RANK-MINIMIZING AND STRUCTURED MODEL INFERENCE[*]

PAWAN GOYAL[†], BENJAMIN PEHERSTORFER[‡], AND PETER BENNER[§]

Abstract. While machine-learning from data is becoming an ever more important part of science and engineering, physical insights and first principles continue to provide critical information about systems of interest. This work introduces a data-driven method for inferring models that encode physical insights in form of structure and have the minimal order that is necessary to fit well the training data. The models are formulated via solution matrices of specific instances of generalized Sylvester equations that enforce interpolation of the training data and at the same relate the model order to the rank of the solution matrices. The proposed method numerically solves the Sylvester equations for minimal-rank solutions to obtain parsimonious models, which ensures that the training data are fitted well while redundant degrees of freedom without conditions and sufficient data to fix them are eliminated. To make the training computationally efficient, the rank minimization is relaxed via weighted nuclear norms. Numerical experiments demonstrate that the combination of structure preservation and order minimization leads to accurate models with orders of magnitude fewer degrees of freedom than models of comparable prediction quality that are learned with structure preservation alone.

1. Introduction.

Learning models that describe the input-output response dynamics of systems from data is an ever more important building component in science and engineering. At the same time, in many applications, there is at least some knowledge available about the physics that are described by the systems of interest [13, 54]. Such physical insights often can be translated into structure, e.g., symmetries, time delays, and high-order time derivatives, which can then be imposed on the models to ensure physically meaningful input-output response predictions.

In this work, we introduce a method for inferring models that preserve the structure given by physical insights and, at the same time, minimizes the order of the model—the number of degrees of freedom of the model parametrization—that is necessary to fit well the training data. Critically, the order is determined during training and does not have to be specified a priori. Minimizing the model order means that our approach finds parsimonious models by adjusting the order during the training so that input-output data are fitted well while redundant degrees of freedom without conditions and sufficient data to fix them are eliminated. Correspondingly, the learned models have few degrees of freedom and thus can be simulated quickly to make predictions about the underlying systems. Additionally, the learned structured input-output models can be realized as systems that describe dynamics of states in the time domain and how they behave under control inputs, which is in contrast to black-box models that only match the input-output behavior. Numerical experiments demonstrate that the combination of structure preservation and order minimization leads to accurate models with orders of magnitude fewer degrees of freedom than models of comparable prediction quality that are learned with structure preservation alone.

There is a wide range of literature on learning models of physical systems from data. First, there are methods that learn models from state observations, such as dynamic mode decomposition [24, 38, 44, 51], sparsity-based methods [10, 42, 41, 50],

[*] Submitted to the editors December 3, 2022.

[†] Corresponding author. Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, 39106 Magdeburg, Germany (goyalp@mpi-magdeburg.mpg.de).

[‡] Courant Institute of Mathematical Sciences, New York University, USA

[§] Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, 39106 Magdeburg, Germany (benner@mpi-magdeburg.mpg.de).

and operator inference [33, 35, 6, 52, 26, 49, 55, 5]. Closer to the approach introduced in this work are methods from the systems and control community that learn models from input-output data, such as the Loewner, AAA, and vector fitting techniques in the frequency domain [1, 25, 2, 20, 4, 16, 15, 53, 30, 21], and time domain [23, 32]. Closest to our work is the approach introduced in [46], which fits structured models by interpolating the training data. However, imposing the structure can mean that there are more degrees of freedom than conditions to fix them. The authors of [46] introduce additional constraints to close all degrees of freedom, which means that the learned model is not parsimonious anymore by construction and so can require more degrees of freedom than necessary for explaining the training data. Several works aim to learn structured models by fitting model parameters via gradient-based learning methods [27, 48, 47, 28]. However, these methods require that the number of parameters and, thus, the order of the models is fixed a priori, and it cannot be adapted during the training. Additionally, the success of these methods critically depends on the initial guess of the model parameters, which can be challenging to obtain and typically relies on heuristic arguments.

Our approach adapts the order of the model during training to obtain parsimonious, structured models without redundant degrees of freedom. Building on model reduction [39, 8, 43, 2], in particular the works [3, 7], the key ingredient is describing models that interpolate the given input-output data as solutions of generalized Sylvester equations, which allows formulating the training problem as finding minimal-rank matrices that solve specific instances of Sylvester equations. We then relax the rank in the objective to the weighted nuclear norm so that the corresponding optimization problems can be solved efficiently with of-the-shelf techniques [11, 17, 37, 22]. It is important to note that structure such as symmetry is encoded in the learned model rather than merely being weakly enforced via penalization in the loss function. In fact, in our numerical experiments, imposing structure onto the model allows the optimizer to find even lower-order models that fit the data well than if the order of the model alone is minimized without structure preservation. In our numerical experiments, the proposed approach also outperforms standard techniques from black-box machine learning in terms of prediction quality by roughly a factor of two on the same training data sets.

The manuscript is structured as follows. In Section 2, we briefly discuss traditional model reduction techniques [39, 8], in particular the interpolatory techniques proposed in [3, 7] to construct low-order models of structured systems when high-dimensional models are available. In Section 3, we carry over the interpolatory model reduction techniques to the data-driven setting to formulate our approach that learns minimal-order, structured models from data. Extensions of our approach to symmetric systems, systems with parameters, and multiple inputs and multiple outputs are introduced in Section 4. Section 5 considers computational aspects of our approach and proposes a relaxation of the rank minimization via the weighted nuclear norm to make it computationally tractable. In Section 6, we illustrate the efficiency of the proposed methodology on various examples, and conclusions are drawn in Section 7.

2. Preliminaries.

We briefly review systems that are given by structured input-output maps, which are transfer functions in the case of dynamical systems.

2.1. Structured systems.

Input-output maps, which for dynamical systems are transfer functions, of a wide class of structured systems have the form

$$\mathbf{H}(\mathbf{s}) = \mathbf{C}\left(\sum\nolimits_{i=1}^{q} \alpha_i(\mathbf{s})\mathbf{A}_i\right)^{-1}\mathbf{B}, \tag{2.1}$$

where

$$\mathbf{A}_1, \dots, \mathbf{A}_q \in \mathbb{R}^{n\times n}, \quad \mathbf{B} \in \mathbb{R}^{n\times m}, \quad \mathbf{C} \in \mathbb{R}^{l\times n} \tag{2.2}$$

are system matrices, the functions $\alpha_i : \mathbb{C}^d \to \mathbb{C}$ are assumed to be smooth meromorphic functions, and $\mathbf{s} \in \mathbb{C}^d$ is a vector argument to these functions that can include parameters of the system. Transfer functions of structured systems such as delay systems, second-order systems, and affine parameter-dependent systems can be described in the form (2.1). For example, for a single-delay system, $\alpha_1(\mathbf{s}), \alpha_2(\mathbf{s})$, and $\alpha_3(\mathbf{s})$ can, respectively, be $1, \mathbf{s}$, and $e^{\tau\mathbf{s}}$, where τ is a delay. If the input-output map (2.1) represents a non-parametric transfer function, then $\mathbf{s}$ is scalar and takes values on the imaginary axis.

In the following, the transfer function $\mathbf{H}(\mathbf{s})$ is strictly proper, which means that

$$\lim_{\|\mathbf{s}\|\to\infty} \mathbf{H}(\mathbf{s}) = 0$$

holds. Furthermore, the functions $\alpha_i, i = 1, \dots, q$ are linear and independent, and thus, there cannot exist a non-zero vector $\boldsymbol{f} \in \mathbb{C}^q$ for which

$$[\alpha_1(\mathbf{s}), \dots, \alpha_q(\mathbf{s})]\, \boldsymbol{f} = 0, \qquad \forall \mathbf{s}.$$

2.2. Intrusive construction of low-order structured models. We briefly discuss the approach introduced in [7] for constructing low-order structured models. For ease of exposition, we focus on single-input single-output (SISO) systems so that $l = m = 1$; extensions to multiple-input multiple-output (MIMO) systems are discussed later in Subsection 4.1. Furthermore, the functions $\alpha_i : \mathbb{C} \to \mathbb{C}$ have a scalar argument $\mathbf{s} \in \mathbb{C}$. The extension to a multi-variate $\mathbf{s}$ and $\alpha_i : \mathbb{C}^d \to \mathbb{C}$ is just a technical extension and is discussed in Subsection 4.3.

The aim of the model reduction approach described in [7] is to determine a low-order model

$$\widehat{\mathbf{H}}(\mathbf{s}) = \widehat{\mathbf{C}} \left(\sum\nolimits_{i=1}^{q} \alpha_i(\mathbf{s}) \widehat{\mathbf{A}}_i\right)^{-1} \widehat{\mathbf{B}}, \tag{2.3}$$

where $\widehat{\mathbf{A}}_i \in \mathbb{R}^{r\times r}, i = 1, \dots, q$, $\widehat{\mathbf{B}} \in \mathbb{R}^{m\times r}$, and $\widehat{\mathbf{C}} \in \mathbb{R}^{l\times r}$. The order of the $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ matrices is r and the aim is to choose r much smaller than n, the order of the matrices of the high-dimensional model (2.1), while ensuring accuracy within a tolerance `tol` with respect to (2.1) via

$$\|\mathbf{H} - \widehat{\mathbf{H}}\| \leq \texttt{tol}\,, \tag{2.4}$$

in an appropriate norm $\|\cdot\|$. The functions α_i in (2.3) are the same as in (2.1). The matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q \in \mathbb{R}^{r\times r}$, $\widehat{\mathbf{B}} \in \mathbb{R}^{m\times r}$, $\widehat{\mathbf{C}} \in \mathbb{R}^{l\times r}$ are constructed via Petrov-Galerkin projection with the projection matrices $\mathbf{V}$ and $\mathbf{W}$,

$$\widehat{\mathbf{A}}_i = \mathbf{W}^\top \mathbf{A}_i \mathbf{V}, \quad \widehat{\mathbf{B}} = \mathbf{W}^\top \mathbf{B}, \quad \widehat{\mathbf{C}} = \mathbf{C}\mathbf{V}. \tag{2.5}$$

There are many ways to construct the projection matrices $\mathbf{V}$ and $\mathbf{W}$; see, e.g., [3, 7, 9]. It is important to note that constructing the matrices as in (2.5) via projection with $\mathbf{V}$ and $\mathbf{W}$ is intrusive because it requires having the matrices (2.2) of the high-dimensional model available either in an assembled form or via a routine that provides the vector-matrix product.

