Approximation of Parametric Dynamical Systems

Andrea Carracedo Rodriguez

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Mathematics

Serkan Gugercin, Chair
Jeffrey T. Borggaard
Christopher A. Beattie
Traian Illiescu
Mark P. Embree

August 11, 2020
Blacksburg, Virginia

Keywords: Rational Approximation, Bilinear Model Reduction, Barycentric, Interpolation

Approximation of Parametric Dynamical Systems

Andrea Carracedo Rodriguez

(ABSTRACT)

Dynamical systems are widely used to model physical phenomena and, in many cases, these physical phenomena are parameter dependent. In this thesis we investigate three prominent problems related to the simulation of parametric dynamical systems and develop the analysis and computational framework to solve each of them.

In many cases we have access to data resulting from simulations of a parametric dynamical system for which an explicit description may not be available. We introduce the parametric AAA (p-AAA) algorithm that builds a rational approximation of the underlying parametric dynamical system from its input/output measurements, in the form of transfer function evaluations. Our algorithm generalizes the AAA algorithm, a popular method for the rational approximation of nonparametric systems, to the parametric case. We develop p-AAA for both scalar and matrix-valued data and study the impact of parameter scaling. Even though we present p-AAA with parametric dynamical systems in mind, the ideas can be applied to parametric stationary problems as well, and we include such examples.

The solution of a dynamical system can often be expressed in terms of an eigenvalue problem (EVP). In many cases, the resulting EVP is nonlinear and depends on a parameter. A common approach to solving (nonparametric) nonlinear EVPs is to approximate them with a rational EVP and then to linearize this approximation. An existing algorithm can then be applied to find the eigenvalues of this linearization. The AAA algorithm has been successfully applied to this scheme for the nonparametric case. We generalize this approach by using our

p-AAA algorithm to find a rational approximation of parametric nonlinear EVPs. We define a corresponding linearization that fits the format of the compact rational Krylov (CORK) algorithm for the approximation of eigenvalues.

The simulation of dynamical systems may be costly, since the need for accuracy may yield a system of very large dimension. This cost is magnified in the case of parametric dynamical systems, since one may be interested in simulations for many parameter values. Interpolatory model order reduction (MOR) tackles this problem by creating a surrogate model that interpolates the original, is of much smaller dimension, and captures the dynamics of the quantities of interest well. We generalize interpolatory projection MOR methods from parametric linear to parametric bilinear systems. We provide necessary subspace conditions to guarantee interpolation of the subsystems and their first and second derivatives, including the parameter gradients and Hessians.

Throughout the dissertation, the analysis is illustrated via various benchmark numerical examples.

Approximation of Parametric Dynamical Systems

Andrea Carracedo Rodriguez

(GENERAL AUDIENCE ABSTRACT)

Simulation of mathematical models plays an important role in the development of science. There is a wide range of models and approaches that depend on the information available and the goal of the problem. In this dissertation we focus on three problems whose solution depends on parameters and for which we have either data resulting from simulations of the model or a explicit structure that describes the model. First, for the case when only data are available, we develop an algorithm that builds a data-driven approximation that is then easy to reevaluate. Second, we embed our algorithm in an already developed framework for the solution of a specific kind of model structure: nonlinear eigenvalue problems. Third, given a model with a specific nonlinear structure, we develop a method to build a model with the same structure, smaller dimension (for faster computation), and that provides an accurate approximation of the original model.

Dedication

A mis padres Otilia y Eloy.

A mi hermano Miguel.

A la memoria de mis abuelas Aurina y Otilia.

Acknowledgments

I will be forever grateful to my advisor Dr. Serkan Gugercin for his invaluable dedication to mentoring that has helped me grow as a learner, researcher, and ultimately, as a person in a healthy joyful way.

Equal in measure is my gratitude for my committee members Dr. Jeff Borggaard, Dr. Chris Beattie, Dr. Traian Iliescu, and Dr. Mark Embree. My motivation and enjoyment throughout this journey has been due in no small part to their contagious passion for research.

To my family, Danny, and friends, old and new, thank you for always making my days brighter.

