entropy stable finite volume schemes [Tad87, Tad03], and then generates entropy stable reduced models by combining hyper-reduction with modified Galerkin projection. Based on the ideas presented in [Cha20], we develop an entropy stable reduced modeling method that operates in a non-intrusive manner. We project the full model trajectories onto the reduced space, and fit the reduced model operators to the projected trajectories. We enforce constraints when solving the optimization problem, which ensure that we learn an entropy stable reduced model, at least on the training data. We reformulate the constraint and vectorize both the objective function and the constraint so that the optimization problem is

linear with respect to the reduced model operators.

5.2 Entropy and full models

In this section we discuss the structure of the full model and the numerical schemes that conserve that structure when discretizing the full model.

5.2.1 Systems of nonlinear conservation laws

We consider nonlinear conservation laws expressed as a system of PDEs

$$\frac{\partial \boldsymbol{q}(t,\xi_i)}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{f}_i(\boldsymbol{q}(t,\xi_i))}{\partial \xi_i} = 0, \tag{5.1}$$

where $\boldsymbol{q} = \boldsymbol{q}(t,\xi_i)$ are conservative variables and $\boldsymbol{f}_i(\boldsymbol{q}(t,\xi_i))$ are nonlinear fluxes. For the ease of notation, $\boldsymbol{q}(t,\xi_i)$ is written simply as $\boldsymbol{q}$ in the rest of the chapter.

The entropy function $S(\boldsymbol{q})$ and entropy variables $\boldsymbol{w}(\boldsymbol{q})$ for (5.1) are defined as

$$\text{convex function } S(\boldsymbol{q}), \quad \boldsymbol{w}(\boldsymbol{q}) = \frac{\partial S(\boldsymbol{q})}{\partial \boldsymbol{q}}.$$

A convex function $S(\boldsymbol{q})$ is called an entropy function if there exist d functions $\boldsymbol{F}_i,\ 1 \le i \le d$, called entropy fluxes, such that the condition

$$\frac{\partial S(\boldsymbol{q})}{\partial \boldsymbol{q}}^T \frac{\partial \boldsymbol{f}_i(\boldsymbol{q})}{\partial \boldsymbol{q}} = \frac{\partial \boldsymbol{F}_i(\boldsymbol{q})}{\partial \boldsymbol{q}}^T, \quad 1 \le i \le d, \tag{5.2}$$

holds. Multiplying (5.1) with $\boldsymbol{w}^T$ yields

$$\boldsymbol{w}^T \frac{\partial \boldsymbol{q}}{\partial t} + \boldsymbol{w}^T \sum_{i=1}^{d} \frac{\partial \boldsymbol{f}_i(\boldsymbol{q})}{\partial \xi_i} = 0. \tag{5.3}$$

Multiplying $\partial \boldsymbol{q}/\partial t$ by $\boldsymbol{w}^T$ gives

$$\boldsymbol{w}^T \frac{\partial \boldsymbol{q}}{\partial t} = \frac{\partial S(\boldsymbol{q})}{\partial \boldsymbol{q}}^T \frac{\partial \boldsymbol{q}}{\partial t} = \frac{\partial S(\boldsymbol{q})}{\partial t}^T.$$

Multiplying the flux gradients by $\boldsymbol{w}^T$ gives

$$\boldsymbol{w}^T \frac{\partial \boldsymbol{f}_i(\boldsymbol{q})}{\partial \xi_i} = \frac{\partial S(\boldsymbol{q})}{\partial \boldsymbol{q}}^T \frac{\partial \boldsymbol{f}_i(\boldsymbol{q})}{\partial \boldsymbol{q}} \frac{\partial \boldsymbol{q}}{\partial \xi_i} = \frac{\partial \boldsymbol{F}_i(\boldsymbol{q})}{\partial \boldsymbol{q}}^T \frac{\partial \boldsymbol{q}}{\partial \xi_i} = \frac{\partial \boldsymbol{F}_i(\boldsymbol{q})}{\partial \xi_i}^T.$$

Substituting these back into (5.3) gives an additional conservation law

$$\frac{\partial S(\boldsymbol{q})}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{F}_i(\boldsymbol{q})}{\partial \xi_i} = 0. \tag{5.4}$$

For weak solutions of (5.1), physically relevant solutions can be realized by vanishing viscosity limits, $\boldsymbol{q} = \lim_{\epsilon \to 0} \boldsymbol{q}_\epsilon$, where

$$\frac{\partial \boldsymbol{q}_\epsilon}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{f}_i(\boldsymbol{q}_\epsilon)}{\partial \xi_i} = \epsilon \Delta \boldsymbol{q}_\epsilon, \tag{5.5}$$

and ϵ is a small positive real number.

By multiplying (5.5) by $\boldsymbol{w}_\epsilon^T$, we get

$$\frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{F}_i(\boldsymbol{q}_\epsilon)}{\partial \xi_i} = \epsilon \boldsymbol{w}_\epsilon^T \Delta \boldsymbol{q}_\epsilon. \tag{5.6}$$

The right hand side term $\boldsymbol{w}_\epsilon^T \Delta \boldsymbol{q}_\epsilon$ can be written as

$$\boldsymbol{w}_\epsilon^T \Delta \boldsymbol{q}_\epsilon = \frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon}^T \frac{\partial^2 \boldsymbol{q}_\epsilon}{\partial \xi^2}.$$

Applying the product rule gives

$$\frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \xi^2} = \frac{\partial}{\partial \xi}\left(\frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial \xi}\right) = \frac{\partial}{\partial \xi}\left(\frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon}^T \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi}\right) = \frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon}^T \frac{\partial^2 \boldsymbol{q}_\epsilon}{\partial \xi^2} + \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi}^T \frac{\partial S^2(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon \partial \xi}.$$

Substituting this into the expression for $\boldsymbol{w}_\epsilon^T \Delta \boldsymbol{q}_\epsilon$ gives

$$\boldsymbol{w}_\epsilon^T \Delta \boldsymbol{q}_\epsilon = \frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon}^T \frac{\partial^2 \boldsymbol{q}_\epsilon}{\partial \xi^2} = \frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \xi^2} - \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi}^T \frac{\partial S^2(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon \partial \xi} = \frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \xi^2} - \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi}^T \frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon^2} \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi}.$$

As $S(\boldsymbol{q}_\epsilon)$ is a convex function, $\partial^2 S(\boldsymbol{q}_\epsilon)/\partial \boldsymbol{q}_\epsilon^2$ is positive definite, thus,

$$\boldsymbol{w}_\epsilon^T \Delta \boldsymbol{q}_\epsilon = \frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \xi^2} - \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi}^T \frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \boldsymbol{q}_\epsilon^2} \frac{\partial \boldsymbol{q}_\epsilon}{\partial \xi} \leq \frac{\partial^2 S(\boldsymbol{q}_\epsilon)}{\partial \xi^2}.$$

Substituting this in (5.6) gives

$$\frac{\partial S(\boldsymbol{q}_\epsilon)}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{F}_i(\boldsymbol{q}_\epsilon)}{\partial \xi_i} \leq \epsilon \Delta S(\boldsymbol{q}_\epsilon).$$

Thus, for $\boldsymbol{q} = \lim_{\epsilon \to 0} \boldsymbol{q}_\epsilon$, we get entropy stability equation

$$\frac{\partial S(\boldsymbol{q})}{\partial t} + \sum_{i=1}^{d} \frac{\partial \boldsymbol{F}_i(\boldsymbol{q})}{\partial \xi_i} \leq 0. \tag{5.7}$$

A weak solution of (5.1) is called entropy solution, if it satisfies (5.7).

5.2.2 Discretization of full model with entropy conserving schemes

In [Cha20], an entropy conservative finite volume scheme is used for discretizing the full model. The following closely follows derivation in [Cha20].

A two-point numerical flux $\boldsymbol{f}_S(\boldsymbol{q}_L, \boldsymbol{q}_R)$, for left and right states $\boldsymbol{q}_L, \boldsymbol{q}_R$, is entropy con-

servative if it satisfies the conditions

$$\begin{aligned} \boldsymbol{f}_S(\boldsymbol{q},\boldsymbol{q}) &= \boldsymbol{f}(\boldsymbol{q}), \\ \boldsymbol{f}_S(\boldsymbol{q}_L,\boldsymbol{q}_R) &= \boldsymbol{f}_S(\boldsymbol{q}_R,\boldsymbol{q}_L), \\ (\boldsymbol{w}_L-\boldsymbol{w}_R)^T\boldsymbol{f}_S(\boldsymbol{q}_L,\boldsymbol{q}_R) &= \psi(\boldsymbol{q}_L)-\psi(\boldsymbol{q}_R), \end{aligned} \tag{5.8}$$

where $\psi(\boldsymbol{q})$ is the entropy potential given by $\psi(\boldsymbol{q}) = \boldsymbol{w}(\boldsymbol{q})^T\boldsymbol{f}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q})$, where the function $\boldsymbol{F}$ is as defined in (5.2).

A matrix-based formulation of the scheme is

$$\Delta\xi\frac{d\boldsymbol{q}_h}{dt} + 2(\boldsymbol{Q}\odot\boldsymbol{F})\mathbf{1} = 0, \tag{5.9}$$

where $\Delta\xi$ is the cell size for the finite volume scheme, and $\boldsymbol{q}_h$ is the finite volume discretized solution, the matrix

$$\boldsymbol{Q} = \frac{1}{2}\begin{bmatrix} 0 & 1 & & \dots & -1 \\ -1 & 0 & 1 & & \\ & -1 & 0 & 1 & \\ & & & \ddots & \\ 1 & & \dots & -1 & 0 \end{bmatrix},$$

and

$$\boldsymbol{F}_{ij} = \boldsymbol{f}_S((\boldsymbol{q}_h)_i,(\boldsymbol{q}_h)_j).$$

The matrix formulation for a generalized system with n conservation laws is

$$\Delta\xi\frac{d\boldsymbol{q}_h}{dt} + 2((\boldsymbol{I}_{n\times n}\otimes\boldsymbol{Q})\odot\boldsymbol{F})\mathbf{1} = 0,$$

where $\boldsymbol{F}$ is a block-diagonal matrix, with i$^{\text{th}}$ block being the flux matrix corresponding to

i$^{\text{th}}$ flux, for $1 \leq i \leq n$.

Multiplying (5.9) with entropy variable vector $\boldsymbol{w}_h^T$ gives

$$\Delta\xi \boldsymbol{w}_h^T \frac{d\boldsymbol{q}_h}{dt} + 2\boldsymbol{w}_h^T(\boldsymbol{Q} \odot \boldsymbol{F})\mathbf{1} = 0,$$

which is equivalent to

$$\Delta\xi \mathbf{1}^T \frac{dS(\boldsymbol{q}_h)}{dt} + 2\boldsymbol{w}_h^T(\boldsymbol{Q} \odot \boldsymbol{F})\mathbf{1} = 0,$$

as

$$\boldsymbol{w}_h^T \frac{d\boldsymbol{q}_h}{dt} = \frac{\partial S(\boldsymbol{q}_h)}{\partial \boldsymbol{q}_h}^T \frac{d\boldsymbol{q}_h}{dt} = \mathbf{1}^T \frac{dS(\boldsymbol{q}_h)}{dt}.$$

Because of (5.4), an entropy conservative model satisfies

$$\Delta\xi \mathbf{1}^T \frac{dS(\boldsymbol{q}_h)}{dt} = 0 \iff 2\boldsymbol{w}_h^T(\boldsymbol{Q} \odot \boldsymbol{F})\mathbf{1} = 0.$$

For a skew-symmetric matrix $\boldsymbol{Q}$, the term $2\boldsymbol{w}_h^T(\boldsymbol{Q} \odot \boldsymbol{F})\mathbf{1}$ can be expanded as

$$\begin{aligned}
\sum_{ij} (\boldsymbol{w}_h)_i^T 2\boldsymbol{Q}_{ij} \boldsymbol{f}_S((\boldsymbol{q}_h)_i, (\boldsymbol{q}_h)_j) &= \sum_{ij} (\boldsymbol{Q}_{ij} - \boldsymbol{Q}_{ji})(\boldsymbol{w}_h)_i^T \boldsymbol{f}_S((\boldsymbol{q}_h)_i, (\boldsymbol{w}_h)_j) \\
&= \sum_{ij} \boldsymbol{Q}_{ij}((\boldsymbol{w}_h)_i - (\boldsymbol{w}_h)_j)^T \boldsymbol{f}_S((\boldsymbol{q}_h)_i, (\boldsymbol{q}_h)_j) \\
&= \sum_{ij} \boldsymbol{Q}_{ij}(\psi((\boldsymbol{q}_h)_i) - \psi((\boldsymbol{q}_h)_j)) \\
&= \boldsymbol{\psi}^T \boldsymbol{Q}\mathbf{1} - \mathbf{1}^T \boldsymbol{Q}\boldsymbol{\psi}.
\end{aligned}$$

As $\boldsymbol{Q}\mathbf{1} = \mathbf{0}$, we have

$$\boldsymbol{\psi}^T \boldsymbol{Q}\mathbf{1} - \mathbf{1}^T \boldsymbol{Q}\boldsymbol{\psi} = 0.$$

This proves that if $\boldsymbol{f}_S(\boldsymbol{q}_L, \boldsymbol{q}_R)$ is entropy conservative as defined in (5.8), $\boldsymbol{Q} = -\boldsymbol{Q}^T$ and $\boldsymbol{Q}\mathbf{1} = \mathbf{0}$, then $\boldsymbol{w}_h^T(\boldsymbol{Q} \odot \boldsymbol{F})\mathbf{1} = 0$ is satisfied, and model is entropy conservative.