2.3. Intrusive interpolatory model reduction. One widely used approach to construct $\mathbf{V}$ and $\mathbf{W}$ is by enforcing interpolation conditions

$$\mathbf{H}(\sigma_i) = \widehat{\mathbf{H}}(\sigma_i), \qquad i = 1, \dots, r \tag{2.6}$$

at interpolation points $\sigma_1, \dots, \sigma_r \in \mathbb{C}$. As shown in [3], if $\sum_{i=1}^{q} \alpha_i(\mathbf{s})\mathbf{A}_i$ is invertible for all $\mathbf{s} \in \{\sigma_1, \dots, \sigma_r\}$, then the low-order model (2.3) with the matrices (2.5) obtained with the projection matrices

$$\mathbf{V} = \left[\left(\textstyle\sum_{i=1}^{q} \alpha_i(\sigma_1)\mathbf{A}_i\right)^{-1}\mathbf{B}, \dots, \left(\textstyle\sum_{i=1}^{q} \alpha_i(\sigma_r)\mathbf{A}_i\right)^{-1}\mathbf{B}\right], \tag{2.7a}$$

$$\mathbf{W} = \left[\left(\textstyle\sum_{i=1}^{q} \alpha_i(\sigma_1)\mathbf{A}_i\right)^{-T}\mathbf{C}, \dots, \left(\textstyle\sum_{i=1}^{q} \alpha_i(\sigma_r)\mathbf{A}_i\right)^{-T}\mathbf{C}\right], \tag{2.7b}$$

satisfies (2.6), provided $\sum_{i=1}^{q} \alpha_i(\mathbf{s})\widehat{\mathbf{A}}_i$ is also invertible for all $\mathbf{s} \in \{\sigma_1, \dots, \sigma_r\}$. Moreover, the derivative information at the interpolation points $\sigma_1, \dots, \sigma_r$ is interpolated as well,

$$\frac{\mathrm{d}}{\mathrm{d}\mathbf{s}}\mathbf{H}(\sigma_i) = \frac{\mathrm{d}}{\mathrm{d}\mathbf{s}}\widehat{\mathbf{H}}(\sigma_i), \quad i = 1, \dots, r.$$

When constructing the matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ defined in (2.5) of the low-order model (2.3) with the $\mathbf{V}$ and $\mathbf{W}$ matrices defined in (2.7), then the order r of the model, i.e., the size of the matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$, equals the number of interpolation conditions (2.6). Thus, the number of given interpolation conditions in the form of the training data determines the order of the model (2.3) that is obtained with this process. However, there can be another model of order $r_{\min} < r$ that also interpolates the data (2.6). In the following, we say that the order $r_{\min}$ is minimal if no matrices of a lower order than $r_{\min}$ exist that lead to a model (2.3) that satisfies the interpolation conditions (2.6). In practice, to reduce the order of a model while maintaining good approximate quality, once the matrices $\widehat{\mathbf{A}}_1, \dots \widehat{\mathbf{A}}_q$ are determined with $\mathbf{V}$ and $\mathbf{W}$, they are often compressed via the singular value decomposition (SVD). The compression typically leads to models that slightly violate the interpolation conditions (2.6) in favor of having lower order models that approximate the high-dimensional model in the sense of (2.4).

The procedure to interpolatory model reduction outlined in this section is intrusive because the matrices of the high-dimensional model (2.2) are needed if the projection matrices $\mathbf{V}$ and $\mathbf{W}$ and the model matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ are assembled as in (2.7) and (2.5), respectively.

3. Rank-minimizing and structured model inference (RSMI). We now introduce rank-minimizing and structured model inference (RSMI) to find a structured low-order system model with the following input-output map

$$\widetilde{\mathbf{H}}(\mathbf{s}) = \widetilde{\mathbf{C}}\left(\sum_{i=1}^{q} \alpha_i(\mathbf{s})\widetilde{\mathbf{A}}_i\right)^{-1}\widetilde{\mathbf{B}} \tag{3.1}$$

from input-output data of a system (2.1)

$$\{(\sigma_1, \mathbf{H}(\sigma_1)), \dots, (\sigma_{\mathcal{N}}, \mathbf{H}(\sigma_{\mathcal{N}}))\} \tag{3.2}$$

and the functions $\alpha_1, \dots, \alpha_q$, but without access to the system matrices of the unknown system (2.2).

We build on a formulation of the projection matrices $\mathbf{V}$ and $\mathbf{W}$ defined in (2.7) as solutions to generalized Sylvester equations, which we use to derive Sylvester equations for the matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ that depend on the training data (3.2) only. This then motivates the objective function of the proposed RSMI approach for inferring matrices $\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q$ of minimal rank that satisfy the generalized Sylvester equations and so give rise to a model that fits the training data well while ensuring a low-order representation.

3.1. Model matrices as solutions of generalized Sylvester equations.

The projection matrices $\mathbf{V}$ and $\mathbf{W}$ in (2.7a) and (2.7b), respectively, solve the following generalized Sylvester equations:

$$\mathbf{A}_1\mathbf{V}\Lambda_1 + \cdots + \mathbf{A}_q\mathbf{V}\Lambda_q = \mathbf{B}\mathbb{1}^\top, \tag{3.3a}$$

$$\mathbf{A}_1^\top\mathbf{W}\Lambda_1 + \cdots + \mathbf{A}_q^\top\mathbf{W}\Lambda_q = \mathbf{C}^\top\mathbb{1}^\top, \tag{3.3b}$$

where $\Lambda_i = \operatorname{diag}(\alpha_i(\sigma_1), \dots, \alpha_i(\sigma_\mathcal{N}))$ and $\mathbb{1}$ is a column vector of ones. Multiplying (3.3a) and (3.3b) by the matrices $\mathbf{W}^\top$ and $\mathbf{V}^\top$, respectively, from the left leads to

$$\mathbf{W}^\top\mathbf{A}_1\mathbf{V}\Lambda_1 + \cdots + \mathbf{W}^\top\mathbf{A}_q\mathbf{V}\Lambda_q = \mathbf{W}^\top\mathbf{B}\mathbb{1}^\top, \tag{3.4a}$$

$$\mathbf{V}^\top\mathbf{A}_1^\top\mathbf{W}\Lambda_1 + \cdots + \mathbf{V}^\top\mathbf{A}_q^\top\mathbf{W}\Lambda_q = \mathbf{V}^\top\mathbf{C}^\top\mathbb{1}^\top. \tag{3.4b}$$

Now, notice that $\mathbf{W}^\top\mathbf{A}_i\mathbf{V} = \widehat{\mathbf{A}}_i$ for $i = 1, \dots, q$ and

$$\mathbf{W}^\top\mathbf{B} = \widehat{\mathbf{B}} = \Big[\mathbf{H}(\sigma_1), \dots, \mathbf{H}(\sigma_\mathcal{N})\Big]^\top, \qquad \mathbf{C}\mathbf{V} = \widehat{\mathbf{C}} = \Big[\mathbf{H}(\sigma_1), \dots, \mathbf{H}(\sigma_\mathcal{N})\Big].$$

Consequently, we can write (3.4) using only matrices that define a model:

$$\widehat{\mathbf{A}}_1\Lambda_1 + \cdots + \widehat{\mathbf{A}}_q\Lambda_q = \mathbf{H}_\sigma\mathbb{1}^\top, \tag{3.5a}$$

$$\widehat{\mathbf{A}}_1^\top\Lambda_1 + \cdots + \widehat{\mathbf{A}}_q^\top\Lambda_q = \mathbf{H}_\sigma\mathbb{1}^\top, \tag{3.5b}$$

where

$$\mathbf{H}_\sigma = \Big[\mathbf{H}(\sigma_1), \dots, \mathbf{H}(\sigma_\mathcal{N})\Big]^\top.$$

It is important to note that (3.5) is independent of the system matrices of the unknown model (2.1) from which data (3.2) are sampled. Thus, the solutions of the equations (3.5) provide us directly with matrices (2.5) of models that interpolate the data (3.2), without access to the high-dimensional matrices (2.2). This is in contrast to finding the matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ via projection as in (2.5), which is based on the projection matrices $\mathbf{V}$ and $\mathbf{W}$ defined in (2.7) that are assembled with matrix-vector products with the system matrices of the unknown model. However, directly solving equations (3.5) for the matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ leads to matrices that form a model of order that scales with the number of data points $\mathcal{N}$, which potentially is much higher than the minimal order required to interpolate the data; see also the comments in Section 2.3.

3.2. Uniqueness of solution of generalized Sylvester equations.

All matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ that solve (3.5) lead to models with map $\widehat{\mathbf{H}}$ that interpolate the training data (3.2). If $q = 2$, it can be shown that the matrices $\widehat{\mathbf{A}}_1$ and $\widehat{\mathbf{A}}_2$ satisfy the following Sylvester equations

$$\Lambda_2^\top\widehat{\mathbf{A}}_1\Lambda_1 - \Lambda_1^\top\widehat{\mathbf{A}}_1\Lambda_2 = \Lambda_2^\top\mathbf{H}_\sigma\mathbb{1}^\top - \mathbb{1}\mathbf{H}_\sigma^\top\Lambda_2^\top, \tag{3.6a}$$

$$\Lambda_2^\top\widehat{\mathbf{A}}_2\Lambda_1 - \Lambda_1^\top\widehat{\mathbf{A}}_2\Lambda_2 = \Lambda_1^\top\mathbf{H}_\sigma\mathbb{1}^\top - \mathbb{1}\mathbf{H}_\sigma^\top\Lambda_1^\top, \tag{3.6b}$$

which have a unique solution if derivative information of $\mathbf{H}(\mathbf{s})$ at σ_i is available, see [25, 46]. Note that the model described by the solution matrices of (3.6) can be represented differently such as via basis transformations and potentially truncation. It further is shown in [46] that the solution of (3.6) can be given analytically by modifying the Loewner and shifted Loewner matrices [25]. For Rayleigh-damped second-order systems, the analytical solution of (3.6) is discussed in [34], and for a class of delay systems in [45].

In the more general case, when $q \geq 2$, there are more degrees of freedom than the number of equations, and thus, there can be arbitrarily many solutions; see the discussion about the work [46] in the introduction in Section 1.

3.3. Rank minimization and structure preserving models. To motivate our RSMI formulation, we first state a result from [7] about how the rank of matrices with interpolating conditions relates to the order of models: Consider the training data (3.2) and notice that the existence of $\mathbf{H}(\sigma_1), \dots, \mathbf{H}(\sigma_\mathcal{N})$ in the second component of the tuples in (3.2) implies that

$$\sum_{i=1}^{q} \alpha_i(\mathbf{s})\mathbf{A}_i$$

is invertible for all $s \in \{\sigma_1, \dots, \sigma_\mathcal{N}\}$. Let now $\mathbf{V}$ and $\mathbf{W}$ be the projection matrices defined in (2.7), define the matrices

$$\widehat{\mathcal{A}} = \left[\mathbf{W}^\top \mathbf{A}_1 \mathbf{V}, \dots, \mathbf{W}^\top \mathbf{A}_q \mathbf{V}\right] = \left[\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q\right], \tag{3.7a}$$

$$\widehat{\mathcal{A}}_T = \begin{bmatrix} \mathbf{W}^\top \mathbf{A}_1 \mathbf{V} \\ \vdots \\ \mathbf{W}^\top \mathbf{A}_q \mathbf{V} \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{A}}_1 \\ \vdots \\ \widehat{\mathbf{A}}_q \end{bmatrix}, \tag{3.7b}$$

and set

$$r_{\min} = \min\left\{\operatorname{rank}(\widehat{\mathcal{A}}), \operatorname{rank}(\widehat{\mathcal{A}}_T)\right\}. \tag{3.8}$$

It is shown in [7] that there exists a model of order $r_{\min}$ that interpolates the training data (3.2) and that the order $r_{\min}$ is minimal. Moreover, an $r_{\min}$-order model can be constructed from the matrices $\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q$ satisfying (3.5) via projection onto appropriate $r_{\min}$-dimensional subspaces; see [7] for more details.