Contents

List of Figures

List of Tables

List of Abbreviations

Notation

$H(s)$ Original or unknown model (single variable)

$H(s,p)$ Original or unknown model (two variables)

$\widetilde{H}(s)$ Approximation of $H(s)$

$\widetilde{H}(s,p)$ Approximation of $H(s,p)$

s Frequency variable

p Parameter variable

$\mathbf{p}$ Parameter vector variable

σ Frequency value where interpolation is attained

$\boldsymbol{\sigma}$ Frequency vector values where interpolation is attained

π Parameter value where interpolation is attained

$\boldsymbol{\pi}$ Parameter vector values where interpolation is attained

$\otimes$ Kronecker product

$\odot$ Hadamard product

$\mathbf{1}_{m\times n}$ Matrix of dimension $m \times n$ with all entries equal to 1

$\mathbf{e}_i$ ith unit vector

Dynamical Systems

MIMO Multi-Input/Multi-Output

SISO Single-Input/Single-Output

Related Abbreviations

B-IRKA Bilinear IRKA

BT Balanced Truncation

FOM Full Order Model

IRKA Iterative Rational Krylov Algorithm

MOR Model Order Reduction

PBMOR Bilinear PMOR

PMOR Parametric MOR

ROM Reduced Order Model

TB-IRKA Truncated B-IRKA

Data Fitting

AAA Adaptive Antoulas-Anderson

LS Least Squares

p-AAA Parametric AAA

VF Vector Fitting

Other

DEIM Discrete Empirical Interpolation Method

FEM Finite Element Method

ODE Ordinary Differential Equation

PDE Partial Differential Equation

PNLEVP Parametric Nonlinear Eigenvalue Problem

POD Proper Orthogonal Decomposition

SVD Singular Value Decomposition

Chapter 1

Introduction

Physical phenomena are often described by a partial differential equation (PDE) that relates a quantity of interest (position, temperature, pressure, etc.), its partial derivatives with respect to time and space, and input forces. The most common approach to finding the solution of a PDE is to find a spatial discretization (via, e.g., finite elements, finite differences, or finite volumes), resulting in a system of ordinary differential equations (ODEs) that we call a dynamical system. Hence simulation of dynamical systems is an essential tool in the development of science and engineering. However, the need for high accuracy results in systems with such a large number of degrees of freedom that numerical simulations become too costly. Model order reduction (MOR) tackles this issue by constructing a substantially lower dimensional representation of the full-order dynamical system, which is cheap to simulate, yet provides high-fidelity, i.e., it provides an accurate approximation of the original quantities of interest. In many applications such as optimization, design, control, uncertainty quantification, and inverse problems, the dynamics of these systems are defined by a set of parameters that describe initial and boundary conditions, material and medium properties, etc. Since carrying out model reduction for every parameter value is not computationally feasible, the goal in the parameterized setting is to construct a parametric reduced model that can approximate one or more quantities of interest well for the whole parameter range of interest. This leads to the parametric model reduction (PMOR) framework.

As an example, consider the following input-output dynamical system

$$\mathbf{E}\dot{\mathbf{x}}(t,p) = \mathbf{A}(p)\mathbf{x}(t,p) + \mathbf{b}u(t), \qquad y(t,p) = \mathbf{c}^\top \mathbf{x}(t,p), \tag{1.1}$$

where $p \in \mathcal{P} \subset \mathbb{R}$ represents the parametric variation in $\mathbf{A}(p) \in \mathbb{R}^{n_{st}\times n_{st}}$; $\mathbf{b}, \mathbf{c} \in \mathbb{R}^{n_{st}}$ are constant; $f(t) \in \mathbb{R}$ is the input (forcing term); $y(t,p) \in \mathbb{R}$ is the output (quantity of interest); and $\mathbf{x}(t,p) \in \mathbb{R}^{n_{st}}$ is the state (internal degrees of freedom). Assuming invertible $\mathbf{E}$ and zero initial conditions, i.e., $\mathbf{x}(0) = \mathbf{0}$, the output $y(t,p)$ can be expressed using the convolution integral

$$y(t,p) = \int_0^t \mathbf{c}^\top e^{(t-\tau)\mathbf{E}^{-1}\mathbf{A}(p)}\mathbf{E}^{-1}\mathbf{b}u(\tau)\, d\tau. \tag{1.2}$$

When the system dimension n_{st} is large, evaluating the quantity of interest $y(t,p)$ repeatedly for different parameter values becomes computationally demanding. The goal of PMOR is to find a dynamical system of dimension $\widetilde{n}_{st} \ll n_{st}$ so that re-evaluations of the system are significantly cheaper, yet accurately capture $y(t,p)$.

For specific details on both parametric and nonparametric model reduction, we refer the reader to [6, 8, 17, 26, 27, 77] and the references therein. We will include some of these details in later chapters. Projection-based PMOR methods have been successfully developed for systems with known internal description as in (1.1), i.e., the full-order operators $\mathbf{A}(p), \mathbf{b}$ and $\mathbf{c}$ are available; see, e.g., the recent survey papers and books [13, 27, 77, 115] for a detailed analysis of projection-based approaches to PMOR.