5.2.3 Entropy stable numerical schemes

In this section, we again very closely follow [Cha20]. For entropy stability, numerical fluxes are constructed as before to conserve the entropy, and an artificial viscosity term is added to the model (5.9) to dissipate entropy. The full model equation with added artificial viscosity is

$$\Delta\xi \frac{d\boldsymbol{q}_h}{dt} + 2(\boldsymbol{Q} \odot \boldsymbol{F})\mathbf{1} + \epsilon \boldsymbol{K}\boldsymbol{q}_h = 0, \tag{5.10}$$

where

$$\boldsymbol{K} = \frac{1}{\Delta\xi}\begin{bmatrix} 1 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 1 \end{bmatrix}.$$

This is equivalent the discrete counterpart of (5.5).

For the entropy function gradient

$$\Delta\xi \mathbf{1}^T \frac{dS(\boldsymbol{q}_h)}{dt} = -\epsilon \boldsymbol{w}_h^T \boldsymbol{K}\boldsymbol{q}_h \leq 0,$$

holds true. Thus, entropy is dissipated, and (5.10) is entropy stable.

5.3 Non-intrusive entropy stable model reduction using operator inference

We learn reduced models in a non-intrusive manner from data and physical insights from the full model using operator inference. We enforce constraints when learning the reduced models, which ensures that the learned model is entropy stable at least for the training data.

5.3.1 Non-intrusive reduced model

For a discretized time domain $0 = t_0 < t_1 < \cdots < t_K = T$, we simulate the full model with M_b initial conditions $\boldsymbol{q}^b_{h,1,0}, \ldots, \boldsymbol{q}^b_{h,M_b,0}$, to generate basis trajectories $\boldsymbol{Q}^b_{h,1}, \ldots, \boldsymbol{Q}^b_{h,M_b}$, where

$$\boldsymbol{Q}^b_{h,i} = [\boldsymbol{q}^b_{h,i,1}, \ldots, \boldsymbol{q}^b_{h,i,K}] \in \mathbb{R}^{N \times K}, \text{ for } i = 1, \ldots, M_b.$$

The corresponding entropy variable trajectories are

$$\boldsymbol{W}^b_{h,i} = [\boldsymbol{w}^b_{h,i,1}, \ldots, \boldsymbol{w}^b_{h,i,K}] \in \mathbb{R}^{N \times K}, \text{ for } i = 1, \ldots, M_b.$$

As the projected entropy variables are used as a part of the constraint when learning the reduced model, it is necessary that they are representable in the reduced basis. Hence, we combine the conservative variable trajectories and the entropy variable trajectories to generate the snapshot matrix

$$\boldsymbol{\mathcal{Q}}^b_h = [\boldsymbol{Q}^b_{h,1}, \boldsymbol{W}^b_{h,1}, \ldots, \boldsymbol{Q}^b_{h,M_b}, \boldsymbol{W}^b_{h,M_b}] \in \mathbb{R}^{N \times 2KM_b}.$$

A POD basis $\boldsymbol{V}$ of dimension $n \ll N$ is constructed from the singular value decomposition of the snapshot matrix $\boldsymbol{\mathcal{Q}}^b_h$.

We aim to learn a reduced model of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\bar{\boldsymbol{q}} - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r = 0, \tag{5.11}$$

where $\bar{\boldsymbol{q}} = \boldsymbol{V}^T \boldsymbol{q}_h$ is the full model data projected onto the reduced space, and $\hat{\boldsymbol{F}}_r \in \mathbb{R}^{n \times n_r}$ are reduced model nonlinear operators, for a reduced model of dimension n. Our goal is now to learn reduced model operators $\hat{\boldsymbol{F}}_r$ that fit the projected data.

For the time steps $0 = t_0 < t_1 < \cdots < t_K = T$, we iterate the full model with M_t initial

conditions $\boldsymbol{q}_{h,1,0}, \ldots, \boldsymbol{q}_{h,M_t,0}$, to generate training trajectories $\boldsymbol{Q}_{h,1}, \ldots, \boldsymbol{Q}_{h,M_t}$, where

$$\boldsymbol{Q}_{h,i} = [\boldsymbol{q}_{h,i,1}, \ldots, \boldsymbol{q}_{h,i,K}] \in \mathbb{R}^{N \times K}, \text{ for } i = 1, \ldots, M_t.$$

We project them onto the reduced space to generate projected trajectories $\bar{\boldsymbol{Q}}_1, \ldots, \bar{\boldsymbol{Q}}_{M_t}$, where $\bar{\boldsymbol{Q}}_i = \boldsymbol{V}^T \boldsymbol{Q}_{h,i}$ for $i = 1, \ldots, M_t$. The operators $\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R$ are fitted via least-squares regression to the projected training trajectories

$$\min_{\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R} J(\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R),$$

with the objective function

$$J(\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R) = \sum_{i=1}^{M_t} \sum_{k=1}^{K} \left\| \bar{\boldsymbol{q}}'_{i,k} - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r_{i,k} \right\|_2^2, \tag{5.12}$$

where $\bar{\boldsymbol{q}}'_{i,k}$ denotes a numerical approximation of the time derivative of the projected state at time k of the i$^{\text{th}}$ trajectory, such as a first-order finite difference approximation.

5.3.2 Non-intrusive reduced model with entropy conservation

There is no guarantee that the reduced model learned by solving the optimization problem (5.12) is entropy stable. We derive and impose certain constraints on $\hat{\boldsymbol{F}}_r$ to ensure that the learned reduced model is entropy conservative, at least on the training data, which is a stricter constraint than entropy stability. In next section, we derive and impose a constraint for entropy stability.

Multiplying (5.11) with $\bar{\boldsymbol{w}}^T$, where $\bar{\boldsymbol{w}} = \boldsymbol{V}^T \boldsymbol{w}_h$, we would like that the following holds

$$\bar{\boldsymbol{w}}^T \frac{d\bar{\boldsymbol{q}}}{dt} - \bar{\boldsymbol{w}}^T \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r \stackrel{!}{=} 0,$$

which is equivalent to

$$\mathbf{1}^T \frac{dS(\bar{\boldsymbol{q}})}{dt} - \bar{\boldsymbol{w}}^T \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r \stackrel{!}{=} 0.$$

For an entropy conserving reduced model

$$\mathbf{1}^T \frac{dS(\bar{\boldsymbol{q}})}{dt} = 0 \iff \bar{\boldsymbol{w}}^T \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r = 0.$$

Thus, for the reduced model we conserve the entropy by enforcing the entropy preservation constraint

$$\bar{\boldsymbol{w}}^T \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r = 0,$$

on the training data.

For entropy conservation, we solve a constrained optimization problem for operator inference with entropy conservation (EC-OpInf)

$$\begin{aligned} \min_{\hat{\boldsymbol{F}}_2,\ldots,\hat{\boldsymbol{F}}_R} \quad & J(\hat{\boldsymbol{F}}_2,\ldots,\hat{\boldsymbol{F}}_R),, \\ \text{such that} \quad & \bar{\boldsymbol{w}}_{i,k}^T \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}_{i,k}^r = 0 \text{ for } i = 1,\ldots,M_t,\ k = 1,\ldots,K, \end{aligned} \tag{5.13}$$

with the same objective function $J(\hat{\boldsymbol{F}}_2,\ldots,\hat{\boldsymbol{F}}_R)$ as (5.12).

5.3.3 Non-intrusive reduced model with entropy stability

For entropy stability, we equip the reduced model with an additional term that serves as artificial viscosity term,

$$\frac{d\bar{\boldsymbol{q}}}{dt} - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r - \hat{\boldsymbol{A}}\bar{\boldsymbol{q}} \stackrel{!}{=} 0. \tag{5.14}$$

Multiplying (5.14) by $\bar{\boldsymbol{w}}^T$ gives

$$\mathbf{1}^T \frac{dS(\bar{\boldsymbol{q}})}{dt} - \bar{\boldsymbol{w}}^T \left(\sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r + \hat{\boldsymbol{A}} \bar{\boldsymbol{q}} \right) \overset{!}{=} 0 \implies \mathbf{1}^T \frac{dS(\bar{\boldsymbol{q}})}{dt} \overset{!}{=} \bar{\boldsymbol{w}}^T \left(\sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r + \hat{\boldsymbol{A}} \bar{\boldsymbol{q}} \right).$$

For entropy stability,

$$\mathbf{1}^T \frac{dS(\bar{\boldsymbol{q}})}{dt} \leq 0 \iff \bar{\boldsymbol{w}}^T \left(\sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r + \hat{\boldsymbol{A}} \bar{\boldsymbol{q}} \right) \leq 0.$$

Thus, the entropy stability constraint is

$$\bar{\boldsymbol{w}}^T \left(\sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}^r + \hat{\boldsymbol{A}} \bar{\boldsymbol{q}} \right) \leq 0.$$

For entropy stability, we solve the constrained optimization problem for operator inference with entropy stability (ES-OpInf)

$$\begin{aligned} \min_{\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R, \hat{\boldsymbol{A}}} \quad & J(\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R, \hat{\boldsymbol{A}}), \\ \text{such that} \quad & \bar{\boldsymbol{w}}_{i,k}^T \left(\sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}_{i,k}^r + \hat{\boldsymbol{A}} \bar{\boldsymbol{q}}_{i,k} \right) \leq 0 \text{ for } i = 1, \ldots, M_t,\ k = 1, \ldots, K, \end{aligned} \tag{5.15}$$

with the objective function

$$J(\hat{\boldsymbol{F}}_2, \ldots, \hat{\boldsymbol{F}}_R, \hat{\boldsymbol{A}}) = \sum_{i=1}^{M_t} \sum_{k=1}^{K} \left\| \bar{\boldsymbol{q}}'_{i,k} - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}_{i,k}^r - \hat{\boldsymbol{A}} \bar{\boldsymbol{q}}_{i,k} \right\|_2^2. \tag{5.16}$$

5.4 Computational procedure

We reformulate the optimization problems (5.13) and (5.15) as optimization problems whose objective function and constraints are linear with respect to the the model operators.

5.4.1 Snapshot preparation

Consider the projected data trajectories $\bar{\boldsymbol{Q}}_1, \ldots, \bar{\boldsymbol{Q}}_{M_t}$, where

$$\bar{\boldsymbol{Q}}_i = [\bar{\boldsymbol{q}}_{i,1} \ , \ldots \ , \bar{\boldsymbol{q}}_{i,K}] \in \mathbb{R}^{n \times K}, \text{ for } i = 1, \ldots, M_t,$$

and the corresponding projected nonlinear data trajectories

$$\bar{\boldsymbol{Q}}_i^r = [\bar{\boldsymbol{q}}_{i,1}^r \ , \ldots \ , \bar{\boldsymbol{q}}_{i,K}^r] \in \mathbb{R}^{n_r \times K}, \text{ for } i = 1, \ldots, M_t,$$

for $r = 2, \ldots, R$, with $n_2 = n(n+1)/2, \ n_3 = n(n+1)(n+2)/6$, etc.

The entropy variable trajectories are $\bar{\boldsymbol{W}}_1, \ldots, \bar{\boldsymbol{W}}_{M_t}$, where

$$\bar{\boldsymbol{W}}_i = [\bar{\boldsymbol{w}}_{i,1} \ , \ldots \ , \bar{\boldsymbol{w}}_{i,K}] \in \mathbb{R}^{n \times K}, \text{ for } i = 1, \ldots, M_t.$$

The snapshot matrices constructed from these trajectories are

$$\begin{aligned}
\bar{\boldsymbol{Q}} &= [\bar{\boldsymbol{Q}}_1 \ , \ldots \ , \bar{\boldsymbol{Q}}_{M_t}] \in \mathbb{R}^{n \times KM_t}, \\
\bar{\boldsymbol{Q}}^r &= [\bar{\boldsymbol{Q}}_1^r \ , \ldots \ , \bar{\boldsymbol{Q}}_{M_t}^r] \in \mathbb{R}^{n_r \times KM_t}, \quad r = 2, \ldots, R, \\
\bar{\boldsymbol{W}} &= [\bar{\boldsymbol{W}}_1 \ , \ldots \ , \bar{\boldsymbol{W}}_{M_t}] \in \mathbb{R}^{n \times KM_t}.
\end{aligned}$$

5.4.2 Entropy conservation

Consider the vector

$$\hat{\boldsymbol{F}}^v = [\hat{\boldsymbol{F}}_2^v, \ldots, \hat{\boldsymbol{F}}_R^v]^T \in \mathcal{R}^{nn_R},$$

where $n_R = \sum_{r=2}^{R} n_r$, and $\hat{\boldsymbol{F}}_r^v = \text{vec}(\hat{\boldsymbol{F}}_r)$ is the vectorized form of matrix $\hat{\boldsymbol{F}}_r$, for $r = 2, \ldots, R$.

Proposition 5.1. *For the vector $\hat{\boldsymbol{F}}^v$, there exists a matrix $\boldsymbol{C}$ that is independent of $\hat{\boldsymbol{F}}^v$ such*

that $\boldsymbol{C}\hat{\boldsymbol{F}}^{v} = \boldsymbol{0}$*, that represents the same set of equations as the constraint in* (5.13)*.*

Proof. Consider a matrix $\boldsymbol{C}_f = \sum_{r=2}^{R} \text{diag}(\boldsymbol{D}_r) \in \mathbb{R}^{KM_t}$, where $\boldsymbol{D}_r = \bar{\boldsymbol{W}}^T \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r$, with $\bar{\boldsymbol{Q}}^r$ and $\bar{\boldsymbol{W}}$ as defined in Section 5.4.1. Comparing the entries of $\boldsymbol{C}_f$ to the equality constraint in (5.13) gives us

$$\boldsymbol{C}_f = \boldsymbol{0} \iff \bar{\boldsymbol{w}}_{i,k}^T \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}_{i,k}^r = 0 \text{ for } i = 1, \ldots, M_t,\ k = 1, \ldots, K. \tag{5.17}$$

The vector $\text{diag}(\boldsymbol{D}_r)$ can be written as

$$\text{diag}(\boldsymbol{D}_r) = \sum_{i=1}^{KM_t} \boldsymbol{e}^T \boldsymbol{P}_i \boldsymbol{D}_r \boldsymbol{P}_i = \sum_{i=1}^{KM_t} \boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r \boldsymbol{P}_i,$$

where $\boldsymbol{e} = \boldsymbol{1} \in \mathbb{R}^{KM_t}$, and $\boldsymbol{P}_i \in \mathbb{R}^{KM_t \times KM_t},\ i = 1, \ldots, KM_t$, such that $\boldsymbol{P}_i(i,i) = 1$, and the rest of the entries of $\boldsymbol{P}_i$ are all zeros.