The insights from the previous paragraph motivate us to look for solutions of the Sylvester equations (3.5) that have minimal rank, which leads to the objective

$$\mathcal{J}(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q) = \min\left\{\operatorname{rank}\left([\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q]\right), \operatorname{rank}\left([\widetilde{\mathbf{A}}_1^\top, \dots, \widetilde{\mathbf{A}}_q^\top]^\top\right)\right\}, \tag{3.9}$$

and the optimization problem

$$\begin{aligned} &\min_{\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q} && \mathcal{J}(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q) \\ &\text{subject to} && \widetilde{\mathbf{A}}_1 \Lambda_1 + \dots + \widetilde{\mathbf{A}}_q \Lambda_q = \mathbf{H}_\sigma \mathbb{1}^\top, \\ &&& \widetilde{\mathbf{A}}_1^\top \Lambda_1 + \dots + \widetilde{\mathbf{A}}_q^\top \Lambda_q = \mathbf{H}_\sigma \mathbb{1}^\top, \end{aligned} \tag{RSMI}$$

where $\Lambda_i, i = 1, \dots, q$ and $\mathbf{H}_\sigma$ are defined in (3.3) and (3.5), respectively. Thus, in (RSMI), we are seeking matrices $\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q$ such that they give rise to a low-order model that interpolates the training data because we know that the objective (3.9) of (RSMI) minimizes the rank of (3.8), which also determines the order of the minimal model. Consequently, once we identified $\widetilde{\mathbf{A}}_1, \ldots, \widetilde{\mathbf{A}}_q$ with (RSMI), we can use them to construct low-order models following [7].

3.4. An illustrative example. Consider a scalar delay example as follows:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= -\mathbf{x}(t) + 0.25\mathbf{x}(t-1) + \mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{x}(t). \end{aligned} \tag{3.10}$$

The transfer function of the above system is $\mathbf{H}(\mathbf{s}) = (s + 1 - 0.25e^{-s})^{-1}$. Let us assume that we have two measurements $\mathbf{H}(\sigma_1)$ and $\mathbf{H}(\sigma_2)$ at σ_1 and σ_2. Using these two data points, we would like to learn a model of the form:

$$\begin{aligned} \mathbf{E}\mathbf{z}(t) &= \mathbf{A}\mathbf{z}(t) + \mathbf{A}_\tau \mathbf{z}(t-1) + \mathbf{B}\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{z}(t), \end{aligned} \tag{3.11}$$

ensuring

$$\mathbf{H}_z(\sigma_i) = \mathbf{H}(\sigma_i), i \in \{1, 2\}, \tag{3.12}$$

where $\mathbf{H}_z(\mathbf{s}) = \mathbf{C}\left(s\mathbf{E} - \mathbf{A} - e^{-s}\mathbf{A}_\tau\right)^{-1}\mathbf{B}$. We now know that if the matrices $\mathbf{E}, \mathbf{A}, \mathbf{A}_\tau$ satisfy

$$\mathbf{E}\begin{bmatrix}\sigma_1 & \\ & \sigma_2\end{bmatrix} + \mathbf{A}\begin{bmatrix}1 & \\ & 1\end{bmatrix} + \mathbf{A}_\tau\begin{bmatrix}e^{-\sigma_1} & \\ & e^{-\sigma_2}\end{bmatrix} = \begin{bmatrix}\mathbf{H}(\sigma_1) \\ \mathbf{H}(\sigma_2)\end{bmatrix}\mathbb{1}^\top, \tag{3.13a}$$

$$\mathbf{E}^\top\begin{bmatrix}\sigma_1 & \\ & \sigma_2\end{bmatrix} + \mathbf{A}^\top\begin{bmatrix}1 & \\ & 1\end{bmatrix} + \mathbf{A}_\tau^\top\begin{bmatrix}e^{-\sigma_1} & \\ & e^{-\sigma_2}\end{bmatrix} = \begin{bmatrix}\mathbf{H}(\sigma_1) \\ \mathbf{H}(\sigma_2)\end{bmatrix}\mathbb{1}^\top, \tag{3.13b}$$

and $\mathbf{B} = \mathbf{C}^\top = [\mathbf{H}(\sigma_1)^\top, \mathbf{H}(\sigma_2)^\top]^\top$, then the model given by the matrices $\mathbf{E}, \mathbf{A}, \mathbf{A}_\tau, \mathbf{B}, \mathbf{C}$ interpolates the data. However, there clearly are arbitrarily many solutions of (3.13) for $\mathbf{E}, \mathbf{A}, \mathbf{A}_\tau$.

We now discuss two particular solutions. First, we set $\mathbf{A}_\tau = 0$ and solve (3.13) for $\mathbf{E}$ and $\mathbf{A}$. It means that we approximate the time-delay system with a rational function. In this case, when we observe the rank of the matrix $[\mathbf{E}, \mathbf{A}, \mathbf{A}_\tau]$, it is 2; hence, we cannot recover the original system. But it interpolates the data. In contrast, if we determine a model using the rank-based optimization problem, i.e.,

$$\min_{\mathbf{E},\mathbf{A},\mathbf{A}_\tau} \operatorname{rank}\left(\left[\mathbf{E}, \mathbf{A}, \mathbf{A}_\tau\right]\right) \quad \text{subject to (3.13)}, \tag{3.14}$$

then we can get the solution

$$\begin{aligned} \mathbf{E} &= \begin{bmatrix}\mathbf{H}(\sigma_1) & \\ & \mathbf{H}(\sigma_2)\end{bmatrix}\begin{bmatrix}1 & 1 \\ 1 & 1\end{bmatrix}\begin{bmatrix}\mathbf{H}(\sigma_1) & \\ & \mathbf{H}(\sigma_2)\end{bmatrix}, \\ \mathbf{A} &= -\begin{bmatrix}\mathbf{H}(\sigma_1) & \\ & \mathbf{H}(\sigma_2)\end{bmatrix}\begin{bmatrix}1 & 1 \\ 1 & 1\end{bmatrix}\begin{bmatrix}\mathbf{H}(\sigma_1) & \\ & \mathbf{H}(\sigma_2)\end{bmatrix}, \\ \mathbf{A}_\tau &= 0.25\begin{bmatrix}\mathbf{H}(\sigma_1) & \\ & \mathbf{H}(\sigma_2)\end{bmatrix}\begin{bmatrix}1 & 1 \\ 1 & 1\end{bmatrix}\begin{bmatrix}\mathbf{H}(\sigma_1) & \\ & \mathbf{H}(\sigma_2)\end{bmatrix}. \end{aligned} \tag{3.15}$$

It can be easily seen that $\operatorname{rank}([\mathbf{E}, \mathbf{A}, \mathbf{A}_\tau]) = 1$. After a compression step, as will be shown later in Algorithm 1, we can recover a model of the original system.

Algorithm 1 Obtaining low-order models via projection onto dominant subspaces.

```
Input: Samples $\mathbf{H}(\sigma_i), i \in \{1, \ldots, \mathcal{N}\}$, and functions $\alpha_i(\cdot)$.
1: Formulate and solve the rank minimization problem as in (RSMI) and denote its
   solution by $\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_q$.
2: Compute SVDs of the following matrices:
   $\mathbf{U}_1\Sigma_1\mathbf{V}_1^\top = \left[\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_q\right]$, and $\mathbf{U}_2\Sigma_2\mathbf{V}_2^\top = \begin{bmatrix} \bar{\mathbf{A}}_1 \\ \vdots \\ \bar{\mathbf{A}}_q \end{bmatrix}$.
3: Determine the order $r$ such that
   $\max\left\{ \frac{\sum_{i=1}^r \sigma_1^{(i)}}{\sum_i \sigma_1^{(i)}}, \frac{\sum_{i=1}^r \sigma_2^{(i)}}{\sum_i \sigma_2^{(i)}} \right\} \leq \texttt{tol}$,
   where $\sigma_1^{(i)}$ and $\sigma_2^{(i)}$ are the $i$-th diagonal entries of $\Sigma_1$ and $\Sigma_2$.
4: Determine projection matrices: $\mathbf{W} = \mathbf{U}_1^{(r)}$ and $\mathbf{V} = \mathbf{V}_2^{(r)}$, where $\mathbf{U}_1^{(r)}$ and $\mathbf{V}_2^{(r)}$
   denote the matrices that contain as columns the first $r$ dominant vectors of $\mathbf{U}_1$
   and $\mathbf{V}_2$, respectively.
5: Construct $\mathbf{H}_\sigma = [\mathbf{H}(\sigma_1), \ldots, \mathbf{H}(\sigma_\mathcal{N})]$.
6: Construct the low-order model of the form (2.1) with matrices
   $\widetilde{\mathbf{A}}_i = \mathbf{W}^\top\bar{\mathbf{A}}_i\mathbf{V}, i = 1, \ldots, q, \quad \widetilde{\mathbf{B}} = \mathbf{W}^\top\mathbf{H}_\sigma^\top, \quad \widetilde{\mathbf{C}} = \mathbf{H}_\sigma\mathbf{V}$.
return $\widetilde{\mathbf{A}}_1, \ldots, \widetilde{\mathbf{A}}_q, \widetilde{\mathbf{B}}, \widetilde{\mathbf{C}}$.
```

3.5. Algorithmic description of RSMI.

We now present an algorithm to construct lower-order models by projections on dominant left and right subspace once the optimization problem (3.9) is solved. The algorithm is inspired from [7, Algo. 1]. The algorithm is sketched in Algorithm 1. In the first step, we solve the optimization problem (RSMI) to obtain $\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_q$. Step 2 computes singular values decompositions of appropriately constructed matrices using $\bar{\mathbf{A}}_1, \ldots, \bar{\mathbf{A}}_q$. It allows us to determine a suitable order of a model that approximates well the given data in Step 3. It is followed by determining dominant subspaces $\mathbf{V}$ and $\mathbf{W}$ in Step 4 that allows us to construct the desired low-order model with matrices $\widetilde{\mathbf{A}}_1, \ldots, \widetilde{\mathbf{A}}_q, \widetilde{\mathbf{B}}, \widetilde{\mathbf{C}}$ in Step 6.

4. Extensions of RSMI.

In this section, we discuss several extensions to RSMI.

4.1. Systems with multiple inputs and multiple outputs.

To extend RSMI to MIMO systems, we build on the concept of tangential interpolation [19]. To that end, for given data set (3.2) with $\mathbf{H}(\sigma_i) \in \mathbb{C}^{l\times m}, i = 1, \ldots, \mathcal{N}$ now being $l \times m$ matrices, we seek to identify a structured model of the form (3.1) such that

$$\begin{aligned} \mathbf{H}(\sigma_i)\mathbf{b}_i &= \widehat{\mathbf{H}}(\sigma_i)\mathbf{b}_i\,, \qquad i = 1, \ldots, \mathcal{N}\,, \\ \mathbf{c}_i^\top\mathbf{H}(\sigma_i) &= \mathbf{c}_i^\top\widehat{\mathbf{H}}(\sigma_i)\,, \qquad i = 1, \ldots, \mathcal{N}, \end{aligned} \tag{4.1}$$

where $\mathbf{b}_i \in \mathbb{C}^m$ and $\mathbf{c}_i \in \mathbb{C}^l$ are right and left tangential directions, respectively, for $i = 1, \ldots, \mathcal{N}$. The interpolation conditions (4.1) lead to generalized Sylvester equations with different right-hand sides than (3.5),

$$\widehat{\mathbf{A}}_1\Lambda_1 + \cdots + \widehat{\mathbf{A}}_q\Lambda_q = \begin{bmatrix} \mathbf{c}_1^\top \mathbf{H}_{\sigma_1} \\ \vdots \\ \mathbf{c}_{\mathcal{N}}^\top \mathbf{H}_{\sigma_{\mathcal{N}}} \end{bmatrix} \mathcal{B}, \tag{4.2a}$$

$$\widehat{\mathbf{A}}_1^\top\Lambda_1 + \cdots + \widehat{\mathbf{A}}_q^\top\Lambda_q = \begin{bmatrix} \mathbf{b}_1^\top \mathbf{H}_{\sigma_1}^\top \\ \vdots \\ \mathbf{b}_{\mathcal{N}}^\top \mathbf{H}_{\sigma_{\mathcal{N}}}^\top \end{bmatrix} \mathcal{C}, \tag{4.2b}$$

where $\mathcal{B} = [\mathbf{b}_1, \dots, \mathbf{b}_{\mathcal{N}}]$ and $\mathcal{C} = [\mathbf{c}_1, \dots, \mathbf{c}_{\mathcal{N}}]$. Equations (4.2) are then used as constraints when minimizing the objective $\mathcal{J}$ defined in (3.9). The rest of the procedure of RSMI, for example, solving optimization problems and constructing lower-order models using dominant subspaces, remains the same as for the SISO case.