In many cases internal description of a system is not accessible and only input/output measurements are available. In our setting, for parametric dynamical systems such as (1.1), input/output measurements/data will correspond to the samples of *the transfer function* of (1.1), i.e., the samples of

$$H(s,p) = \mathbf{c}^\top(s\mathbf{E} - \mathbf{A}(p))^{-1}\mathbf{b}, \tag{1.3}$$

where $H(s,p)$ is the Laplace transform of the convolution kernel $h(t) = \mathbf{c}^\top e^{t\mathbf{E}^{-1}\mathbf{A}(p)}\mathbf{E}^{-1}\mathbf{b}$ in (1.2). Then, given the samples

$$h_{ij} = H(s_i, p_j) \in \mathbb{C} \qquad \text{for } i = 1, \ldots, N \text{ and } j = 1, \ldots, M, \tag{1.4}$$

our goal is to build a two-variable rational function with barycentric representation

$$\widetilde{H}(s,p) = \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\beta_{ij}}{(s-\sigma_i)(p-\pi_j)} \Big/ \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij}}{(s-\sigma_i)(p-\pi_j)},$$

that approximates the data in (1.4) in an appropriate measure. In Chapter 2 we will introduce our method to tackle this problem. Since our method approximates a function from data, it may be applied to any circumstance in which such data are available. Note that if access to the state-space representation is not available but data from black-box simulations is, then data-driven PMOR may not be an option but a necessity. In such cases there is no order reduction, since we are creating a dynamical system from measurements.

Even though our motivation comes from approximating parametric dynamical systems, similar approximation problems may arise under different circumstances. For example, consider modeling a stationary PDE, such as

$$w_{xx} + pw_{yy} + zw = f(x,y) \qquad \text{on} \qquad \Omega = [a,b] \times [c,d],$$

with appropriately defined initial and boundary conditions. A spatial discretization on Ω yields

$$\mathbf{A}(p,z)\mathbf{w} = \mathbf{b}.$$

Then, the samples of $H(p,z) = \mathbf{A}(p,z)^{-1}\mathbf{b}$ can be used to build an approximation to the solution $w(x,y)$. We study two such problems in Section 2.3.2.

Another example of rational approximation that does not necessarily lie in the MOR field is the parametric nonlinear eigenvalue problem (PNLEVP):

$$\begin{gathered}\text{Given the nonlinear function } \mathbf{T} : \mathbb{C} \times \mathbb{R} \to \mathbb{C}^{n\times n} \\ \text{find } \lambda \in \mathbb{C},\ \mathbf{x}, \mathbf{y} \in \mathbb{C}^n \setminus \{\mathbf{0}\} \text{ such that} \\ \mathbf{T}(\lambda, p)\mathbf{x} = \mathbf{0} \text{ and } \mathbf{y}^*\mathbf{T}(\lambda, p) = \mathbf{0} \\ \text{for some } p \in \mathbb{R},\end{gathered}$$

which we will discuss in Chapter 3. As an example consider the following problem, which we will revisit in Section 3.3.3:

$$\mathbf{T}(\lambda, p) = \mathbf{K} - \lambda^2\mathbf{M} + \underbrace{\frac{G_0 + G_\infty(\iota\lambda p)^\alpha}{1 + (\iota\lambda p)^\alpha}}_{g(\lambda,p)}\mathbf{C},$$

where $\mathbf{K}, \mathbf{M}, \mathbf{C} \in \mathbb{R}^{168\times 168}$ and $G_0, G_\infty, \alpha \in \mathbb{R}$ are constant. This PNLEVP is the result of a finite element discretization of a clamped sandwich beam. The nonlinearity $g(\lambda, p)$ describes the shear modulus of the beam in terms of the static shear modulus $G_0 = 350.4$kPa, the asymptotic shear modulus $G_\infty = 3.062$MPa, the fractional parameter $\alpha = 0.675$, the relaxation time p, and the angular frequency λ. Note that we focus on problems that, for fixed parameter value $p = \pi \in \mathbb{R}$, find the eigentriples $(\lambda, \mathbf{x}, \mathbf{y})$ corresponding to $p = \pi$. Hence $(\lambda, \mathbf{x}, \mathbf{y})$ depend on p. This is not to be confused with multi-parameter eigenvalue problems (see, for example, [36])

$$\begin{aligned}\mathbf{A}_1\mathbf{x} &= \lambda\mathbf{B}_1\mathbf{x} + p\mathbf{C}_1\mathbf{x}, \\ \mathbf{A}_2\mathbf{y} &= \lambda\mathbf{B}_2\mathbf{y} + p\mathbf{C}_2\mathbf{y},\end{aligned}$$

where $\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i \in \mathbb{C}^{n\times n}$ for $i = 1, 2$ and the pair (λ, p) is an eigenvalue for nonzero $\mathbf{x}$

and $\mathbf{y}$. Eigenvalue problems hardly need any introduction due to their extensive research and history. In particular, the study of nonlinear eigenvalue problems (NLEVP) has seen many recent developments, e.g., see [70, 73, 98]. A collection of NLEVP commonly used for testing can be found in [34], and a library of algorithms is available in SLEPC [72]. These algorithms can be classified in three groups based on their approach: Newton methods, contour integration, and polynomial/rational approximation. In this thesis, we will make use of an algorithm in the latter class: the CORK [125] implementation of the rational Krylov method.