The vector $\boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r \boldsymbol{P}_i$ can be written as

$$\boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r \boldsymbol{P}_i = (\boldsymbol{L}_{r,i} \otimes \boldsymbol{R}_{r,i}) \hat{\boldsymbol{F}}_r^{v},$$

where $\boldsymbol{L}_{r,i} = \boldsymbol{P}_i^T (\bar{\boldsymbol{Q}}^r)^T \in \mathbb{R}^{KM_t \times n_r},\ \boldsymbol{R}_{r,i} = \boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \in \mathbb{R}^{1 \times n}$, and $\hat{\boldsymbol{F}}_r^{v} = \text{vec}(\hat{\boldsymbol{F}}_r) \in \mathbb{R}^{nn_r}$ is the vectorized form of matrix $\hat{\boldsymbol{F}}_r$.

Substituting this back into the expression for $\text{diag}(\boldsymbol{D}_r)$ gives us

$$\text{diag}(\boldsymbol{D}_r) = \sum_{i=1}^{KM_t} (\boldsymbol{L}_{r,i} \otimes \boldsymbol{R}_{r,i}) \hat{\boldsymbol{F}}_r^{v}.$$

Substituting this back into the expression for $\boldsymbol{C}_f$ gives us

$$\boldsymbol{C}_f = \sum_{r=2}^{R} \left(\sum_{i=1}^{KM_t} (\boldsymbol{L}_{r,i} \otimes \boldsymbol{R}_{r,i}) \right) \hat{\boldsymbol{F}}_r^{v} = \boldsymbol{C}\hat{\boldsymbol{F}}^{v},$$

where

$$\boldsymbol{C} = \begin{bmatrix} \sum_{i=1}^{KM_t} (\boldsymbol{L}_{2,i} \otimes \boldsymbol{R}_{2,i}) & \dots & \sum_{i=1}^{KM_t} (\boldsymbol{L}_{R,i} \otimes \boldsymbol{R}_{R,i}) \end{bmatrix} \in \mathcal{R}^{KM_t \times nn_R}.$$

Thus, from (5.17) we get that for the matrix $\boldsymbol{C}$ and vector $\hat{\boldsymbol{F}}^v$, the equation

$$\boldsymbol{C}\hat{\boldsymbol{F}}^v = \boldsymbol{0},$$

represents the same set of equations as the constraint in (5.13). □

The entropy conservation constraint is linear w.r.t $\hat{\boldsymbol{F}}^v$. As n_R scales with n^R, the constraint matrix $\boldsymbol{C}$ scales with n^{R+1}, where n is the reduced model dimension, and R is the degree of the highest degree nonlinear term. The constraint matrix $\boldsymbol{C}$ also scales linearly with number of time steps K and number of training trajectories M_t.

As the constraint is a function of $\hat{\boldsymbol{F}}^v$, we vectorize the objective function as well, to make sure it is also a function of $\hat{\boldsymbol{F}}^v$. The objective function (5.12) can be rewritten as

$$\left\| \bar{\boldsymbol{Q}}' - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r \right\|_F^2.$$

Vectorizing $\bar{\boldsymbol{Q}}' - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r$ yields

$$\begin{aligned} \operatorname{vec}\left(\bar{\boldsymbol{Q}}' - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r \right) &= \operatorname{vec}(\bar{\boldsymbol{Q}}') - \sum_{r=2}^{R} ((\bar{\boldsymbol{Q}}^r)^T \otimes \boldsymbol{I}_n) \hat{\boldsymbol{F}}_r^v, \\ &= \operatorname{vec}(\bar{\boldsymbol{Q}}') - \boldsymbol{C}_o \hat{\boldsymbol{F}}^v, \end{aligned}$$

where

$$\boldsymbol{C}_o = \begin{bmatrix} (\bar{\boldsymbol{Q}}^2)^T \otimes \boldsymbol{I}_n & \dots & (\bar{\boldsymbol{Q}}^R)^T \otimes \boldsymbol{I}_n \end{bmatrix} \in \mathcal{R}^{nKM_t \times nn_R}.$$

The data matrix $\boldsymbol{C}_o$ scales with n^{R+2}, and it also scales linearly w.r.t. number of time steps K and number of training trajectories M_t.

The updated optimization problem for EC-OpInf (5.13) is

$$\begin{aligned} \min_{\hat{\boldsymbol{F}}^v} \quad & J(\hat{\boldsymbol{F}}^v), \\ \text{such that} \quad & \boldsymbol{C}\hat{\boldsymbol{F}}^v = \boldsymbol{0}\,, \end{aligned} \tag{5.18}$$

with the objective function

$$J(\hat{\boldsymbol{F}}^v) = \left\| \mathrm{vec}(\bar{\boldsymbol{Q}}') - \boldsymbol{C}_o \hat{\boldsymbol{F}}^v \right\|_2^2 .$$

We solve the optimization problem (5.18) with a linear conservation constraint, using the linear least-squares solver 'lsqlin' [CL96, GMW19] in MATLAB.

5.4.3 Entropy stability

Consider the vector

$$\hat{\boldsymbol{F}}_A^v = [\hat{\boldsymbol{A}}^v, \hat{\boldsymbol{F}}_2^v, \ldots, \hat{\boldsymbol{F}}_R^v]^T \in \mathcal{R}^{n^2+nn_R},$$

where $\hat{\boldsymbol{A}}^v = \mathrm{vec}(\hat{\boldsymbol{A}})$ is the vectorized form of matrix $\hat{\boldsymbol{A}}$.

Proposition 5.2. *For the vector $\hat{\boldsymbol{F}}_A^v$, there exists a matrix $\boldsymbol{C}$ that is independent of $\hat{\boldsymbol{F}}_A^v$ such that $\boldsymbol{C}\hat{\boldsymbol{F}}_A^v \leq \boldsymbol{0}$ represents the same set of equations as the constraint in* (5.15).

Proof. Consider a matrix $\boldsymbol{C}_f = \mathrm{diag}(\boldsymbol{D}_A) + \sum_{r=2}^{R} \mathrm{diag}(\boldsymbol{D}_r) \in \mathbb{R}^{KM_t}$, where $\boldsymbol{D}_A = \bar{\boldsymbol{W}}^T \hat{\boldsymbol{A}} \bar{\boldsymbol{Q}}$ and $\boldsymbol{D}_r = \bar{\boldsymbol{W}}^T \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r$, with $\bar{\boldsymbol{Q}}$, $\bar{\boldsymbol{Q}}^r$ and $\bar{\boldsymbol{W}}$ as defined in Section 5.4.1. Comparing the entries of $\boldsymbol{C}_f$ to the constraint in (5.15) gives us

$$\boldsymbol{C}_f \leq \boldsymbol{0} \iff \bar{\boldsymbol{w}}_{i,k}^T \left(\sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{q}}_{i,k}^r + \hat{\boldsymbol{A}} \bar{\boldsymbol{q}}_{i,k} \right) \leq 0 \text{ for } i = 1, \ldots, M_t,\ k = 1, \ldots, K\,. \tag{5.19}$$

The vector $\text{diag}(\boldsymbol{D}_A)$ can be written as

$$\text{diag}(\boldsymbol{D}_A) = \sum_{i=1}^{KM_t} \boldsymbol{e}^T \boldsymbol{P}_i \boldsymbol{D}_A \boldsymbol{P}_i = \sum_{i=1}^{KM_t} \boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \hat{\boldsymbol{A}} \bar{\boldsymbol{Q}} \boldsymbol{P}_i,$$

where $\boldsymbol{e} = \boldsymbol{1} \in \mathbb{R}^{KM_t}$, and $\boldsymbol{P}_i \in \mathbb{R}^{KM_t \times KM_t},\ i = 1, \ldots, KM_t$, such that $\boldsymbol{P}_i(i,i) = 1$, and the rest of the entries of $\boldsymbol{P}_i$ are all zeros. The vector $\text{diag}(\boldsymbol{D}_r)$ is as defined in Proposition 5.1.

The vector $\boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \hat{\boldsymbol{A}} \bar{\boldsymbol{Q}} \boldsymbol{P}_i$ can be written as

$$\boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \hat{\boldsymbol{A}} \bar{\boldsymbol{Q}} \boldsymbol{P}_i = (\boldsymbol{L}_{A,i} \otimes \boldsymbol{R}_{A,i}) \hat{\boldsymbol{A}}^v,$$

where $\boldsymbol{L}_{A,i} = \boldsymbol{P}_i^T \bar{\boldsymbol{Q}}^T \in \mathbb{R}^{KM_t \times n},\ \boldsymbol{R}_{A,i} = \boldsymbol{e}^T \boldsymbol{P}_i \bar{\boldsymbol{W}}^T \in \mathbb{R}^{1 \times n}$, and $\hat{\boldsymbol{A}}^v = \text{vec}(\hat{\boldsymbol{A}}) \in \mathbb{R}^{n^2}$ is the vectorized form of matrix $\hat{\boldsymbol{A}}$.

Substituting this back into the expression for $\text{diag}(\boldsymbol{D}_A)$ gives us

$$\text{diag}(\boldsymbol{D}_A) = \sum_{i=1}^{KM_t} (\boldsymbol{L}_{A,i} \otimes \boldsymbol{R}_{A,i}) \hat{\boldsymbol{A}}^v.$$

Substituting this, and the expression for $\text{diag}(\boldsymbol{D}_r)$ computed in Proposition 5.1, back into the expression for $\boldsymbol{C}_f$ gives us

$$\boldsymbol{C}_f = \left(\sum_{i=1}^{KM_t} (\boldsymbol{L}_{A,i} \otimes \boldsymbol{R}_{A,i}) \right) \hat{\boldsymbol{A}}^v + \sum_{r=2}^{R} \left(\sum_{i=1}^{KM_t} (\boldsymbol{L}_{r,i} \otimes \boldsymbol{R}_{r,i}) \right) \hat{\boldsymbol{F}}_r^v = \boldsymbol{C} \hat{\boldsymbol{F}}_A^v,$$

where matrix $\boldsymbol{C} \in \mathcal{R}^{KM_t \times (n^2 + nn_R)}$ is given by

$$\boldsymbol{C} = \begin{bmatrix} \sum_{i=1}^{KM_t} (\boldsymbol{L}_{A,i} \otimes \boldsymbol{R}_{A,i}) & \sum_{i=1}^{KM_t} (\boldsymbol{L}_{2,i} \otimes \boldsymbol{R}_{2,i}) & \ldots & \sum_{i=1}^{KM_t} (\boldsymbol{L}_{R,i} \otimes \boldsymbol{R}_{R,i}) \end{bmatrix}.$$

Thus, from (5.19) we get that for the matrix $\boldsymbol{C}$ and vector $\hat{\boldsymbol{F}}_A^v$, the equation

$$\boldsymbol{C}\hat{\boldsymbol{F}}_A^v \leq \boldsymbol{0},$$

represents the same set of equations as the constraint in (5.15). □

The entropy stability constraint is linear w.r.t. to $\hat{\boldsymbol{F}}_A^v$. The constraint matrix $\boldsymbol{C}$ scales with n^{R+1}, and scales linearly with number of time steps K and number of training trajectories M_t.

The objective function (5.16) can be rewritten as

$$\left\| \bar{\boldsymbol{Q}}' - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r - \hat{\boldsymbol{A}}\bar{\boldsymbol{Q}} \right\|_F^2.$$

Vectorizing $\bar{\boldsymbol{Q}}' - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r - \hat{\boldsymbol{A}}\bar{\boldsymbol{Q}}$ yields

$$\begin{aligned}\operatorname{vec}\left(\bar{\boldsymbol{Q}}' - \sum_{r=2}^{R} \hat{\boldsymbol{F}}_r \bar{\boldsymbol{Q}}^r - \hat{\boldsymbol{A}}\bar{\boldsymbol{Q}}\right) &= \operatorname{vec}(\bar{\boldsymbol{Q}}') - ((\bar{\boldsymbol{Q}}^r)^T \otimes \boldsymbol{I}_n)\hat{\boldsymbol{F}}_r^v - (\bar{\boldsymbol{Q}}^T \otimes \boldsymbol{I}_n)\hat{\boldsymbol{A}}^v, \\ &= \operatorname{vec}(\bar{\boldsymbol{Q}}') - \boldsymbol{C}_o\hat{\boldsymbol{F}}_A^v,\end{aligned}$$

where

$$\boldsymbol{C}_o = \begin{bmatrix} \bar{\boldsymbol{Q}}^T \otimes \boldsymbol{I}_n & (\bar{\boldsymbol{Q}}^2)^T \otimes \boldsymbol{I}_n & \dots & (\bar{\boldsymbol{Q}}^R)^T \otimes \boldsymbol{I}_n \end{bmatrix} \in \mathcal{R}^{nKM_t \times (n^2 + nn_R)}.$$

The data matrix $\boldsymbol{C}_o$ scales with n^{R+2}, and it also scales linearly w.r.t. number of time steps K and number of training trajectories M_t.

The updated optimization problem for ES-OpInf (5.15) is

$$\begin{aligned} \min_{\hat{\boldsymbol{F}}_A^v} \quad & J(\hat{\boldsymbol{F}}_A^v), \\ \text{such that} \quad & \boldsymbol{C}\hat{\boldsymbol{F}}_A^v \leq \boldsymbol{0} \end{aligned} \tag{5.20}$$

with the objective function

$$J(\hat{\boldsymbol{F}}_A^v) = \left\| \text{vec}(\bar{\boldsymbol{Q}}') - \boldsymbol{C}_o \hat{\boldsymbol{F}}_A^v \right\|_2^2,$$

We solve the optimization problem (5.20) with a linear stability constraint, using the linear least-squares solver 'lsqlin' [CL96, GMW19] in MATLAB.