4.2. Symmetric systems. The input-output map is symmetric if, for the data, it holds that $\mathbf{H}(\mathbf{s}) = \mathbf{H}^\top(\mathbf{s})$ for all $\mathbf{s}$. We then typically want to preserve that symmetry by constraining the matrices in the RSMI optimization problem (RSMI) to be symmetric. We derive the approach for the SISO case for ease of exposition, but it readily generalizes to the MIMO case following Section 4.1.

First, note that the constraint $\widetilde{\mathbf{A}}_i = \widetilde{\mathbf{A}}_i^\top$ for $i = 1, \dots, q$ means that the two constraints in (RSMI) coincide. Furthermore, the equality

$$\operatorname{rank}\left(\left[\widehat{\mathbf{A}}_1, \dots, \widehat{\mathbf{A}}_q\right]\right) = \operatorname{rank}\left(\begin{bmatrix} \widehat{\mathbf{A}}_1 \\ \vdots \\ \widehat{\mathbf{A}}_q \end{bmatrix}\right) \tag{4.3}$$

holds. To avoid having to add the symmetry constraints explicitly as $\widetilde{\mathbf{A}}_i = \widetilde{\mathbf{A}}_i^\top$, we parametrize $\widetilde{\mathbf{A}}_i$ as $\widetilde{\mathbf{A}}_i = \widetilde{\mathbf{K}}_i + \widetilde{\mathbf{K}}_i^\top$ for $i = 1, \dots r$. We then obtain the objective

$$\mathcal{J}_s(\widetilde{\mathbf{K}}_1, \dots, \widetilde{\mathbf{K}}_q) = \operatorname{rank}\left([\widetilde{\mathbf{K}}_1 + \widetilde{\mathbf{K}}_1^\top, \dots, \widetilde{\mathbf{K}}_q + \widetilde{\mathbf{K}}_q^\top]\right) .$$

Hence, the optimization problem becomes

$$\begin{aligned} \min_{\widetilde{\mathbf{K}}_1,\dots,\widetilde{\mathbf{K}}_q} \quad & \mathcal{J}_s(\widetilde{\mathbf{K}}_1, \dots, \widetilde{\mathbf{K}}_q) \\ \text{subject to} \quad & (\widetilde{\mathbf{K}}_1 + \widetilde{\mathbf{K}}_1^\top)\Lambda_1 + \cdots + (\widetilde{\mathbf{K}}_q + \widetilde{\mathbf{K}}_q^\top)\Lambda_q = \mathbf{H}_\sigma \mathbb{1}^\top , \end{aligned} \tag{sRSMI}$$

with $\Lambda_1, \dots, \Lambda_q$ and $\mathbf{H}_\sigma$ defined as in the RSMI problem (RSMI). From the matrix $\widetilde{\mathbf{K}}_i$ the matrix $\widetilde{\mathbf{A}}_i = \widetilde{\mathbf{K}}_i + \widetilde{\mathbf{K}}_i^\top$ can be readily constructed for $i = 1, \dots, q$. Furthermore, Step 2 in Algorithm 1 simplifies because $\mathbf{U}_1\Sigma_1\mathbf{V}_1^\top = \mathbf{V}_2\Sigma_2\mathbf{U}_2^\top$, which means that $\mathbf{W} = \mathbf{V}$ in Step 4.

4.3. Systems with parameters. We developed RSMI for function $\alpha_i : \mathbb{C} \to \mathbb{C}, i = 1, \dots, q$ in (2.1) with scalar input $\mathbf{s}$. Although the discussions so far readily cover parametric cases, it is worthwhile to explicitly highlight that we can apply RSMI for stationary parametric problems, which are of the form:

$$\begin{aligned} (\alpha_1(\mathbf{p})\mathbf{A}_1 + \cdots + \alpha_q(\mathbf{p})\mathbf{A}_q))\, \mathbf{X}(\mathbf{p}) &= \mathbf{B}, \\ \mathbf{y}(\mathbf{p}) &= \mathbf{C}\mathbf{X}(\mathbf{p}). \end{aligned} \tag{4.4}$$

Hence, we can obtain structured models for parametric systems, which interpolate the outputs at given parameter values and are of low order. In our numerical section, we will illustrate this scenario using a thermal block example.

5. Relaxation of rank-minimization problem.

The objective (RSMI) depends on the rank of a matrix. Directly solving rank-minimization problems is challenging because they lead to non-convex NP-hard optimization problems. In this section, we build on relaxations of rank-minimization problems based on the nuclear norm [17, 18, 37], which is the sum of singular values. This results in optimization problems that can be solved efficiently with gradient-based methods. However, even though the nuclear norm leads to the best convex relaxation of rank-minimization objectives, the solutions to the relaxed problems are not close enough to the original problem. To address this issue, the concept of the weighted nuclear norm was proposed in [22], which, in practice, often yields a solution closer to the original rank-minimization problem.

5.1. Weighted nuclear-norm formulation of RSMI.

The weighted nuclear norm of a matrix $\mathbf{T} \in \mathbb{R}^{n\times m}$ with $m \geq n$ and weight weight $\mathbf{w} \in \mathbb{R}^n$ is

$$\|\mathbf{T}\|_{\mathbf{w},*} = \sum_{i=1}^{n} w_i\sigma_i,$$

where σ_i are the singular values of the matrix $\mathbf{T}$ with $\sigma_{i+1} \geq \sigma_i$, and w_i is the i^{th} component of the vector $\mathbf{w}$ [22, 29]. We consider a vector $\mathbf{w}$ with $w_i \geq w_{i-1}, i = 2, \dots, n$. Defining the vector $\mathbf{w}$ with non-decreasing weights gives larger weights to smaller singular values. As a result, the smaller singular values in the course of the optimization tend to be even smaller due to a higher weight and so are nudged closer to the solution of the rank-minimization problem. The weighted nuclear norm with non-decreasing vector $\mathbf{w}$ is also a convex function [22]. Note that $\mathbf{w} = [1, \dots, 1]^\top$ is a feasible non-decreasing weight vector, which coincides with the classical nuclear norm.

Based on the weighted nuclear norm, we formulate the objective $\mathcal{J}_r$ as

$$\text{(rRSMI)} \quad \begin{aligned} \mathcal{J}_r(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q, \lambda) = \lambda\|[\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q]\|_{\mathbf{w},*} + \lambda\|[\widetilde{\mathbf{A}}_1^\top, \dots, \widetilde{\mathbf{A}}_q^\top]^\top\|_{\mathbf{w},*} + \\ \|\mathcal{R}_1(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q)\| + \|\mathcal{R}_2(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q)\| \end{aligned}$$

with the terms

$$\begin{aligned} \mathcal{R}_1(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q) &= \widetilde{\mathbf{A}}_1\Lambda_1 + \cdots + \widetilde{\mathbf{A}}_q\Lambda_q - \mathbf{H}_\sigma\mathbb{1}^\top, \\ \mathcal{R}_2(\widetilde{\mathbf{A}}_1, \dots, \widetilde{\mathbf{A}}_q) &= \widetilde{\mathbf{A}}_1^\top\Lambda_1 + \cdots + \widetilde{\mathbf{A}}_q^\top\Lambda_q - \mathbf{H}_\sigma\mathbb{1}^\top, \end{aligned} \tag{5.1}$$

that are obtained from the constraints in (RSMI) with the Lagrange multiplier λ.

The weight vectors influence the optimization process. Typically, a good choice for the weight vector can be based on the singular values of the solution, and the weights can be updated iteratively during the optimization. In our setting of identifying input-output maps of systems, the singular vectors corresponding to the most significant singular values are important for capturing the system dynamics. In contrast, the singular vectors corresponding to small singular values carry little information about the system dynamics and, thus, typically can be safely ignored. We, therefore, define the weights inversely proportional to the singular values in the following. Such a scheme has been initially discussed in [56], which is inspired by the re-weighting scheme proposed in [12] for weighted l_1 problems to obtain sparse solutions.

Algorithm 2 An iterative scheme to solve the relaxed RSMI problem.

```
Input: Samples $\mathbf{H}(\sigma_i), i \in \{1, \dots, \mathcal{N}\}$, functions $\alpha_i(\cdot)$, number of iterations (iters), regularizing parameters $\lambda^{(0)}, \lambda^{(1)}, \dots, \lambda^{(\text{iters})}$.
Solve the optimization problem (rRSMI) with regularizing parameter $\lambda^{(0)}$ and the constant weight vector $\mathbf{w} = [1, \dots, 1]$. Denote the solution as $\widetilde{\mathbf{A}}_1^{(0)}, \dots, \widetilde{\mathbf{A}}_q^{(0)}$.
for $i = 1, \dots,$ iters do
    Compute singular values $s_1, \dots, s_{\mathcal{N}}$ of $[\widetilde{\mathbf{A}}_1^{(i-1)}, \dots, \widetilde{\mathbf{A}}_q^{(i-1)}]$.
    Define the $j$-th component of the weight vector $\mathbf{w}$ as
        $$\mathbf{w}_j = \min\{\texttt{max\_val}, 1./(s_j + \epsilon)\}.$$
    The threshold $\epsilon$ is a small number to avoid division by zero and max_val prevents too high values in the weight vector.
    Solve (rRSMI) with regularizing parameter $\lambda^{(i)}$ and the weight vector. $\mathbf{w}$. Denote the solution as $\widetilde{\mathbf{A}}_1^{(i)}, \dots, \widetilde{\mathbf{A}}_q^{(i)}$
end for
return $\widetilde{\mathbf{A}}_1^{(\texttt{iters})}, \dots, \widetilde{\mathbf{A}}_q^{(\texttt{iters})}$.
```

Example	$\lambda^{(0)}$	`lr`
Delay heat rod model	5×10^{-3}	5×10^{-3}
Fishtail robot	5×10^{-3}	1×10^{-2}
Parametric thermal block	5×10^{-3}	5×10^{-3}

Table 1: Parameters used in the numerical experiments.

5.2. Algorithm to solve relaxed RSMI problem.

In Algorithm 2, we show a computational procedure for solving the relaxed optimization problem of RSMI based on the weighted-nuclear norm subject to linear constraints. In the first step, we solve (rRSMI) with a constant weight vector $\mathbf{w} = [1, \dots, 1]$ because we do not have a good choice for the weight vector at the beginning. In Steps 3 and 4, at iteration i, we update the weight vector $\mathbf{w}$ based on the current solution $\widetilde{\mathbf{A}}_0^{(i)}, \dots, \widetilde{\mathbf{A}}_q^{(i)}$. For this, we compute the singular values of the matrix $[\widetilde{\mathbf{A}}_0^{(i)}, \dots, \widetilde{\mathbf{A}}_q^{(i)}]$ and define the weight vector to have as components the inverse of the singular values plus a safety threshold for avoiding division by zero. With the updated weighting vector, we solve (rRSMI) again and continue the iterations. We also can vary the regularizing parameter with each iteration. Typically, we use regularization parameters that are non-decreasing with iterations; see numerical results in Section 6. Once we have a solution, then we can obtain a lower-order model using Steps 2-6 of Algorithm 1.

Remark 5.1. It is often desirable to obtain models with matrices with real entries only, but interpolation points σ_i and training data $\{\sigma_i, \mathbf{H}(\sigma_i)\}$ can be complex. If the data comes from a real model and training data are closed under conjugation, one can transform the inferred model into a real model as described in, e.g., [46].

6. Numerical Experiments.

In this section, we illustrate RSMI on three numerical examples.

6.1. Implementation details. We make use of the PyTorch library [31] to compute gradients with automatic differentiation and optimize with `NAdam` [14]. To compute $\|\mathcal{R}_i\|$ in (rRSMI), we use a combination of l_1 and l_2-norms with an equal weight as in

$$\|\mathcal{R}_i\| = \|\texttt{vect}(\mathcal{R}_i)\|_{l_1} + \|\texttt{vect}(\mathcal{R}_i)\|_{l_2}^2, \tag{6.1}$$

where $\texttt{vect}(\cdot)$ is the vectorization of a matrix.

We compare models obtained with three different approaches:

Benchmark approach. We construct a structured model with no regularization by setting $\lambda^{(0)} = 0$ in (rRSMI) and `iters` to 0 in Algorithm 2. We take $50,000$ optimization iterations with an initial learning rate `lr` which is reduced by the factor of 5 after every $12,500$-th update. This approach serves as our benchmark, and it is closely related to the approach discussed in [46] for the identification of structured systems. We denote this approach by `benchmark` in the following.

RSMI with nuclear norm. We learn structured models with (rRSMI) with the constant-weight nuclear norm and with regularization parameter as given in Table 1. The rest of the setup is the same as in the previous graph. We denote this approach by `rRSMI(eq_wts)`.

RSMI with weighted nuclear norm. In this approach, we employ a weighted nuclear norm in (rRSMI). We first construct a model in Step 1 in Algorithm 2 with $\lambda^{(0)} = \lambda_{\text{reg}}$ and learning rate given in Table 1. Then, in Step 6 of Algorithm 2 and the i-th iteration, we use $\lambda^{(i)} = \lambda^{(0)}/i$. A motivation to gradually decrease the regularization parameter is that we slowly give more weight to satisfying the interpolation conditions. But note that smaller singular values tend to stay small due to their high weight, which is inversely proportional to the singular value. Together with decreasing the regularization parameter, in Step 6, we also reduce the initial learning rate by a factor of two. Furthermore, `max_val` in Algorithm 2 is set to 10^4. We denote this approach by `rRSMI`.

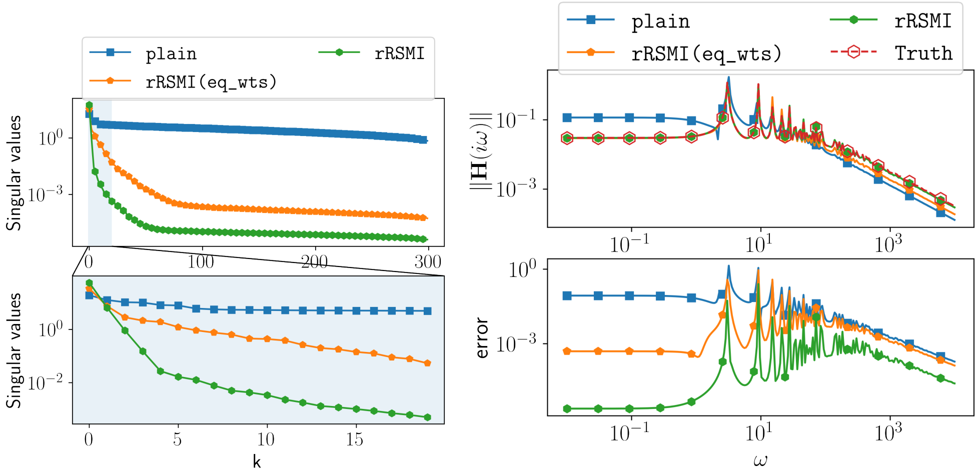
6.2. Delay heat rod model. Consider the heating road example with delay described in [46]. We construct a state-space model of order $n = 101$ by setting the same parameters as in [46]. The system has delay $\tau = 1$. The transfer function of the system is of the form:

$$\mathbf{H}(s) = \mathbf{C}\left(s\mathbf{E} - \mathbf{A} - \mathbf{A}_\tau e^{-\tau s}\right)^{-1}\mathbf{B}. \tag{6.2}$$

For training data, we consider $\mathcal{N} = 150$ points $\sigma_1, \dots, \sigma_{\mathcal{N}}$ on the imaginary axis in the frequency range $[10^{-1}, 10^3]$ and the corresponding outputs $\mathbf{H}(\sigma_1), \dots, \mathbf{H}(\sigma_{\mathcal{N}})$. Using these training data, we aim to learn a dynamical-system model with a transfer function that interpolates the training data. We construct interpolating models using the three approaches `benchmark`, `rRSMI(eq_wts)`, and `rRSMI`.

6.2.1. Results. In Figure 1a, we plot the singular values of the matrices of the models obtained by the three approaches. Note that all models aim to interpolate the training data and encode the same structure. The decay of the singular values corresponding to the models obtained from the three approaches indicates how much the models can be compressed, determining if there exist lower-order models that can also interpolate well.

The singular values corresponding to the model obtained with `benchmark` decay slowly, which implies that the model cannot be compressed well. In contrast, the



(a) Decay of singular values.

(b) Transfer function comparison.

Fig. 1: Delay heat rod problem: Plot (a) shows the decay of the singular values corresponding to the models obtained with `benchmark`, `rRSMI(eq_wts)`, and `rRSMI`. The singular values corresponding to the model obtained with `rRSMI` decays fastest, which means that it can be compressed most. Plot (b) shows the error of the learned models after compressing them to order $r = 3$, where the `rRSMI` model obtains the lowest error.

model learned with our approach `rRSMI(eq_wts)` leads to a faster decay of the singular values, and thus the learned model can be compressed while approximating the training data well. The decay of the singular values is even faster for the model learned with `rRSMI` that relies on the weighted nuclear norm for regularization. This implies that the model obtained with `rRSMI` is the most compressible, resulting in the lowest order.

We now construct low-order models by truncating the original models obtained with `benchmark`, `rRSMI(eq_wts)`, and `rRSMI`. We truncate after the first three most significant singular values. As a result, we obtain models of order three that have the same delay structure as (6.2). We determine the quality of these models on test data, which are different from the training data. We consider 250 frequency points in the interval $[10^{-2}, 10^4]$ on a logarithm scale, which is also outside of the domain of the training data. We compare the obtained truncated models with the ground truth in Figure 1b. The plots show that the model obtained with our approach `rRSMI` provides a low-dimensional model with the lowest error on the training and test data. Figure 2 shows the median error over all test data points, where the model obtained with the proposed `rRSMI` approach achieves orders of magnitude lower errors than the model obtained with `benchmark` approach that has no rank-minimization constraints.

6.2.2. Imposing symmetry onto matrices. We repeat the analog experiments to Subsection 6.2.1 but impose symmetry as described in Subsection 4.2. We first examine the decay of singular values shown in Figure 3a, which again indicates a similar trend as in the non-symmetric case. The first singular vector corresponding

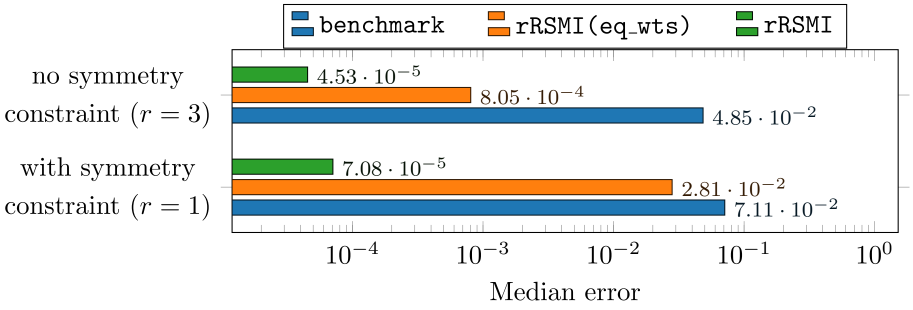
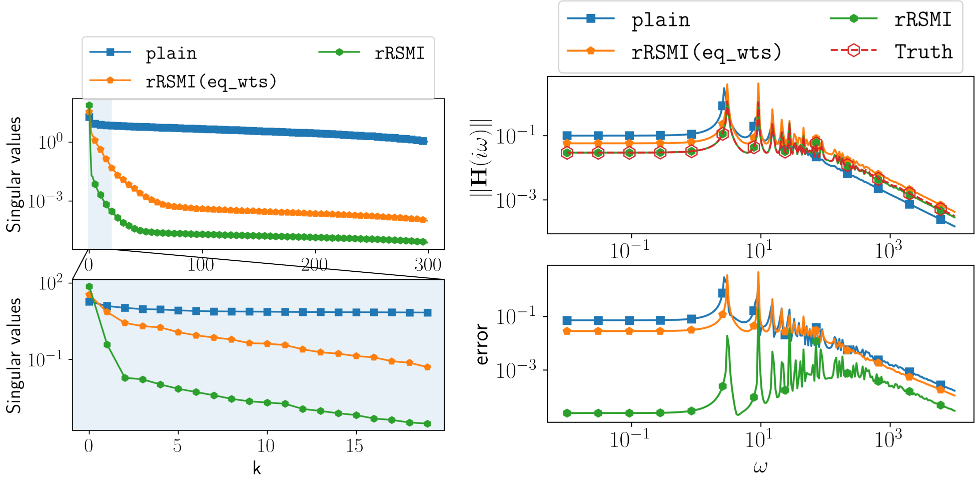


Fig. 2: Delay heat rod model: The plot shows that models learned with our approach `rRSMI` achieve orders of magnitude lower median errors over the test data points.



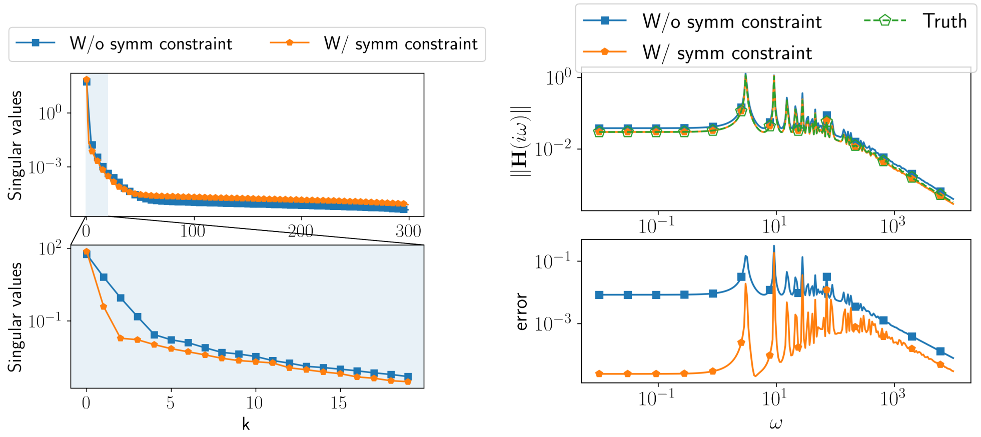
(a) Decay of singular values.