5.5 Numerical experiments

We consider two test cases to demonstrate entropy conserving and entropy stable operator inference. First, we demonstrate operator inference on the Burgers' equation, which has a quadratic flux, and where we impose the entropy stability constraint. Second, we demonstrate operator inference on the shallow water equation that has a quadratic-cubic flux, and where we impose the entropy conservation constraint.

5.5.1 Burgers' equation

The conservative form of the Burgers' equation is

$$\frac{\partial q(t,\xi)}{\partial t} + \frac{\partial (q^2(t,\xi)/2)}{\partial \xi} = 0, \quad \xi \in [-1,1],\ t \in [0,4],$$

with

$$\begin{aligned}
&\text{Entropy function } S(q) = \frac{q^2}{2},\\
&\text{Entropy variable } v(q) = q,\\
&\text{Entropy flux } F(q) = \frac{q^3}{3},\\
&\text{Entropy potential } \psi(q) = \frac{q^3}{6}.
\end{aligned}$$

The entropy conserving numerical flux is

$$\boldsymbol{f}_S(q_L, q_R) = \frac{q_L^2 + q_L q_R + q_R^2}{6}.$$

As shock discontinuities are observed, we add an artificial viscosity term with $\epsilon = 10^{-3}$ to the model. We discretize the spatial domain $[-1, 1]$ using $\Delta\xi = 10^{-2}$ and finite volume scheme (5.10) and the time domain $[0, 4]$ using explicit Euler with time-step size $\delta t = 5 \times 10^{-3}$. The dimension of the high-dimensional model is $N = 200$. To construct the reduced space, we generate $M_b = 5$ basis trajectories with initial conditions $\boldsymbol{q}_{i,0} = (-0.5\ i/5)\sin(\pi x)$, for $i = 1, \ldots, 5$. Figure 5.1a plots the normalized singular values for the model, to show the decay.

To train the model, we generate $M_t = 10$ training trajectories with initial conditions $\boldsymbol{q}_{i,0} = c_i \sin(\pi x)$, for $i = 1, \ldots, 10$, where c_i are randomly generated via a uniform distribution in $[-0.1, -0.5]$. We test the model on a test trajectory with initial condition $\boldsymbol{q}_0^{\text{test}} = -0.3\sin(\pi x)$. Figure 5.1b plots the test error. The reduced model learned via unconstrained operator inference is unstable for all dimensions except 8 and 12, while the reduced model learned via ES-OpInf is stable for all dimensions.

The reduced model is iterated with the projected test initial condition $\bar{\boldsymbol{q}}_0^{\text{test}} = \boldsymbol{V}^T \boldsymbol{q}_0^{\text{test}}$, to generate reduced test state trajectory $\hat{\boldsymbol{Q}}^{\text{test}}$ and corresponding reduced test entropy variable trajectory $\hat{\boldsymbol{W}}^{\text{test}}$. The trajectories $\hat{\boldsymbol{Q}}^{\text{test}}$ and $\hat{\boldsymbol{W}}^{\text{test}}$ are then used to generate entropy trajectory $S(\hat{\boldsymbol{Q}}^{\text{test}})$. Figures 5.2 and 5.3 plot entropy function summed over all nodes $\mathbf{1}^T S(\hat{\boldsymbol{Q}}^{\text{test}})$, and entropy gradient $\mathbf{1}^T dS(\hat{\boldsymbol{Q}}^{\text{test}})/dt$, at all time steps, for the reduced models of dimension $10, 14, 16, 18$ and 20. For the reduced model learned via ES-OpInf, the entropy value goes down with time, and the entropy gradient satisfies $\mathbf{1}^T dS(\hat{\boldsymbol{Q}}^{\text{test}})/dt \leq 0$ at all times, which indicates an entropy stable model. For the reduced model learned via unconstrained operator inference, the entropy blows up, thus the model is unstable.

Figure 5.4 plots the test state trajectory for full and reduced model at $t = 0$ and $t = 4$.

For the reduced model with dimension 10, the reduced state trajectory projected back onto the full model space $\boldsymbol{V}\hat{\boldsymbol{Q}}^{\text{test}}$ is plotted. As seen in Figure 5.4, the state trajectory for the reduced model is similar to the full model, thus showing that the reduced model accurately captures the full model behavior.

5.5.2 Shallow water equation

A conservative form of shallow water equation is

$$\begin{aligned}
&\frac{\partial h(t,\xi)}{\partial t} + \frac{\partial (h(t,\xi)\ \phi(t,\xi))}{\partial \xi} = 0, \\
&\frac{\partial (h(t,\xi)\ \phi(t,\xi))}{\partial t} + \frac{\partial (h(t,\xi)\ \phi^2(t,\xi) + h^2(t,\xi)/2)}{\partial \xi} = 0,
\end{aligned}$$

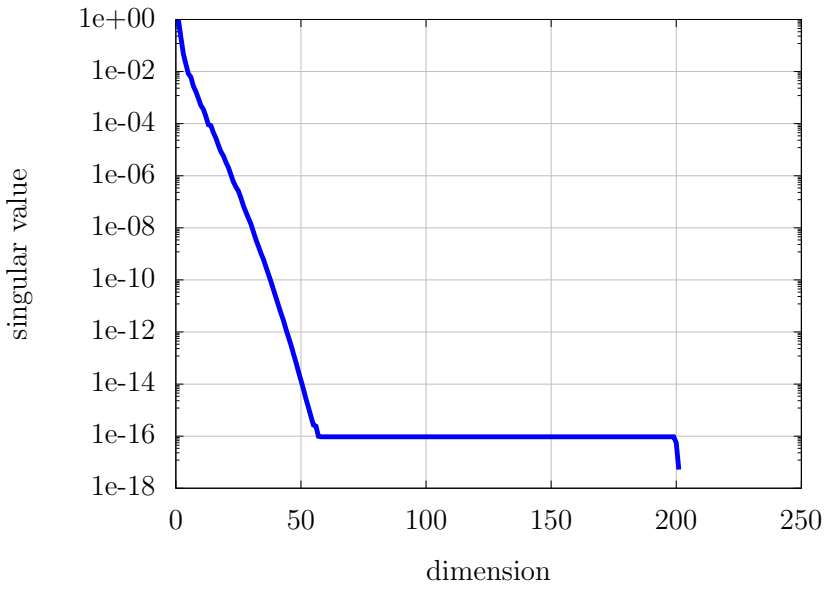
with $\xi \in [-10, 10], t \in [0, 5]$, and

$$\begin{aligned}
&\text{Entropy function } S(h,\phi) = \frac{1}{2}h\phi^2 + \frac{1}{2}h^2, \\
&\text{Entropy variable } v(h,\phi) = \begin{bmatrix} h - \phi^2/2 \\ \phi \end{bmatrix} \\
&\text{Entropy flux } F(h,\phi) = \frac{1}{2}h\phi^3 + h^2\phi, \\
&\text{Entropy potential } \psi(h,\phi) = \frac{1}{2}h^2\phi.
\end{aligned}$$

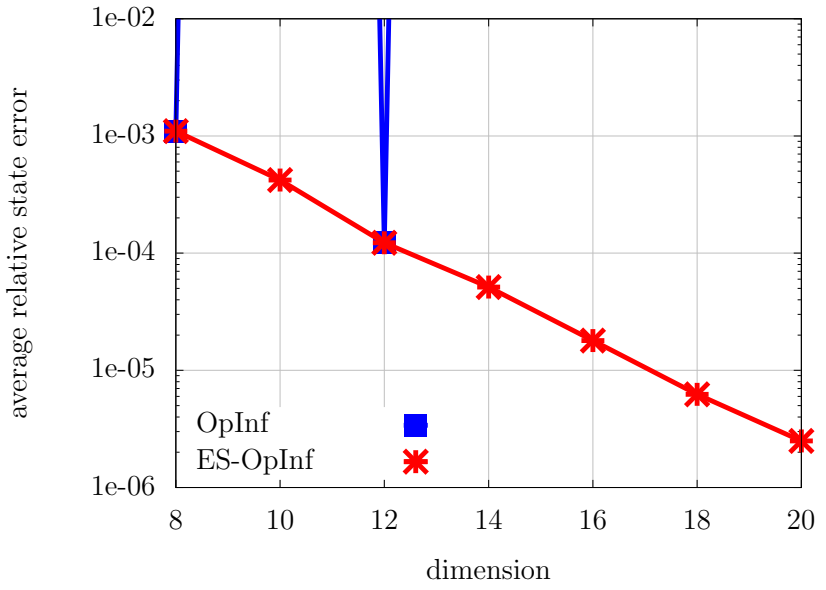
The entropy conserving numerical flux is

$$\boldsymbol{f}_S\Big([h_L\ \phi_L], [h_R\ \phi_R]\Big) = \begin{bmatrix} (h_L\phi_L + h_R\phi_R)/2 \\ (h_L\phi_L + h_R\phi_R)(\phi_L + \phi_R)/4 + h_L h_R/2 \end{bmatrix}.$$

Shallow-water equations describe the kinetic behavior of a thin inviscid fluid layer flowing over a variable topography. The variable h is the height of the free-surface, and the variable

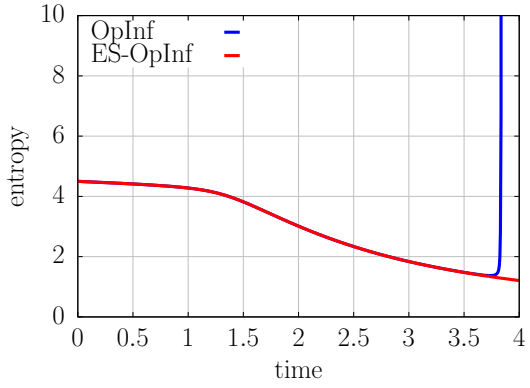


(a) decay of singular values

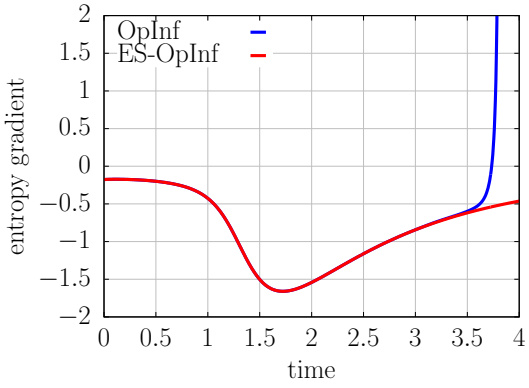


(b) test error

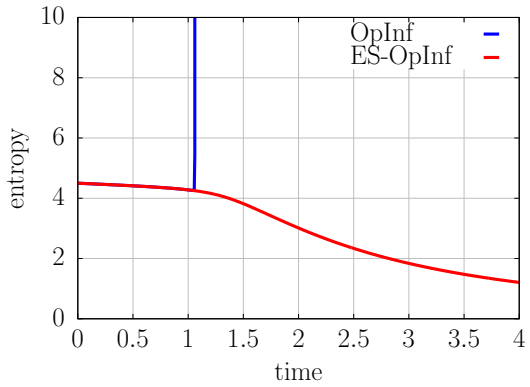
Figure 5.1: Burgers' equation: The reduced model learned via unconstrained OpInf is unstable for all dimensions except 8 and 12, while the reduced model learned via ES-OpInf is stable for all dimensions, with accuracy improving with increasing reduced model dimension.



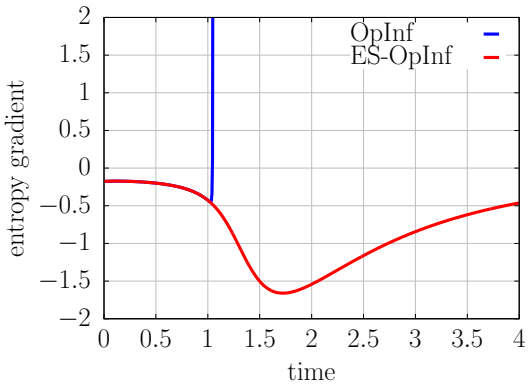
(a) dim 10 entropy



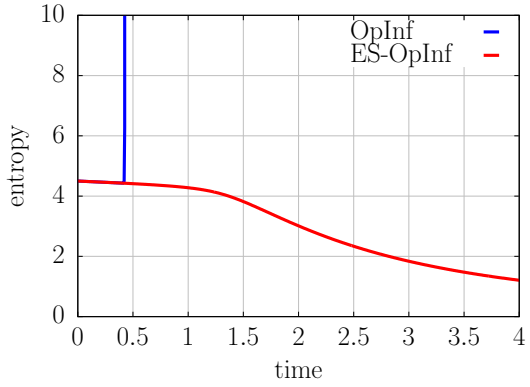
(b) dim 10 entropy gradient



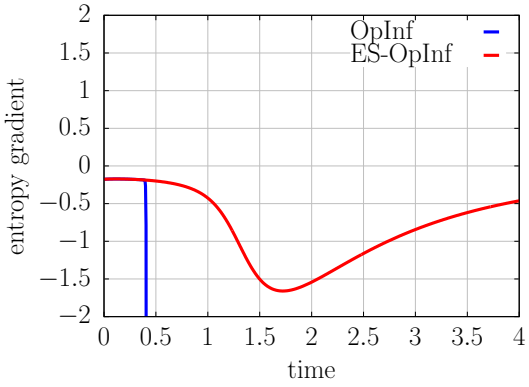
(c) dim 14 entropy



(d) dim 14 entropy gradient

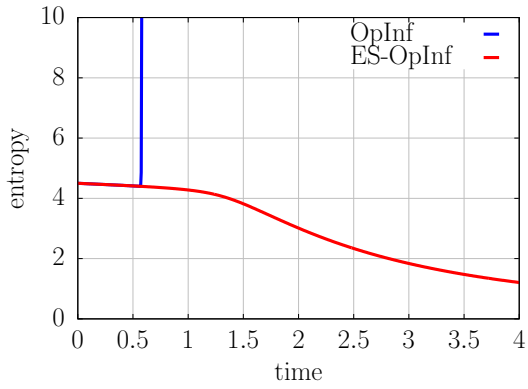


(e) dim 16 entropy

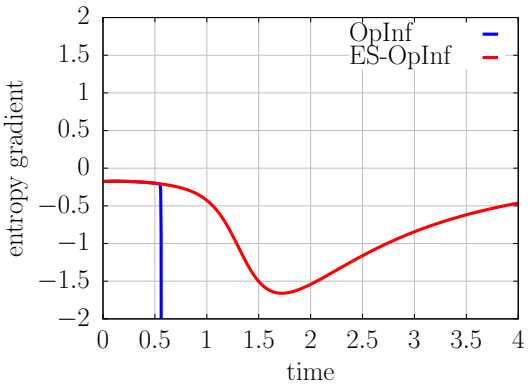


(f) dim 16 entropy gradient

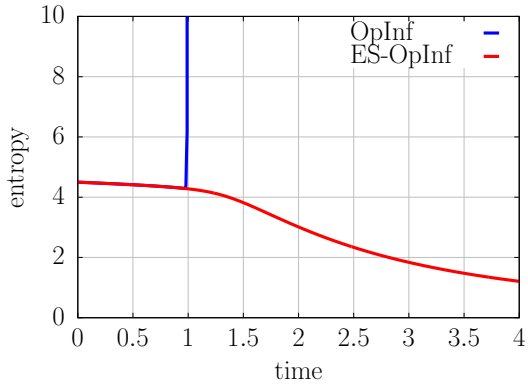
Figure 5.2: Burgers' equation: For dimensions $10, 14$ and 16, the reduced model learned via unconstrained OpInf is unstable, while the reduced model learned via ES-OpInf seems to satisfy entropy dissipation with time.