(b) Transfer function comparison.

Fig. 3: Delay heat rod model: Plot (a) shows that imposing symmetry leads to a faster decay of the singular values corresponding to the model learned with our approach `rRSMI`, compared to not imposing symmetry as shown in Figure 1. The faster decay means that the model can be compressed to just one dimension $r = 1$ and still provides accurate predictions on test data, as shown in plot (b).

to the model obtained with `rRSMI` captures more than 99% energy. Thus, we construct one-dimensional models by truncating the models obtained with `benchmark`, `rRSMI(eq_wts)`, and `rRSMI` after the first mode and compare their performance on the test data in Figure 3b. Our approach `rRSMI` yields the model with the lowest median error, which is below 10^{-4} and shown in Figure 2.

We explicitly compare the performance of the models that impose symmetry versus the models obtained without symmetry constraints. We do this only for the models obtained with the `rRSMI` approach. What we observe is a faster decay of singular values of the models obtained with `rRSMI` when the symmetry in matrices is imposed,



(a) Decay of singular values.

(b) Transfer function comparison.

Fig. 4: Delay heat rod model: The plots show the faster decay of the singular values corresponding to models learned with `rRSMI` with symmetry constraints. Compression to just one dimension leads to a model with two orders of magnitude lower error on test data if symmetries are enforced.

see Figure 4a. Therefore, we obtain lower dimensional models with `rRSMI` in this example with symmetry constraints. We now compare the model obtained with truncating after the first mode with symmetry imposed to the truncated model without symmetry; see Figure 4b. We observe that the symmetry constraint leads to almost two orders of magnitude lower error in this example.

6.3. Fishtail robot. In our second example, we consider an artificial fishtail robotic model from [40] as shown in Figure 5a. The state-space model from which we generate data has a second-order structure of the form:

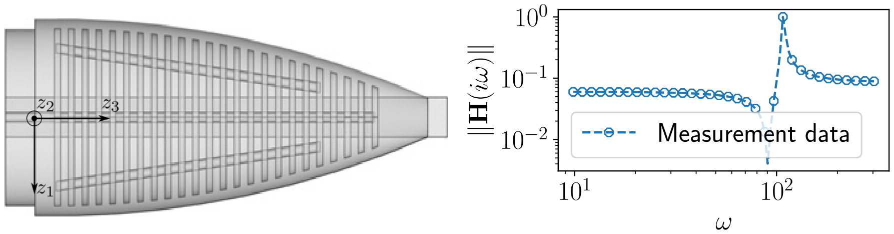
$$\begin{aligned}\mathbf{M}\ddot{\mathbf{x}}(t)+\mathbf{D}\dot{\mathbf{x}}(t)+\mathbf{K}\mathbf{x}(t)&=\mathbf{B}\mathbf{u}(t),\\ \mathbf{y}(t)&=\mathbf{C}\mathbf{x}(t),\end{aligned}\tag{6.3}$$

and its state has dimension $\approx 700,000$. In our setting, we have only available 100 measurements of the transfer function at the frequencies in the interval $[10^1, 3\times 10^2]$ as shown in Figure 5b. We have normalized the transfer function by dividing by the maximum absolute value; otherwise $\|\mathbf{H}(s)\|$ is in the range of 10^{-7} and thus becomes numerically too small for the considered frequency range.

6.3.1. Results. We leverage that transfer functions of second-order systems have the form

$$\mathbf{H}_{\mathsf{so}}=\mathbf{C}\left(s^2\mathbf{M}+s\mathbf{D}+\mathbf{K}\right)^{-1}\mathbf{B}.\tag{6.4}$$

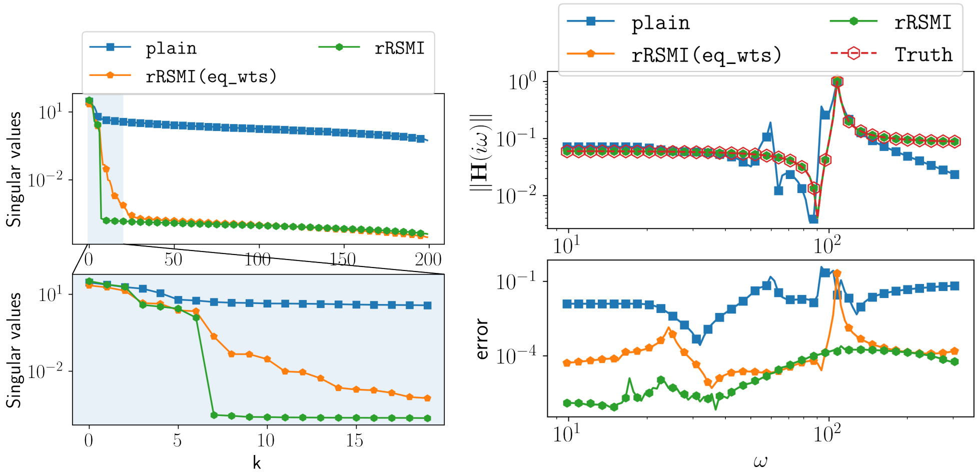
We employ the `benchmark`, `rRSMI(eq_wts)`, and `rRSMI` approaches to construct interpolating models. Before doing so, we scale the function s^2 and s by factor $\gamma_{\mathbf{M}}$ and $\gamma_{\mathbf{D}}$. The reason for this is that while assembling the matrices for s^2 when the frequency interval is $[10^1, 3\times 10^2]$, the corresponding matrix becomes dominating, especially



(a) A semantic figure of the fishtail robot from [40].

(b) Training data.

Fig. 5: Fishtail robot: The plots show a sketch of the robot and the training data.



(a) The decay of singular values.

(b) Transfer function comparison.

Fig. 6: Fishtail Robot: Plot (a) shows that the singular values corresponding to the model learned with our approach `rRSMI` decay fastest, which indicates that the model can be compressed well. Plot (b) shows the prediction error on data of the learned models when compressed to order $r = 7$, where our approach leads to the lowest error.

for high frequencies. As a result, the optimization problems become computationally harder to solve. In such scenarios, scaling can help the optimizer. However, note that the scaling does not affect the transfer function, as it is invariant under scaling,

$$\mathbf{H}_{\mathrm{so}} = \mathbf{C}\left(s^2\mathbf{M} + s\mathbf{D} + \mathbf{K}\right)^{-1}\mathbf{B} = \mathbf{C}\left(\gamma_{\mathbf{M}} s^2\widetilde{\mathbf{M}} + \gamma_{\mathbf{D}} s\widetilde{\mathbf{D}} + \widetilde{\mathbf{K}}\right)^{-1}\mathbf{B}, \tag{6.5}$$

where $\widetilde{\mathbf{M}} = \mathbf{M}/\gamma_{\mathbf{M}}$ and $\widetilde{\mathbf{D}} = \mathbf{D}/\gamma_{\mathbf{D}}$ and $\widetilde{\mathbf{K}} = \mathbf{K}$. Therefore, we seek to learn $\widetilde{\mathbf{M}}, \widetilde{\mathbf{D}}$ and $\widetilde{\mathbf{K}}$ with the scaled $\alpha_1(s) = \gamma_{\mathbf{M}} s^2$ and $\alpha_2(s) = \gamma_{\mathbf{D}} s$. For this example, we choose $\gamma_{\mathbf{M}} = 10^{-3}$ and $\gamma_{\mathbf{D}} = 10^{-1.5}$. We identify the interpolating realizations using all three approaches, as in the previous example.

We plot the singular values corresponding to the learned models in Figure 6a.

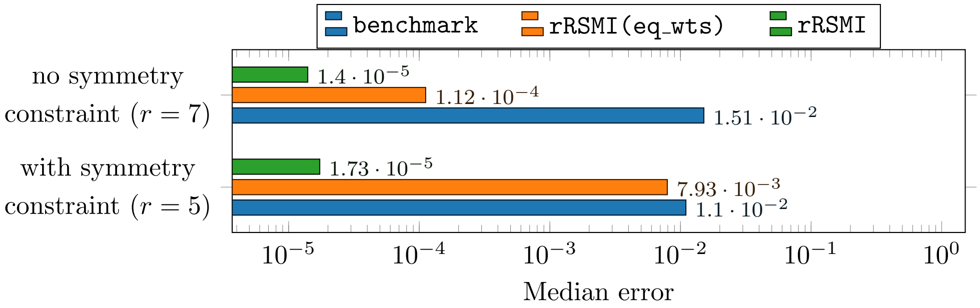


Fig. 7: Fishtail robot: The median error on the data is orders of magnitude lower for models learned with our approach `rRSMI` than without rank-minimization as in the `benchmark` approach.

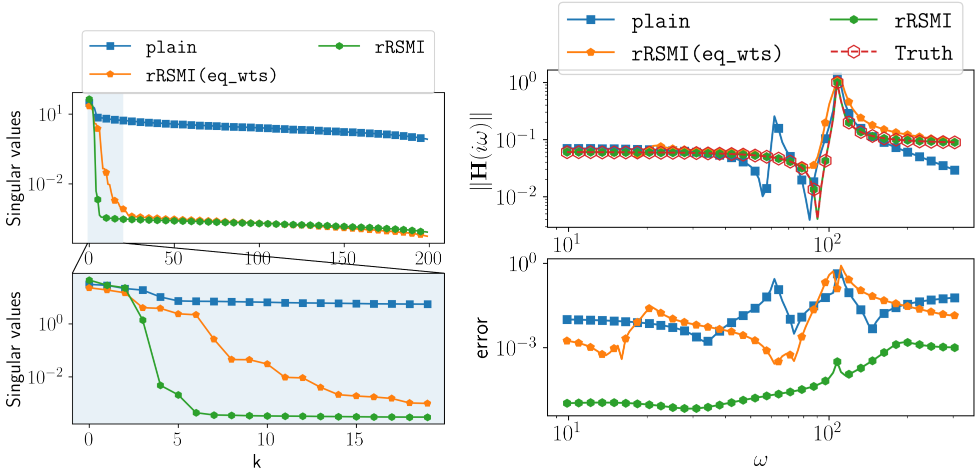
The singular values exhibit similar behavior as in the previous example, in the sense that the singular values corresponding to the model learned with our `rRSMI` approach decay fastest. This indicates that the model learned with `rRSMI` can be compressed efficiently to a lower-order model without significant loss of accuracy. We truncate after the first $r = 7$ dominating singular values and so obtain models of order seven. We compare the truncated models on the training data in Figures 6b and 7. Again, `rRSMI` yields the model with the lowest median error among all models, showing the robustness of the `rRSMI` approach to learn lower-order models that approximately interpolate the data.

6.3.2. Imposing symmetry. Having symmetry in the matrices $\mathbf{M}, \mathbf{D}$, and $\mathbf{K}$ is of high interest in mechanical systems. We, therefore, enforce symmetry in these matrices while learning them with the `benchmark`, `rRSMI(eq_wts)`, and `rRSMI` approaches. The rest of the setup is the same as in the previous experiment. The singular values corresponding to the models learned with symmetry constraints are shown in Figure 8a. Again, the model obtained with `rRSMI` shows the fastest decay of the singular values. Moreover, with symmetry imposed, the decay is even faster than in the case without symmetry. Consequently, we truncate earlier and obtain a model of lower order than when symmetry is not imposed. We construct models of the order $r = 5$ using all three methods `benchmark`, `rRSMI(eq_wts)`, `rRSMI` by projection using the corresponding dominant subspaces spanned by the left singular vectors and compare the error of the truncated models in Figures 7 and 8b. Our proposed approach `rRSMI` leads to the model with the lowest median error by more than one order of magnitude.

6.4. Thermal block example. We now consider a 2×2 thermal block, whose solution is described by the Poisson equation with the following diffusion term:

(6.6)
$$\eta(\zeta, p) = p_1 \mathcal{X}_{\left(0,\frac{1}{2}\right)^2}(\zeta) + p_2 \mathcal{X}_{\left(0,\frac{1}{2}\right)\times\left(\frac{1}{2},1\right)}(\zeta) + p_3 \mathcal{X}_{\left(\frac{1}{2},1\right)^2}(\zeta) + p_4 \mathcal{X}_{\left(0,\frac{1}{2}\right)\times\left(\frac{1}{2},1\right)}(\zeta),$$

where $p_i \in [0.1, 10], i \in \{1, \ldots, 4\}$ and $\mathcal{X}_X$ is the indicator function of the set X. This is a widely used benchmark example [36] and has also been studied in [27]. Having



(a) Decay of singular values.

(b) Transfer function comparison.

Fig. 8: Fishtail robot: Plot (a) shows that imposing symmetry helps to achieve a faster decay of the singular values corresponding to the learned models, which means the models can be compressed while remaining predictive. Plot (b) shows after compression to order $r = 5$, the model obtained with our approach `rRSMI` achieves the lowest error in this experiment.

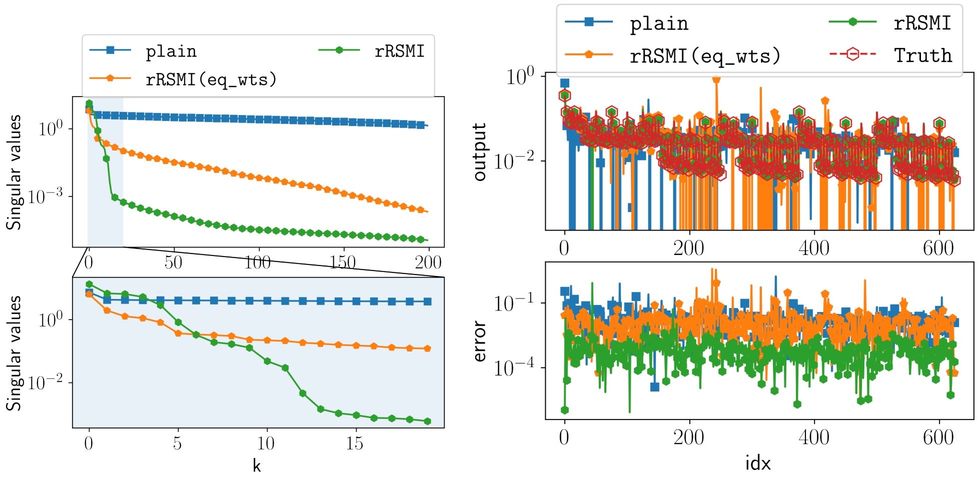
discretized the Poisson equation, we get a model of the form:

$$\begin{aligned} (\mathbf{A}_0 + p_1\mathbf{A}_1 + p_2\mathbf{A}_2 + p_3\mathbf{A}_3 + p_4\mathbf{A}_4)\,\mathbf{x}(p) &= \mathbf{B}, \\ \mathbf{y}(p) &= \mathbf{C}\mathbf{x}(p). \end{aligned} \tag{6.7}$$

For more details on the governing equation, we refer to [27, 36]. We collect the data by considering four equidistant points in $[0.1, 10]$ for each $p_i, i = 1, \dots, 4$. Consequently, we get a total of $4^4 = 256$ training data points. For the construction of the models, we uniformly draw 200 data points from the training data and determine a suitable order of lower-order models by validating the learned models on the 56 left-out data points. We additionally construct another test data set for this example by considering five equidistant points in $[0.1, 10]$ for each $p_i, i = 1, \dots 4$, which leads to a total of $5^4 = 625$ test points.

6.4.1. Results. We employ the `benchmark`, `rRSMI(eq_wts)`, and `rRSMI` approaches to construct models using the training data. The singular values are shown in Figure 9a, which exhibit similar decay behavior as in the previous example: The model obtained with `rRSMI` achieves a faster decay than the models obtained with either `benchmark` or `rRSMI(eq_wts)`. We compress the model to order $r = 10$ and show the error of the models in Figure 9b. The median errors on the test data for realizations obtained using `benchmark`, `rRSMI(eq_wts)`, and `rRSMI` are reported in Figure 10. The `rRSMI` approach again yields a model that achieves an order of magnitude lower median error than the other approaches `rRSMI(eq_wts)` and `benchmark`.

6.4.2. Imposing symmetry. We now learn models with symmetric matrices $\mathbf{A}_i, i \in \{0, \dots, 4\}$. The rest of the setup is the same as in the experiment without sym-



(a) The decay of singular values.

(b) Output comparison on the testing data.

Fig. 9: Thermal block example: Plot (a) shows the decay of the singular values, which is fastest for the model learned with `rRSMI`. Plot (b) shows the error on test data after compression to order $r = 10$, where the model obtained with `rRSMI` achieves the lowest error.

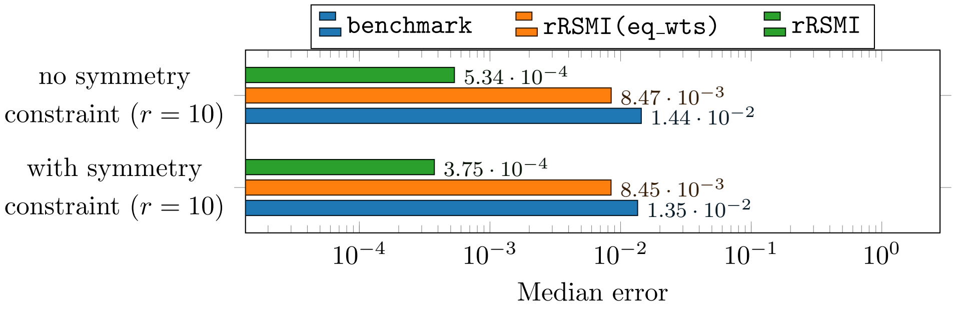
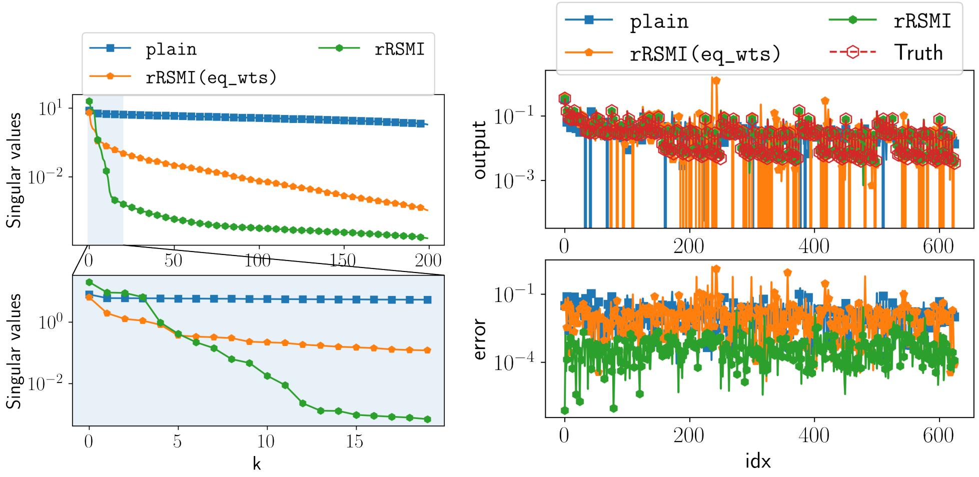


Fig. 10: Thermal block example: The proposed approach `rRSMI` leads to models that achieve almost two orders of magnitude lower median errors on test data compared to models obtained with rank minimization as in `benchmark`.

metry constraints. Analogous to previous experiments, imposing symmetry leads to a quicker decay of the singular values, and the model obtained with `rRSMI` has again the fastest decay. In contrast, the models obtained with `benchmark` and `rRSMI(eq_wts)` are hardly compressible and prone to over-fitting to the training data. We construct models of order $r = 10$ via projection onto the subspace spanned by the first r left-singular vectors and report the error of the models on the test data in Figures 10 and 11. The model obtained with our approach `rRSMI` achieves the lowest median error by more than one order of magnitude.



(a) Decay of singular values.

(b) Output comparison.

Fig. 11: Thermal block example: Imposing symmetry helps to obtain a faster decay of the singular values with `rRSMI`, which leads to a lower error in predictions on test data.

6.4.3. Comparison with deep neural networks. We compare our `rRSMI` approach to a machine-learning method that fits a deep neural network to the input-output data. The machine-learning approach is a black box because it ignores the structure. The network architecture is fully connected. To train the network, we use the same training, validation, and testing data sets as for our `rRSMI` approach. To choose the hyper-parameters, in particular the number of hidden nodes and layers, we perform a grid search in $[4, 8, 16, 32] \times [1, 2, 3]$, where $[4, 8, 16, 32]$ and $[1, 2, 3]$ are possible numbers of nodes and layers, respectively. We then select the hyper-parameters that lead to the lowest error on the validation data set. In our case, we use one hidden layer and eight nodes. Figure 12 shows the error of the trained network on the test data and compares it to the error obtained with the model of $r = 10$ with our `rRSMI` approach with symmetry imposed. Our approach leads to a model that has a lower median error by a factor of two.

7. Conclusions. The proposed RSMI approach has two key properties that make it stand out: First, it leverages physical insights that are often given in the form of structure, such as symmetries, time delays, and degrees of time derivatives. The proposed RSMI approach bakes in such physical insights by imposing the corresponding structure onto the model. Critically, the structure is encoded in the model so that it cannot be violated rather than being weakly enforced via penalization terms. Additionally, the learned structured input-output models can be realized as systems that describe dynamics of states in the time domain and how they behave under control inputs, which is in contrast to black-box models that only match the input-output behavior. The second key property of RSMI is that the order of the inferred model, i.e., the number of degrees of freedom, is a training variable that is minimized during the optimization. The order of the model, therefore, is determined during training

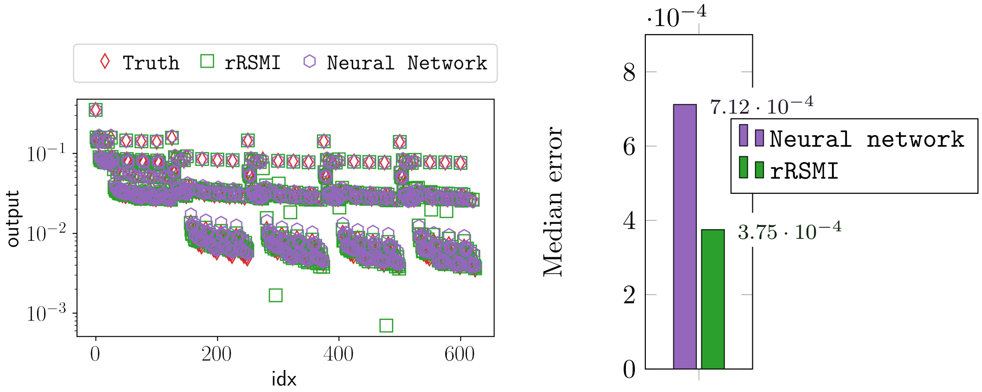


Fig. 12: Thermal block example: Plot (a) shows the outputs predicted by our approach `rRSMI` and by a neural network and compares them to the truth. Plot (b) compares the median error on the test data, where our approach achieves a factor two improvement over the network in this example.

and does not have to be fixed based on heuristics a priori, as is typical in many other data-driven modeling approaches. By minimizing the order, RSMI learns models that are parsimonious because redundant degrees of freedom are eliminated when there are not sufficiently many training samples available to fix them. The numerical experiments demonstrate that the combination of structure preservation and optimizing the model order leads to models with low degrees of freedom while achieving high prediction capabilities. More broadly speaking, the results indicate that optimizing over the model architecture during the training can lead to more accurate predictions while keeping the number of degrees of freedom of the learned models low. There is a range of future research directions. For example, we have used (weighted-) nuclear norm minimization schemes as a proxies of rank-minimization problems. In the future, we would like to explore other relaxations and computationally efficient approaches to obtain a approximate solutioons to the rank-minimization problems.