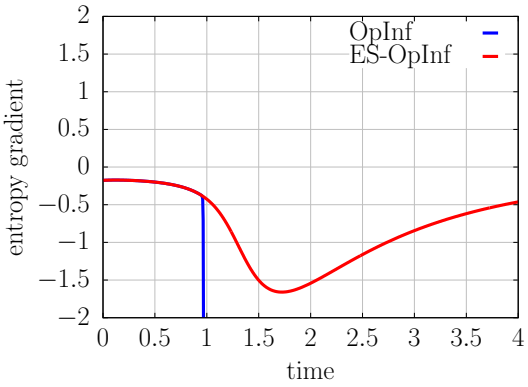
(a) dim 18 entropy



(b) dim 18 entropy gradient

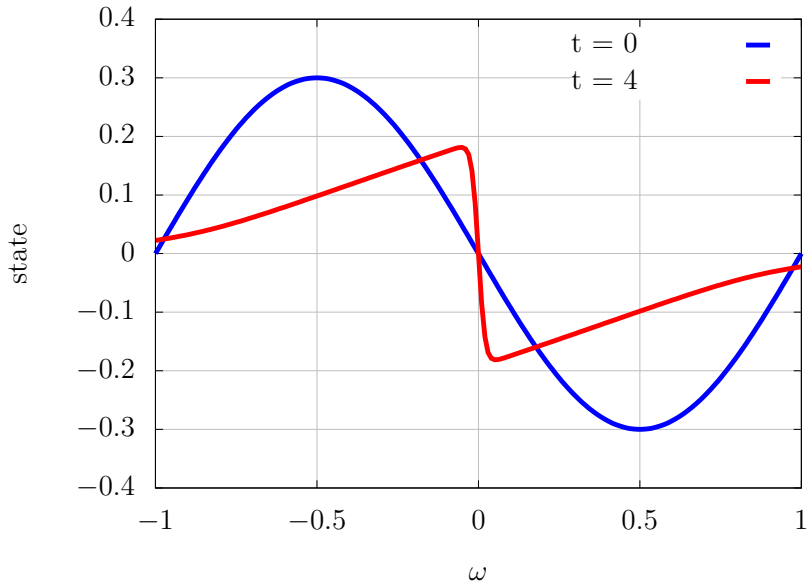


(c) dim 20 entropy

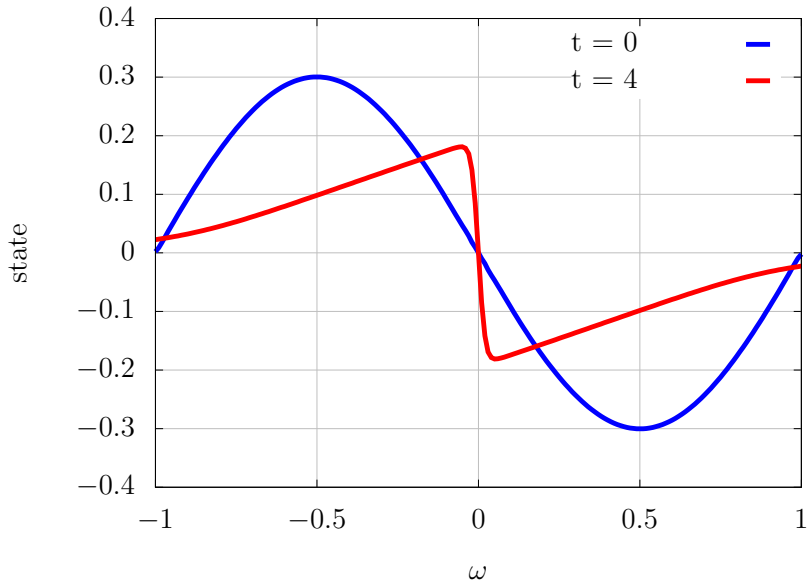


(d) dim 20 entropy gradient

Figure 5.3: Burgers' equation: Continuation of Figure 5.2 for dimensions 18 and 20.



(a) test state for full model



(b) test state for reduced model

Figure 5.4: Burgers' equation: State at time $t = 0$ and end time $t = 4$ is plotted for the full model and the reduced model with dimension 10, which is projected back onto the full model space. Results show that the reduced model learned via ES-OpInf when simulated forward in time generates test state trajectories similar to the full-model trajectories.

ϕ is the scalar potential of the fluid. We discretize spatial domain $[-10, 10]$ using $\Delta\xi = 0.2$ and finite volume scheme (5.9), with $N = 100$ grid points for both h and $h\phi$, and time domain $[0, 5]$ using explicit Euler with time-step size $\delta t = 10^{-3}$. We use parameter domains $D_1 = [\frac{1}{10}, \frac{1}{7}]$ and $D_2 = [\frac{2}{10}, \frac{15}{10}]$ for generating the initial conditions for the model. We select $M_b = 6$ equidistant parameters $\alpha_i \in D_1,\ \beta_i \in D_2,$ for $i = 1, \ldots, M_b$, and use initial conditions $\boldsymbol{h}^b_{i,0} = 1 + \alpha_i e^{-\beta_i x^2},\ \boldsymbol{\phi}^b_{i,0} = 0,$ for $i = 1, \ldots, M_b$, to generate basis trajectories used to construct the reduced space. Figure 5.5a plots the normalized singular values for the model.

For $M_t = 10$, we select $M_t + 2$ equidistant parameters $\alpha_i \in D_1,\ \beta_i \in D_2,$ for $i = 1, \ldots, M_t + 2$, and use initial conditions $\boldsymbol{h}_{i,0} = 1 + \alpha_i e^{-\beta_i x^2},\ \boldsymbol{\phi}_{i,0} = 0,$ for $i = 2, \ldots, M_t + 1$, to generate training trajectories. For the training trajectories, we use snapshots at every 10 time steps when training the reduced model. We test the model on a test trajectory with initial condition $\boldsymbol{h}^{\text{test}}_{i,0} = 1 + \alpha e^{-\beta x^2},\ \boldsymbol{\phi}^{\text{test}}_{i,0} = 0,$ where $\alpha = 0.1676, \beta = 2.25$ lie outside the training and basis parameter range. As seen in Figure 5.5b, the reduced model learned via unconstrained operator inference is unstable for all dimensions except 2 and 14, while the reduced model learned via EC-OpInf is stable for all dimensions.

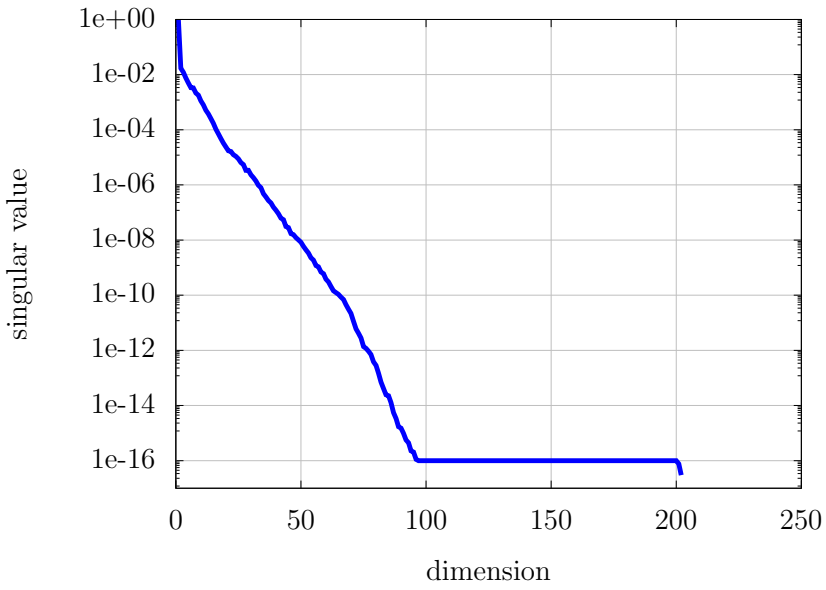
The reduced model is iterated with the projected test initial condition

$$\bar{\boldsymbol{q}}^{\text{test}}_0 = \boldsymbol{V}^T \begin{bmatrix} \boldsymbol{h}^{\text{test}}_{i,0} \\ \boldsymbol{\phi}^{\text{test}}_{i,0} \end{bmatrix},$$

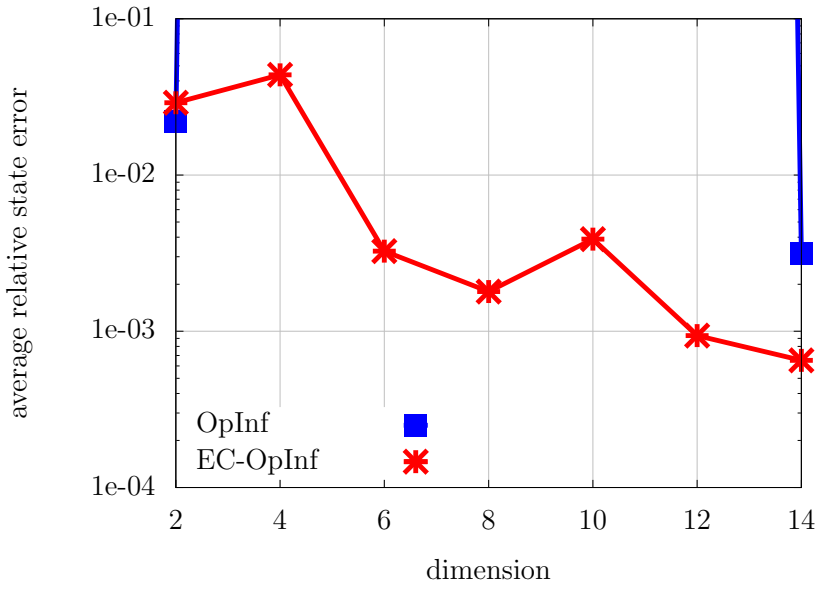
to generate reduced test state trajectory $\hat{\boldsymbol{Q}}^{\text{test}}$ and corresponding reduced test entropy variable trajectory $\hat{\boldsymbol{W}}^{\text{test}}$. Figures 5.6 and 5.7 plot the entropy function summed over all nodes $\mathbf{1}^T S(\hat{\boldsymbol{Q}}^{\text{test}})$ and the entropy gradient $\mathbf{1}^T dS(\hat{\boldsymbol{Q}}^{\text{test}})/dt$, at all time steps, for reduced models of dimension 4 to 12. For the reduced model learned via EC-OpInf, the entropy value hardly changes with time and we obtain $\mathbf{1}^T dS(\hat{\boldsymbol{Q}}^{\text{test}})/dt \approx 0$; which indicates that the model is

entropy conserving. For the reduced model learned via unconstrained operator inference, the entropy blows up, thus the model is unstable.

Figure 5.8 plots the test state trajectory for both h and $h\phi$ variables, for the full and reduced model at $t = 0$ and $t = 5$. For the reduced model with dimension 14, the reduced state trajectory projected back onto the full model space $\boldsymbol{V}\hat{\boldsymbol{Q}}^{\text{test}}$ is plotted. As seen in Figure 5.8, the state trajectory for the reduced model is similar to the full model, thus showing that reduced model accurately captures the full model behavior.