Acknowledgements. The second author was partially supported by the National Science Foundation under Grant No. 2012250.

REFERENCES

[1] A. C. Antoulas and B. D. O. Anderson, *On the scalar rational interpolation problem*, IMA J. Math. Control and Inform., 3 (1986), pp. 61–88.

[2] A. C. Antoulas, C. A. Beattie, and S. Gugercin, *Interpolatory Methods for Model Reduction*, Computational Science & Engineering, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2021.

[3] C. A. Beattie and S. Gugercin, *Interpolatory projection methods for structure-preserving model reduction*, Systems Control Lett., 58 (2009), pp. 225–232.

[4] P. Benner and P. Goyal, *Interpolation-based model order reduction for polynomial systems*, SIAM J. Sci. Comput., 43 (2021), pp. A84–A108.

[5] P. Benner, P. Goyal, J. Heiland, and I. Pontes Duff, *Operator inference and physics-informed learning of low-dimensional models for incompressible flows*, Electron. Trans. Numer. Anal., 56 (2022), pp. 28–51. Special Issue on Scientific Machine Learning.

[6] P. Benner, P. Goyal, B. Kramer, B. Peherstorfer, and K. Willcox, *Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms*, Comp.

Meth. Appl. Mech. Eng., 372 (2020), p. 113433.

[7] P. Benner, P. Goyal, and I. Pontes Duff, *Identification of dominant subspaces for linear structured parametric systems and model reduction*, (2019).

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

[9] T. Breiten, *Structure-preserving model reduction for integro-differential equations*, SIAM J. Cont. Optim., 54 (2016), pp. 2992–3015.

[10] S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, Proc. Nat. Acad. Sci. U.S.A., 113 (2016), pp. 3932–3937.

[11] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Foundations Comput. Math., 9 (2009), p. 717.

[12] E. J. Candes, M. B. Wakin, and S. P. Boyd, *Enhancing sparsity by reweighted l_1 minimization*, J. Fourier Analysis Appl., 14 (2008), pp. 877–905.

[13] P. V. Coveney, E. R. Dougherty, and R. R. Highfield, *Big data need big theory too*, Philosophical Trans. Royal Society A: Mathematical, Physical and Engineering Sciences, 374 (2016), p. 20160153.

[14] T. Dozat, *Incorporating nesterov momentum into adam*, in Proc. 4th International Conference on Learning Representations, Workshop Track, 2016.

[15] Z. Drmač, S. Gugercin, and C. Beattie, *Quadrature-based vector fitting for discretized $\mathcal{H}_2$ approximation*, SIAM J. Sci. Comput., 37 (2015), pp. A625–A652.

[16] Z. Drmač and B. Peherstorfer, *Learning low-dimensional dynamical-system models from noisy frequency-response data with Loewner rational interpolation*, in Realization and Model Reduction of Dynamical Systems: A Festschrift in Honor of the 70th Birthday of Thanos Antoulas, C. Beattie, P. Benner, M. Embree, S. Gugercin, and S. Lefteriu, eds., Cham, 2022, Springer International Publishing, pp. 39–57.

[17] M. Fazel, *Matrix rank minimization with applications*, PhD thesis, PhD thesis, Stanford University, 2002.

[18] M. Fazel, H. Hindi, S. P. Boyd, et al., *A rank minimization heuristic with application to minimum order system approximation*, in Proc. American Control Conference, vol. 6, Citeseer, 2001, pp. 4734–4739.

[19] K. Gallivan, A. Vandendorpe, and P. Van Dooren, *Model reduction of MIMO systems via tangential interpolation*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 328–349.

[20] I. V. Gosea and A. C. Antoulas, *Data-driven model order reduction of quadratic-bilinear systems*, Numer. Lin. Alg. Appl., 25 (2018), p. e2200.

[21] I. V. Gosea, S. Gugercin, and C. Beattie, *Data-driven balancing of linear dynamical systems*, SIAM Journal on Scientific Computing, 44 (2022), pp. A554–A582.

[22] S. Gu, L. Zhang, W. Zuo, and X. Feng, *Weighted nuclear norm minimization with application to image denoising*, in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2014, pp. 2862–2869.

[23] D. S. Karachalios, I. V. Gosea, and A. C. Antoulas, *On bilinear time-domain identification and reduction in the Loewner framework*, in Model Reduction of Complex Dynamical Systems, vol. 171 of International Series of Numerical Mathematics, Birkhäuser, Cham, 2021, pp. 3–30.

[24] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*, Society of Industrial and Applied Mathematics, Philadelphia, USA, 2016.

[25] A. J. Mayo and A. C. Antoulas, *A framework for the solution of the generalized realization problem*, Linear Algebra Appl., 425 (2007), pp. 634–662.

[26] S. A. McQuarrie, C. Huang, and K. Willcox, *Data-driven reduced-order models via regularised operator inference for a single-injector combustion process*, J. Royal Society of New Zealand, 51 (2021), pp. 194–211.

[27] P. Mlinarić and S. Gugercin, *$\mathcal{L}_2$-optimal reduced-order modeling using parameter-separable forms*, e-print 2206.02929, arXiv, 2022.

[28] P. Mlinarić and S. Gugercin, *A unifying framework for interpolatory $\mathcal{L}_2$-optimal reduced-order modeling*, (2022).

[29] K. Mohan and M. Fazel, *Iterative reweighted algorithms for matrix rank minimization*, J. Machine Learning Research, 13 (2012), pp. 3441–3473.

[30] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, *The AAA algorithm for rational approximation*, SIAM J. Sci. Comput., 40 (2018), pp. A1494–A1522.

[31] A. Paszke et al., *Pytorch: An imperative style, high-performance deep learning library*, in Proc. Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle,

A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., vol. 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[32] B. Peherstorfer, S. Gugercin, and K. Willcox, *Data-driven reduced model construction with time-domain Loewner models*, SIAM J. Sci. Comput., 39 (2017), pp. A2152–A2178.

[33] B. Peherstorfer and K. Willcox, *Data-driven operator inference for nonintrusive projection-based model reduction*, Computer Methods in Applied Mechanics and Engineering, 306 (2016), pp. 196–215.

[34] I. Pontes Duff, P. Benner, and P. Goyal, *Data-driven identification of Rayleigh-damped second-order systems*, arXiv:1910.00838, 2019.

[35] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, *Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems*, Physica D: Nonlinear Phenomena, 406 (2020), p. 132401.

[36] S. Rave and J. Saak, *A non-stationary thermal-block benchmark model for parametric model order reduction*, in Model Reduction of Complex Dynamical Systems, P. Benner, T. Breiten, H. Faßbender, M. Hinze, T. Stykel, and R. Zimmermann, eds., vol. 171 of International Series of Numerical Mathematics, Birkhäuser, Cham, 2021, pp. 349–356.

[37] B. Recht, M. Fazel, and P. A. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM Rev., 52 (2010), pp. 471–501.

[38] C. W. Rowley, I. Mezic, S. Bagheri, P. Schlatter, and D. S. Henningson, *Spectral analysis of nonlinear flows*, Journal of Fluid Mechanics, 641 (2009), pp. 115–1127.

[39] G. Rozza, D. B. P. Huynh, and A. 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 (2008), pp. 229–275.

[40] J. Saak, D. Siebelts, and S. W. R. Werner, *A comparison of second-order model order reduction methods for an artificial fishtail*, at-Automatisierungstechnik, 67 (2019), pp. 648–667.

[41] H. Schaeffer, *Learning partial differential equations via data discovery and sparse optimization*, Proc. Royal Society A: Mathematical, Physical and Engineering Sciences, 473 (2017), p. 20160446.

[42] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher, *Sparse dynamics for partial differential equations*, Proc. Nat. Acad. Sci. U.S.A., 110 (2013), pp. 6634–6639.

[43] W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, *Model Order Reduction: Theory, Research Aspects and Applications*, Springer-Verlag, Berlin, Heidelberg, 2008.

[44] P. J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, J. Fluid Mech., 656 (2010), pp. 5–28.

[45] P. Schulze and B. Unger, *Data-driven interpolation of dynamical systems with delay*, Systems Control Lett., 97 (2016), pp. 125–131.

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

[47] P. Schwerdtner and M. Schaller, *Structured optimization-based model order reduction for parametric systems*, (2022).

[48] P. Schwerdtner and M. Voigt, *Structure preserving model order reduction by parameter optimization*, arXiv preprint arXiv:2011.07567, (2020).

[49] H. Sharma, Z. Wang, and B. Kramer, *Hamiltonian operator inference: Physics-preserving learning of reduced-order models for Hamiltonian systems*, Physica D: Nonlinear Phenomena, 431 (2022), p. 133122.

[50] G. Tran and R. Ward, *Exact recovery of chaotic systems from highly corrupted data*, Multiscale Modeling & Simulation, 15 (2017), pp. 1108–1129.

[51] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, *On dynamic mode decomposition: Theory and applications*, J. Comp. Dyn., 1 (2014), pp. 391–421.

[52] W. I. T. Uy and B. Peherstorfer, *Probabilistic error estimation for non-intrusive reduced models learned from data of systems governed by linear parabolic partial differential equations*, ESAIM: M2AN, 55 (2021), pp. 735–761.

[53] S. W. R. Werner, I. V. Gosea, and S. Gugercin, *Structured vector fitting framework for mechanical systems*, e-print 2110.09220, arXiv, 2021. Numerical Analysis (math.NA).

[54] K. E. Willcox, O. Ghattas, and P. Heimbach, *The imperative of physics-based modeling and inverse theory in computational science*, Nature Comput. Sci., 1 (2021), pp. 166–168.

[55] S. Yildiz, P. Goyal, P. Benner, and B. Karasözen, *Learning reduced-order dynamics for a parametrized shallow water equation*, Internat. J. Numerical Methods in Fluids, 93 (2021), pp. 2803–2821.

[56] X. Zhong, L. Xu, Y. Li, Z. Liu, and E. Chen, *A nonconvex relaxation approach for rank minimization problems*, in Proc. AAAI Conf. Artificial Intelligence, vol. 29, 2015.