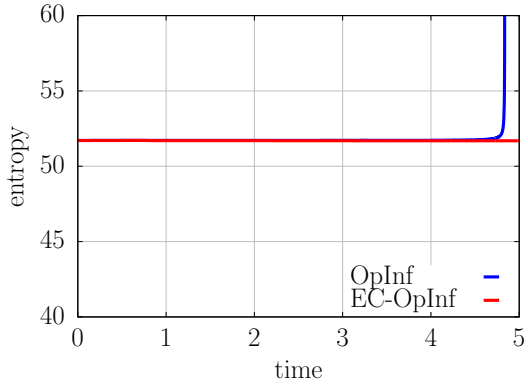


(a) decay of singular values

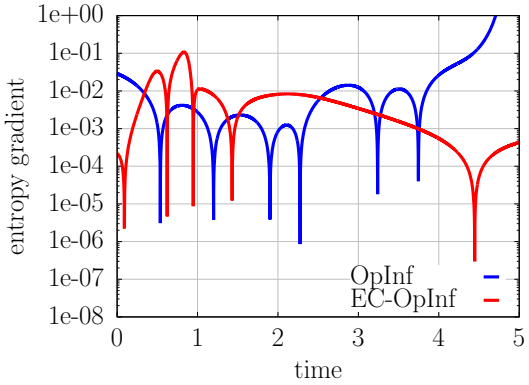


(b) test error

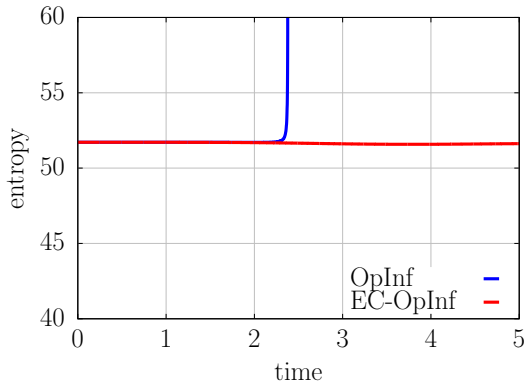
Figure 5.5: Shallow water equation: The reduced model learned via unconstrained OpInf is unstable for all dimensions except 2 and 14, while the reduced model learned via EC-OpInf is stable for all dimensions.



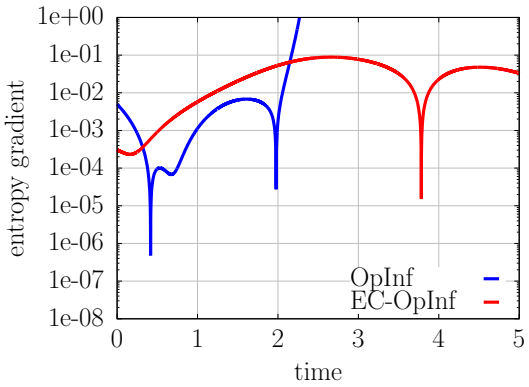
(a) dim 4 entropy



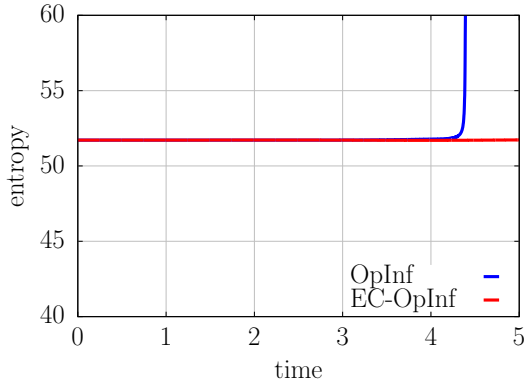
(b) dim 4 entropy gradient



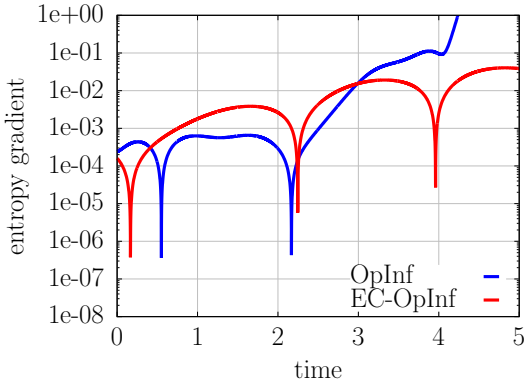
(c) dim 6 entropy



(d) dim 6 entropy gradient

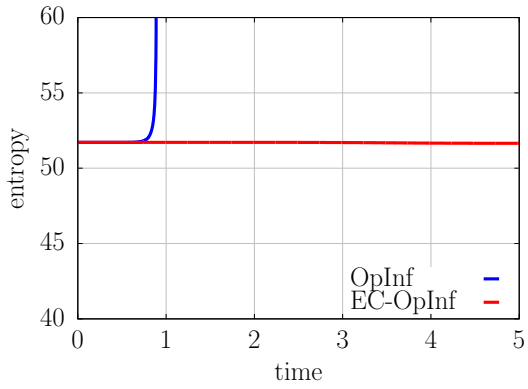


(e) dim 8 entropy

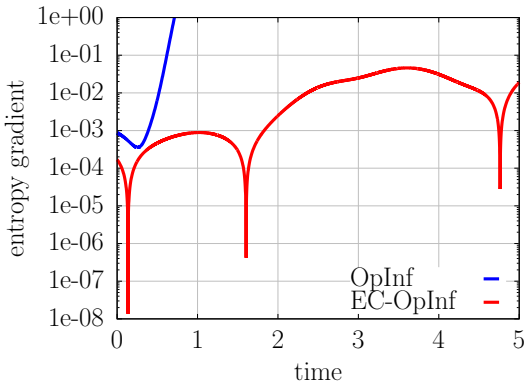


(f) dim 8 entropy gradient

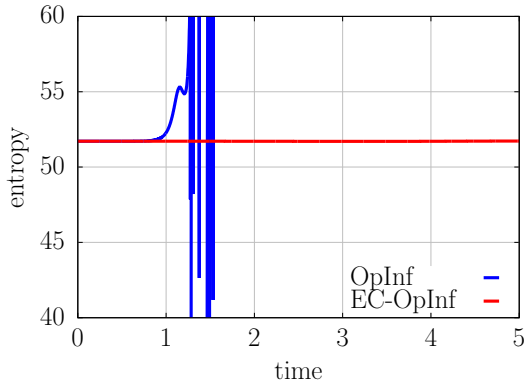
Figure 5.6: Shallow water equation: For dimensions 4 to 8, the reduced model learned via unconstrained OpInf is unstable, while the reduced model learned via EC-OpInf satisfies shows little growth in the entropy over time.



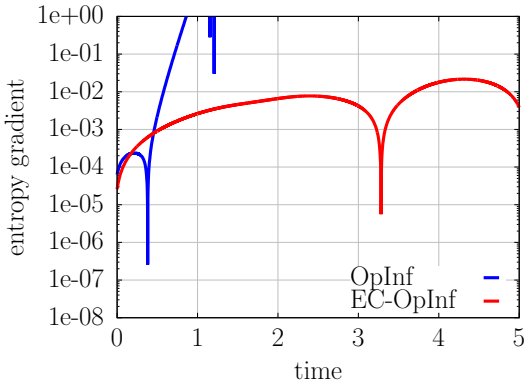
(a) dim 10 entropy



(b) dim 10 entropy gradient

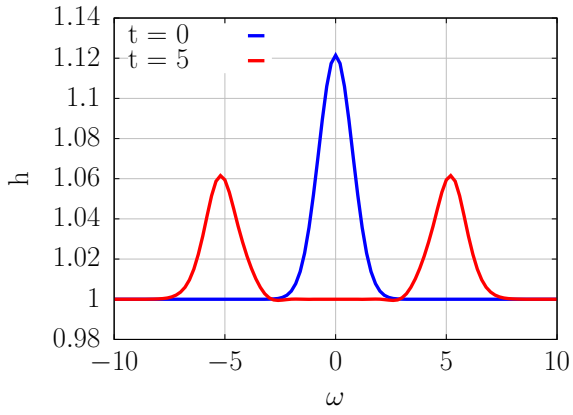


(c) dim 12 entropy

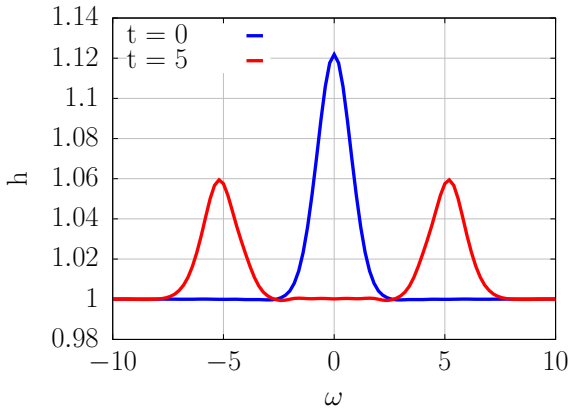


(d) dim 12 entropy gradient

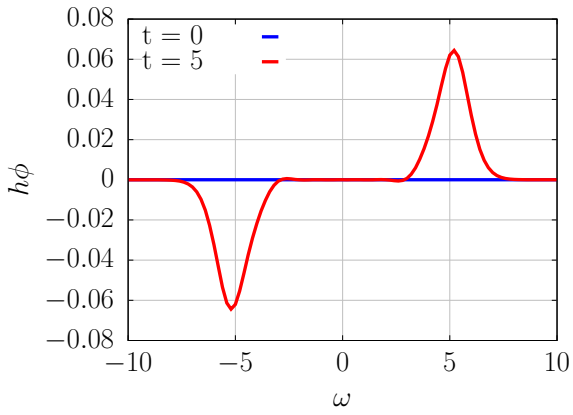
Figure 5.7: Shallow water equation: Continuation of Figure 5.6 for dimensions 10 and 12.



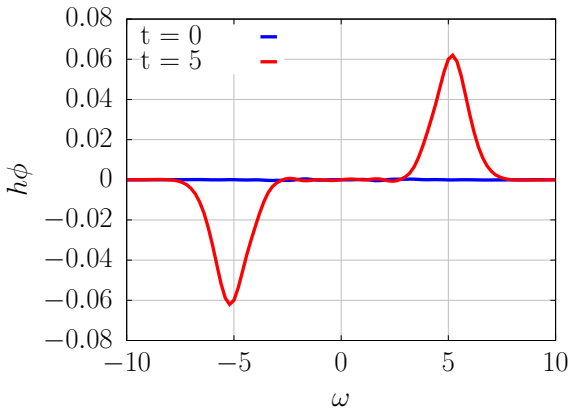
(a) test h for full model



(b) test h for reduced model



(c) test $h\phi$ for full model



(d) test $h\phi$ for reduced model

Figure 5.8: Shallow water equation: State at time $t = 0$ and end time $t = 5$ is plotted for the full model and the reduced model with dimension 14, which is projected back onto the full model space. Results show that the reduced model learned via EC-OpInf when iterated generates test state trajectories similar to the full model.

Chapter 6

Summary of contributions and outlook

6.1 Summary of contributions

For high-dimensional dynamical systems that are computationally expensive to evaluate, model reduction derives low-dimensional reduced models that are quicker to simulate. Classical projection-based model reduction is intrusive in nature as it requires access to full model operators. We used operator inference, which is non-intrusive in nature, to learn reduced models from data trajectories and physical insights from a gray box model. Based on the Lyapunov stability analysis of nonlinear differential equations, we developed a physics-informed regularizer that penalizes the norm of nonlinear operators of the reduced model, to increase its estimated stability radius. We demonstrated operator inference with physics-informed regularizer (PIR-OpInf) on dynamical systems with quadratic and cubic terms, and showed that PIR-OpInf increases the stability radius of the reduced models, and learns more stable and accurate reduced models compared OpInf without regularization and OpInf with Tikhonov regularization (T-OpInf).

We added constraints to PIR-OpInf to preserve the underlying structure of the dynamical system in the reduced model, and implemented operator inference with physics-informed regularizer and structure preservation (SPIR-OpInf). We showed that SPIR-OpInf preserved the symmetric negative definiteness property of the linear operators, which PIR-OpInf, T-OpInf and OpInf without regularization failed to do. We demonstrated SPIR-OpInf on dynamical systems with quadratic and cubic terms, and showed that SPIR-OpInf learns more stable and accurate reduced models compared to T-OpInf and OpInf without regularization.

We preserved the entropy stability and entropy conservation properties of a dynamical system of nonlinear conservation laws. We implemented operator inference with entropy conservation (EC-OpInf) with equality constraints to conserve the entropy for the training trajectories, and operator inference with entropy stability (ES-OpInf) with $\leq \mathbf{0}$ constraints to dissipate the entropy for the training trajectories. We demonstrated EC-OpInf and ES-

OpInf on dynamical systems with quadratic and cubic fluxes, and showed that EC-OpInf conserved the reduced model entropy and ES-OpInf dissipated the reduced model entropy, and both learned a more stable model compared to OpInf without any entropy constraints.

6.2 Outlook

For EC-OpInf and ES-OpInf, we conserve and dissipate the entropy only for the training data, and thus we cannot guarantee that the entropy constraints will hold true for the test data. For the numerical examples we studied, entropy conservation for the test trajectories holds true for reduced model learned via EC-OpInf, and entropy stability for the test trajectories holds true for the reduced model learned via ES-OpInf, however it is not provably guaranteed. A possible solution would be identifying certain structure in the model operators that enforces entropy conservation and entropy stability and preserving this structure using constraints when learning the reduced model via operator inference and also when interpolating the operators at new parameters for a parameterized full model.

For both ES-OpInf and EC-OpInf, the constraint matrix and the data matrix scale with n^{R+1} and n^{R+2} respectively, where n is the dimension of the reduced model, and R is the degree of the highest degree nonlinear term in the system. Thus, ES-OpInf and EC-OpInf do not scale well because the constraint and data matrices grow exponentially in model dimension. One possible solution is using the lift & learn approach [QFW22, Qia21] to transform a system of nonlinear conservation laws with higher degree nonlinear fluxes into a quadratic system, to limit the scaling with respect the reduced model dimension.

Bibliography

[AA86] A C Antoulas and B D O Anderson. On the scalar rational interpolation problem. *IMA Journal of Mathematical Control and Information*, 3(2-3):61–88, 1986.

[ABG21] Athanasios C. Antoulas, Christopher A. Beattie, and S. Gugercin. *Interpolatory Methods for Model Reduction*. SIAM, 2021.

[AF08] David Amsallem and Charbel Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.

[AF12] David Amsallem and Charbel Farhat. Stabilization of projection-based reduced-order models. *International Journal for Numerical Methods in Engineering*, 91(4):358–377, 2012.

[AGI16] Athanasios C Antoulas, Ion Victor Gosea, and Antonio Cosmin Ionita. Model reduction of bilinear systems in the Loewner framework. *SIAM Journal on Scientific Computing*, 38(5):B889–B916, 2016.

[AH17] Babak Maboudi Afkham and Jan S Hesthaven. Structure preserving model reduction of parametric hamiltonian systems. *SIAM Journal on Scientific Computing*, 39(6):A2616–A2644, 2017.

[AKJ08] Steven S An, Theodore Kim, and Doug L James. Optimizing cubature for efficient integration of subspace deformations. *ACM transactions on graphics (TOG)*, 27(5):1–10, 2008.

[Ant05] Athanasios C Antoulas. *Approximation of large-scale dynamical systems*. SIAM, 2005.

[ARWH20] Babak Maboudi Afkham, Nicolo Ripamonti, Qian Wang, and Jan S Hesthaven. Conservative model order reduction for fluid flow. *Quantification of Uncertainty: Improving Efficiency and Technology: QUIET selected contributions*, pages 67–99, 2020.

[AWWB08] P. Astrid, S. Weiland, K. Willcox, and T. Backx. Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251, 2008.

[AZW15] David Amsallem, Matthew J. Zahr, and Kyle Washabaugh. Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction. *Advances in Computational Mathematics*, 41(5):1187–1230, Oct 2015.

[BB12] T Breiten and P Benner. Interpolation-based h2-model reduction of bilinear control system. *SIAM J. Matrix Anal. Appl*, 33(3):859–885, 2012.

[BB15] Peter Benner and Tobias Breiten. Two-sided projection methods for nonlinear model order reduction. *SIAM Journal on Scientific Computing*, 37(2):B239–B260, 2015.

[BBPK16] Steven L Brunton, Bingni W Brunton, Joshua L Proctor, and J Nathan Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS one*, 11(2):e0150171, 2016.

[BG21] Peter Benner and Pawan Goyal. Interpolation-based model order reduction for polynomial systems. *SIAM Journal on Scientific Computing*, 43(1):A84–A108, 2021.

[BGG18] P. Benner, P. Goyal, and S. Gugercin. H2-quasi-optimal model order reduction for quadratic-bilinear control systems. *SIAM Journal on Matrix Analysis and Applications*, 39(2):983–1032, 2018.

[BGHD22] Peter Benner, Pawan Goyal, Jan Heiland, and Igor Pontes Duff. Operator inference and physics-informed learning of low-dimensional models for incompressible flows. *Electronic Transactions on Numerical Analysis*, 56:28–51, 2022.

[BGK+20] Peter Benner, Pawan Goyal, Boris Kramer, Benjamin Peherstorfer, and Karen Willcox. Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms. *Computer Methods in Applied Mechanics and Engineering*, 372:113433, 2020.

[BGW15] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.

[BHL93] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[BKST09] Matthew F Barone, Irina Kalashnikova, Daniel J Segalman, and Heidi K Thornquist. Stable galerkin reduced order models for linearized compressible flow. *Journal of Computational Physics*, 228(6):1932–1946, 2009.

[BMQR15] Francesco Ballarin, Andrea Manzoni, Alfio Quarteroni, and Gianluigi Rozza. Supremizer stabilization of pod–galerkin approximation of parametrized steady incompressible navier–stokes equations. *International Journal for Numerical Methods in Engineering*, 102(5):1136–1161, 2015.

[BNMP04] M. Barrault, N. C. Nguyen, Y. Maday, and A. T. Patera. An empirical interpolation method: Application to efficient reduced-basis discretization of partial differential equations. *C. R. Acad. Sci. Paris, Série I.*, 339:667–672, 2004.

[BPK16] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[BTD16] Maciej Balajewicz, Irina Tezaur, and Earl Dowell. Minimal subspace rotation on the stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible navier–stokes equations. *Journal of Computational Physics*, 321:224–241, 2016.

[Car15] Kevin Carlberg. Adaptive h-refinement for reduced-order models. *International Journal for Numerical Methods in Engineering*, 102(5):1192–1210, 2015.

[CBA17] Kevin Carlberg, Matthew Barone, and Harbir Antil. Galerkin v. least-squares petrov–galerkin projection in nonlinear model reduction. *Journal of Computational Physics*, 330:693–734, 2017.

[CBG16] Saifon Chaturantabut, Chris Beattie, and Serkan Gugercin. Structure-

preserving model reduction for nonlinear port-hamiltonian systems. *SIAM Journal on Scientific Computing*, 38(5):B837–B865, 2016.

[CBMF11] Kevin Carlberg, Charbel Bou-Mosleh, and Charbel Farhat. Efficient nonlinear model reduction via a least-squares Petrov–Galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2):155–181, 2011.

[CFCA13] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem. The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows. *Journal of Computational Physics*, 242:623–647, 2013.

[Cha20] Jesse Chan. Entropy stable reduced order modeling of nonlinear conservation laws. *Journal of Computational Physics*, 423:109789, 2020.

[Che07] Graziano Chesi. Estimating the domain of attraction via union of continuous families of Lyapunov estimates. *Systems & Control Letters*, 56(4):326–333, 2007.

[CHK02] Alexandre Chorin, Ole H. Hald, and Raz Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166(3):239 – 257, 2002.

[CIJS14] Alfonso Caiazzo, Traian Iliescu, Volker John, and Swetlana Schyschlowa. A numerical investigation of velocity–pressure reduced order models for incompressible flows. *Journal of Computational Physics*, 259:598–616, 2014.

[CKMP20] Alice Cortinovis, Daniel Kressner, Stefano Massei, and Benjamin Peherstorfer. Quasi-optimal sampling to learn basis updates for online adaptive model reduction with adaptive empirical interpolation. In *2020 American Control Conference (ACC)*, pages 2472–2477, 2020.

[CL96] Thomas F Coleman and Yuying Li. A reflective newton method for minimizing a quadratic function subject to bounds on some of the variables. *SIAM Journal on Optimization*, 6(4):1040–1058, 1996.

[CS10] S. Chaturantabut and D.C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[CTB15] Kevin Carlberg, Ray Tuminaro, and Paul Boggs. Preserving lagrangian structure in nonlinear model reduction with application to structural dynamics. *SIAM Journal on Scientific Computing*, 37(2):B153–B184, 2015.

[DG16] Z. Drmač and S. Gugercin. A new selection operator for the Discrete Empirical Interpolation Method – improved a priori error bound and extensions. *SIAM Journal on Scientific Computing*, 38(2):A631–A648, 2016.

[DGB15] Z. Drmač, S. Gugercin, and C. Beattie. Quadrature-based vector fitting for discretized $\mathcal{H}_2$ approximation. *SIAM Journal on Scientific Computing*, 37(2):A625–A652, 2015.

[DP22] Zlatko Drmač and Benjamin Peherstorfer. Learning low-dimensional dynamical-system models from noisy frequency-response data with Loewner rational interpolation. In Christopher Beattie, Peter Benner, Mark Embree, Serkan Gugercin, and Sanda Lefteriu, editors, *Realization and Model Reduction of Dynamical Systems: A Festschrift in Honor of the 70th Birthday of Thanos Antoulas*, pages 39–57, Cham, 2022. Springer International Publishing.

[DVW10] Joris Degroote, Jan Vierendeels, and Karen Willcox. Interpolation among reduced-order matrices to obtain parameterized models for design, optimiza-

tion and probabilistic analysis. *International Journal for Numerical Methods in Fluids*, 63(2):207–230, 2010.

[EI22] Mark Embree and A. Cosmin Ionita. Pseudospectra of Loewner matrix pencils. In Christopher Beattie, Peter Benner, Mark Embree, Serkan Gugercin, and Sanda Lefteriu, editors, *Realization and Model Reduction of Dynamical Systems: A Festschrift in Honor of the 70th Birthday of Thanos Antoulas*, pages 59–78, Cham, 2022. Springer International Publishing.

[ES95] R. Everson and L. Sirovich. The Karhunen-Loeve Procedure for Gappy Data. *Journal of the Optical Society of America*, 12:1657–1664, 1995.

[FACC14] Charbel Farhat, Philip Avery, Todd Chapman, and Julien Cortial. Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency. *International Journal for Numerical Methods in Engineering*, 98(9):625–662, 2014.

[FCA15] Charbel Farhat, Todd Chapman, and Philip Avery. Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models. *International journal for numerical methods in engineering*, 102(5):1077–1110, 2015.

[FGMW23] Ionut Farcaș, Rayomand Gundevia, Ramakanth Munipalli, and Karen E. Willcox. Parametric non-intrusive reduced-order models via operator inference for large-scale rotating detonation engine simulations. In *AIAA SCITECH 2023 Forum*, 2023.

[FMW22] Ionut Farcaș, Ramakanth Munipalli, and Karen E. Willcox. On filtering in non-intrusive data-driven reduced-order modeling. In *AIAA AVIATION 2022 Forum*, 2022.

[FPN+23] I.G. Farcas, B. Peherstorfer, T. Neckel, F. Jenko, and H.J. Bungartz. Context-aware learning of hierarchies of low-fidelity models for multi-fidelity uncertainty quantification. *Computer Methods in Applied Mechanics and Engineering*, 2023.

[GA16] Ion Victor Gosea and Athanasios C. Antoulas. Stability preserving post-processing methods applied in the Loewner framework. In *2016 IEEE 20th Workshop on Signal and Power Integrity (SPI)*, pages 1–4, 2016.

[GA18] Ion Victor Gosea and Athanasios C Antoulas. Data-driven model order reduction of quadratic-bilinear systems. *Numerical Linear Algebra with Applications*, 25(6):e2200, 2018.

[GB08] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. http://stanford.edu/~boyd/graph_dcp.html.

[GB14] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014.

[GKA21] Ion Victor Gosea, Dimitrios S. Karachalios, and Athanasios C. Antoulas. Learning reduced-order models of quadratic dynamical systems from input-output data. In *2021 European Control Conference (ECC)*, pages 1426–1431, 2021.

[GMW19] Philip E Gill, Walter Murray, and Margaret H Wright. *Practical optimization*. SIAM, 2019.

[GMW22] Mengwu Guo, Shane A McQuarrie, and Karen E Willcox. Bayesian operator inference for data-driven reduced-order modeling. *Computer Methods in Applied Mechanics and Engineering*, 402:115336, 2022.

[GPA18] I. V. Gosea, M. Petreczky, and A. C. Antoulas. Data-driven model order reduction of linear switched systems in the Loewner framework. *SIAM Journal on Scientific Computing*, 40(2):B572–B610, 2018.

[GPA21] Ion Victor Gosea, Mihaly Petreczky, and Athanasios C. Antoulas. Reduced-order modeling of LPV systems in the Loewner framework. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 3299–3305, 2021.

[GPBVDS12] Serkan Gugercin, Rostyslav V Polyuga, Christopher Beattie, and Arjan Van Der Schaft. Structure-preserving tangential interpolation for model reduction of port-hamiltonian systems. *Automatica*, 48(9):1963–1974, 2012.

[GS99] B. Gustavsen and A. Semlyen. Rational approximation of frequency domain responses by vector fitting. *Power Delivery, IEEE Transactions on*, 14(3):1052–1061, Jul 1999.

[GT89] R Genesio and A Tesi. Stability analysis of quadratic systems. *IFAC Proceedings Volumes*, 22(3):195–199, 1989.

[Gu11] Chenjie Gu. QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(9):1307–1320, 2011.

[GWW17] Yuezheng Gong, Qi Wang, and Zhu Wang. Structure-preserving galerkin pod reduced-order modeling of hamiltonian systems. *Computer Methods in Applied Mechanics and Engineering*, 315:780–798, 2017.

[HCF17] Joaquin Alberto Hernandez, Manuel Alejandro Caicedo, and Alex Ferrer. Dimensional hyper-reduction of nonlinear finite element models via empirical cubature. *Computer methods in applied mechanics and engineering*, 313:687–722, 2017.

[Hig88] Nicholas J. Higham. Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications*, 103:103–118, 1988.

[HP05] Diederich Hinrichsen and Anthony J Pritchard. *Mathematical systems theory I: modelling, state space analysis, stability and robustness*, volume 134, page 529. Springer, 2005.

[HPR22] Jan S. Hesthaven, Cecilia Pagliantini, and Gianluigi Rozza. Reduced basis methods for time-dependent problems. *Acta Numerica*, 31:265–345, 2022.

[HRS16] Jan S Hesthaven, Gianluigi Rozza, and Benjamin Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer International Publishing, 2016.

[IA14] Antonio Cosmin Ionita and Athanasios C Antoulas. Data-driven parametrized model reduction in the Loewner framework. *SIAM Journal on Scientific Computing*, 36(3):A984–A1007, 2014.

[KB10] Irina Kalashnikova and Matthew F Barone. On the stability and convergence of a galerkin reduced order model (rom) of compressible flow with solid wall and far-field boundary treatment. *International journal for numerical methods in engineering*, 83(10):1345–1375, 2010.

[KBAvBW14] Irina Kalashnikova, Matthew F Barone, Srinivasan Arunajatesan, and Bart G van Bloemen Waanders. Construction of energy-stable projection-based reduced order models. *Applied Mathematics and Computation*, 249:569–596, 2014.

[KBBP16] J Nathan Kutz, Steven L Brunton, Bingni W Brunton, and Joshua L Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

[KG16] B. Kramer and S. Gugercin. The eigensystem realization algorithm from tangentially interpolated data. *Mathematical and Computer Modelling of Dynamical Systems*, 2016. to appear.

[KGA21] D. S. Karachalios, I. V. Gosea, and A. C. Antoulas. *On Bilinear Time-Domain Identification and Reduction in the Loewner Framework*, pages 3–30. Springer International Publishing, Cham, 2021.

[KPW17] B. Kramer, B. Peherstorfer, and K. Willcox. Feedback control for systems with uncertain parameters using online-adaptive reduced models. *SIAM Journal on Applied Dynamical Systems*, 16(3):1563–1586, 2017.

[Kra21] Boris Kramer. Stability domains for quadratic-bilinear reduced-order models. *SIAM Journal on Applied Dynamical Systems*, 20(2):981–996, 2021.

[KvAB14] Irina Kalashnikova, Bart van Bloemen Waanders, Srinivasan Arunajatesan, and Matthew Barone. Stabilization of projection-based reduced order models for linear time-invariant systems via optimization-based eigenvalue reassignment. *Computer Methods in Applied Mechanics and Engineering*, 272:251–270, 2014.

[KW19] B. Kramer and K. Willcox. Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition. *AIAA Journal*, 57(6):2297–2307, 2019.

[KW22] Parisa Khodabakhshi and Karen E Willcox. Non-intrusive data-driven model reduction for differential–algebraic equations derived from lifting transformations. *Computer Methods in Applied Mechanics and Engineering*, 389:114296, 2022.

[Kö14] M. Köhler. On the closest stable descriptor system in the respective spaces RH_2 and RH_∞. *Linear Algebra and its Applications*, 443:34–49, 2014.

[LBCK14] Zhen Li, Xin Bian, Bruce Caswell, and George Em Karniadakis. Construction of dissipative particle dynamics models for complex fluids via the Mori–Zwanzig formulation. *Soft Matter*, 10:8659–8672, 2014.

[LCV17] S. Le Clainche and J. Vega. Higher order dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 16(2):882–925, 2017.

[LIA10] Sanda Lefteriu, Antonio C. Ionita, and Athanasios C. Antoulas. Modeling systems based on noisy frequency and time domain measurements. In Jan C. Willems, Shinji Hara, Yoshito Ohta, and Hisaya Fujioka, editors, *Perspectives in Mathematical System Theory, Control, and Signal Processing: A Festschrift in Honor of Yutaka Yamamoto on the Occasion of his 60th Birthday*, pages 365–378, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[Lin19] Zhenhua Lin. Riemannian geometry of symmetric positive definite matrices via Cholesky decomposition. *SIAM Journal on Matrix Analysis and Applications*, 40(4):1353–1370, 2019.

[LKM03] Sanjay Lall, Petr Krysl, and Jerrold E Marsden. Structure-preserving model reduction for mechanical systems. *Physica D: Nonlinear Phenomena*, 184(1-4):304–318, 2003.

[MA07] A J Mayo and A C Antoulas. A framework for the solution of the generalized realization problem. *Linear algebra and its applications*, 425(2-3):634–662, 2007.

[MAH19] Babak Maboudi Afkham and Jan S Hesthaven. Structure-preserving model-reduction of dissipative hamiltonian systems. *Journal of Scientific Computing*, 81(1):3–21, 2019.

[MBK17] Krithika Manohar, Steven L. Brunton, and J. Nathan Kutz. Environment identification in flight using sparse approximation of wing strain. *Journal of Fluids and Structures*, 70:162 – 180, 2017.

[McC76] Garth P McCormick. Computability of global solutions to factorable nonconvex programs: Part i–convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[Mez05] Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1-3):309–325, 2005.

[MHW21] Shane A. McQuarrie, Cheng Huang, and Karen Willcox. Data-driven reduced-order models via regularised operator inference for a single-injector combustion process. *Journal of the Royal Society of New Zealand*, 51(2):194–211, 2021.

[MM13] Yvon Maday and Olga Mula. A generalized empirical interpolation method: Application of reduced basis techniques to data assimilation. In Franco Brezzi,

Piero Colli Franzone, Ugo Gianazza, and Gianni Gilardi, editors, *Analysis and Numerics of Partial Differential Equations*, pages 221–235. Springer, 2013.

[MMPY15] Y. Maday, O. Mula, A.T. Patera, and M. Yano. The generalized empirical interpolation method: Stability theory on Hilbert spaces with an application to the Stokes equation. *Computer Methods in Applied Mechanics and Engineering*, 287:310–334, 2015.

[MPR01] Yvon Maday, Anthony T Patera, and Dimitrios V Rovas. A blackbox reduced-basis output bound method for noncoercive linear problems. 2001.

[NST18] Yuji Nakatsukasa, Olivier Sète, and Lloyd N. Trefethen. The AAA algorithm for rational approximation. *SIAM Journal on Scientific Computing*, 40(3):A1494–A1522, 2018.

[PBWB14] Benjamin Peherstorfer, Daniel Butnaru, Karen Willcox, and Hans-Joachim Bungartz. Localized discrete empirical interpolation method. *SIAM Journal on Scientific Computing*, 36(1):A168–A192, 2014.

[PD18] S. Pan and K. Duraisamy. Data-driven discovery of closure models. *SIAM Journal on Applied Dynamical Systems*, 17(4):2381–2413, 2018.

[PDG20] B. Peherstorfer, Z. Drmac, and S. Gugercin. Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points. *SIAM Journal on Scientific Computing*, 42:A2837–A2864, 2020.

[Peh20a] Benjamin Peherstorfer. Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling. *SIAM Journal on Scientific Computing*, 42(5):A2803–A2836, 2020.

[Peh20b] Benjamin Peherstorfer. Sampling low-dimensional markovian dynamics for preasymptotically recovering reduced models from data with operator inference. *SIAM Journal on Scientific Computing*, 42(5):A3489–A3515, 2020.

[Peh22] B. Peherstorfer. Breaking the Kolmogorov barrier with nonlinear model reduction. *Notices of the American Mathematical Society*, 69:725–733, 2022.

[PGW17] Benjamin Peherstorfer, Serkan Gugercin, and Karen Willcox. Data-driven reduced model construction with time-domain Loewner models. *SIAM Journal on Scientific Computing*, 39(5):A2152–A2178, 2017.

[PM16] Liqian Peng and Kamran Mohseni. Symplectic model reduction of hamiltonian systems. *SIAM Journal on Scientific Computing*, 38(1):A1–A27, 2016.

[PW15a] Benjamin Peherstorfer and Karen Willcox. Dynamic data-driven reduced-order models. *Computer Methods in Applied Mechanics and Engineering*, 291:21–41, 2015.

[PW15b] Benjamin Peherstorfer and Karen Willcox. Online adaptive model reduction for nonlinear systems via low-rank updates. *SIAM Journal on Scientific Computing*, 37(4):A2123–A2150, 2015.

[PW16a] B. Peherstorfer and K. Willcox. Dynamic data-driven model reduction: Adapting reduced models from incomplete data. *Advanced Modeling and Simulation in Engineering Sciences*, 3(11), 2016.

[PW16b] Benjamin Peherstorfer and Karen Willcox. Data-driven operator inference for nonintrusive projection-based model reduction. *Computer Methods in Applied Mechanics and Engineering*, 306:196–215, 2016.

[QFW22] Elizabeth Qian, Ionut-Gabriel Farcaș, and Karen Willcox. Reduced operator inference for nonlinear partial differential equations. *SIAM Journal on Scientific Computing*, 44(4):A1934–A1959, 2022.

[Qia21] Elizabeth Qian. *A scientific machine learning approach to learning reduced models for nonlinear partial differential equations*. PhD thesis, Massachusetts Institute of Technology, 2021.

[Qin06] S. Joe Qin. An overview of subspace identification. *Computers & Chemical Engineering*, 30(10–12):1502 – 1513, 2006.

[QKMW19] Elizabeth Qian, Boris Kramer, Alexandre Marques, and Karen Willcox. Transform & learn: A data-driven approach to nonlinear model reduction. In *AIAA Aviation 2019 Forum*, pages 1–11. AIAA, 2019.

[QKPW20] Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, and Karen Willcox. Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. *Physica D: Nonlinear Phenomena*, 406:132401, 2020.

[QR14] Alfio Quarteroni and Gianluigi Rozza, editors. *Reduced order methods for modeling and computational reduction*. Springer, 2014.

[QRM11] Alfio Quarteroni, Gianluigi Rozza, and Andrea Manzoni. Certified reduced basis approximation for parametrized partial differential equations and applications. *Journal of Mathematics in Industry*, 1(1):3, 2011.

[Rey12] Edwin Reynders. System identification methods for (operational) modal analysis: Review and comparison. *Archives of Computational Methods in Engineering*, 19(1):51–124, 2012.

[RHM13] Gianluigi Rozza, DB Phuong Huynh, and Andrea Manzoni. Reduced basis approximation and a posteriori error estimation for stokes flows in parametrized geometries: roles of the inf-sup stability constants. *Numerische Mathematik*, 125:115–152, 2013.

[RHP07] Gianluigi Rozza, Dinh Bao Phuong Huynh, and Anthony T Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15(3):1, 2007.

[RMB+09] CLARENCE W. Rowley, IGOR Mezic, SHERVIN Bagheri, PHILIPP Schlatter, and DAN S. Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009.

[RP03] Muruhan Rathinam and Linda R Petzold. A new look at proper orthogonal decomposition. *SIAM Journal on Numerical Analysis*, 41(5):1893–1925, 2003.

[RV07] Gianluigi Rozza and Karen Veroy. On the stability of the reduced basis method for stokes equations in parametrized domains. *Computer methods in applied mechanics and engineering*, 196(7):1244–1260, 2007.

[SBK15] Syuzanna Sargsyan, Steven L. Brunton, and J. Nathan Kutz. Nonlinear model reduction for dynamical systems using sparse sensor locations from learned libraries. *Phys. Rev. E*, 92:033304, Sep 2015.

[Sch10] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of fluid mechanics*, 656:5–28, 2010.

[Sch17] Hayden Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

[SCHO13] Hayden Schaeffer, Russel Caflisch, Cory D. Hauck, and Stanley Osher. Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences*, 110(17):6634–6639, 2013.

[SGP22] N. Shyamkumar, S. Gugercin, and B. Peherstorfer. Towards context-aware learning for control: Balancing stability and model-learning error. In *IEEE American Control Conference*, 2022.

[Sir87] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of applied mathematics*, 45(3):561–571, 1987.

[SK22] Harsh Sharma and Boris Kramer. Preserving Lagrangian structure in data-driven reduced-order modeling of large-scale mechanical systems. *arXiv preprint arXiv:2203.06361*, 2022.

[SKHW20] Renee Swischuk, Boris Kramer, Cheng Huang, and Karen Willcox. Learning physics-based reduced-order models for a single-injector combustion process. *AIAA Journal*, 58(6):2658–2672, 2020.

[SLGB12] Gilles Serre, Philippe Lafon, Xavier Gloerfelt, and Christophe Bailly. Reliable reduced-order models for time-dependent linearized euler equations. *Journal of Computational Physics*, 231(15):5176–5194, 2012.

[SU16] Philipp Schulze and Benjamin Unger. Data-driven interpolation of dynamical systems with delay. *Systems & Control Letters*, 97:125–131, 2016.

[SUBG18] Philipp Schulze, Benjamin Unger, Christopher Beattie, and Serkan Gugercin. Data-driven structured realization. *Linear Algebra and its Applications*, 537:250–286, 2018.

[SWK22] Harsh Sharma, Zhu Wang, and Boris Kramer. Hamiltonian operator inference: Physics-preserving learning of reduced-order models for Hamiltonian systems. *Physica D: Nonlinear Phenomena*, 431:133122, 2022.

[Tad87] Eitan Tadmor. The numerical viscosity of entropy stable schemes for systems of conservation laws. i. *Mathematics of Computation*, 49(179):91–103, 1987.

[Tad03] Eitan Tadmor. Entropy stability theory for difference approximations of nonlinear conservation laws and related time-dependent problems. *Acta Numerica*, 12:451–512, 2003.

[TGDW19] Erik H. Thiede, Dimitrios Giannakis, Aaron R. Dinner, and Jonathan Weare. Galerkin approximation of dynamical quantities using trajectory data. *The Journal of Chemical Physics*, 150(24):244111, 2019.

[Tod01] Michael J Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, 2001.

[TRL+14] Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, 2014.

[TVG94] A. Tesi, F. Villoresi, and R. Genesio. On stability domain estimation via a quadratic Lyapunov function: convexity and optimality properties for polynomial systems. In *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, volume 2, pages 1907–1912, 1994.

[TW17] Giang Tran and Rachel Ward. Exact recovery of chaotic systems from highly corrupted data. *Multiscale Modeling & Simulation*, 15(3):1108–1129, 2017.

[UHP22] Wayne Isaac Tan Uy, D. Hartmann, and B. Peherstorfer. Operator inference

with roll outs for learning reduced models from scarce and low-quality data. *arXiv:2212.01418*, 2022.

[UP21a] Wayne Isaac Tan Uy and Benjamin Peherstorfer. Operator inference of non-Markovian terms for learning reduced models from partially observed state trajectories. *Journal of Scientific Computing*, 88(3):91, Aug 2021.

[UP21b] Wayne Isaac Tan Uy and Benjamin Peherstorfer. Probabilistic error estimation for non-intrusive reduced models learned from data of systems governed by linear parabolic partial differential equations. *ESAIM: M2AN*, 55(3):735–761, 2021.

[UWWP21] Wayne Isaac Tan Uy, Yuepeng Wang, Yuxiao Wen, and Benjamin Peherstorfer. Active operator inference for learning low-dimensional dynamical-system models from noisy data. *arXiv preprint arXiv:2107.09256*, 2021.

[Vib95] Mats Viberg. Subspace-based methods for the identification of linear time-invariant systems. *Automatica*, 31(12):1835 – 1851, 1995. Trends in System Identification.

[WABI12] Zhu Wang, Imran Akhtar, Jeff Borggaard, and Traian Iliescu. Proper orthogonal decomposition closure models for turbulent flows: a numerical comparison. *Computer Methods in Applied Mechanics and Engineering*, 237:10–26, 2012.

[WKR15] Matthew O Williams, Ioannis G Kevrekidis, and Clarence W Rowley. A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

[WP22] S.W.R. Werner and B. Peherstorfer. On the sample complexity of stabilizing

linear dynamical systems from data. *Foundations of Computational Mathematics*, 2022.

[WP23] S.W.R. Werner and B. Peherstorfer. Context-aware controller inference for stabilizing dynamical systems from scarce data. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2023.

[ZPW18] R. Zimmermann, B. Peherstorfer, and K. Willcox. Geometric subspace updates with applications to online adaptive nonlinear model reduction. *SIAM Journal on Matrix Analysis and Applications*, 39(1):234–261, 2018.