A plethora of work exists on model reduction of parametrized *linear* dynamical systems such as (1.1); see, for example, [4, 15, 22, 26, 39, 45, 46, 68, 107] and the references therein. However, in many applications the systems arising from modeling the underlying dynamics are *nonlinear*. In this thesis we will focus on a specific type of nonlinearity, namely bilinear dynamical systems. The nonlinearity results from adding a multiplication of the state and input to the linear model in (1.1), leading to the form

$$\mathbf{E}\dot{\mathbf{x}}(t,p) = \mathbf{A}(p)\mathbf{x}(t,p) + \mathbf{N}(p)\mathbf{x}(t,p)u(t) + \mathbf{b}u(t); \qquad y(t,p) = \mathbf{c}^\top \mathbf{x}(t,p), \tag{1.5}$$

where $\mathbf{N}(p) \in \mathbb{C}^{n_{st}\times n_{st}}$ and $\mathbf{N}(p)\mathbf{x}(t,p)u(t)$ is the bilinear term linking the input and state variables. Such systems may arise naturally from the problem they describe (see for example the references in [30] for population and economical dynamics, electrical circuits, plasma devices, and medical models [2, 99, 100, 106, 114, 118]) or they may be the result of approximating a general nonlinear system via Carleman bilinearization [40, 84]. We want to find a reduced bilinear system of the form

$$\widetilde{\mathbf{E}}\dot{\widetilde{\mathbf{x}}}(t,p) = \widetilde{\mathbf{A}}(p)\widetilde{\mathbf{x}}(t,p) + \widetilde{\mathbf{N}}(p)\widetilde{\mathbf{x}}(t,p)u(t) + \widetilde{\mathbf{b}}u(t); \qquad \widetilde{y}(t,p) = \widetilde{\mathbf{c}}^\top \widetilde{\mathbf{x}}(t,p),$$

where $\widetilde{\mathbf{E}}, \widetilde{\mathbf{A}}(p), \widetilde{\mathbf{N}}(p) \in \mathbb{C}^{\widetilde{n}_{st} \times \widetilde{n}_{st}}$, $\widetilde{\mathbf{b}} \in \mathbb{C}^{\widetilde{n}_{st}}$ with $\widetilde{n}_{st} \ll n_{st}$, and such that $\widetilde{y}(t,p)$ approximates $y(t,p)$ in (1.5) well. We are interested in input-independent (transfer-function-based) model reduction of parametric bilinear systems, where only the state-space matrices enter into the model reduction process and there is no need to choose a specific input $u(t)$ nor to simulate the full order model (1.5). As opposed to treating the dynamics as a general nonlinear system, we take advantage of the special bilinear nonlinearity. More specifically, we are focused on parametric model reduction that uses the concept of (parametric) rational interpolation. For details on interpolatory model reduction, see the recent book [13].

Main contributions and organization of the thesis. Our main contributions are threefold:

- Develop an algorithm for the rational approximation of multivariable functions from data. Our algorithm applies to both scalar and matrix-valued functions and the only choice the user needs to make is the stopping tolerance.
- Develop a new method for PNLEVP. We apply our algorithm to define a linearization of PNLEVP. Eigenvalues of our linearization can then be found using existing methods.
- Add to the field of PMOR via projection by stating and proving necessary conditions for interpolation of bilinear dynamical systems.

The rest of this dissertation is organized as follows:

- Chapter 2: We first review the AAA algorithm for data-driven rational approximation of one-variable scalar functions. We present our interpretation of AAA as a hybrid approach combining two frameworks. Then, we develop the first main contribution of this thesis, namely the p-AAA algorithm for data-driven modeling of two-variable

scalar functions. We discuss further generalization to more than two variables and matrix valued functions, and the impact of variable scaling.

- Chapter 3: This chapter presents the second major contribution of this dissertation. We combine our p-AAA algorithm with the CORK algorithm (for NLEVP) to solve PNLEVP. We define a new CORK linearization that includes the parameter dependency and the p-AAA approximation.

- Chapter 4: We extend interpolatory PMOR to bilinear systems, the third major contribution of this dissertation. We review the definition of the (infinitely many) transfer functions of a multi-input/multi-output bilinear dynamical system and what it means to interpolate them tangentially. We provide necessary conditions for tangential interpolation of the subsystems' transfer function and their derivatives.

Chapter 2

The p-AAA Algorithm

Consider a scalar-valued function $H(s,p)$ of two scalar variables s and p, and assume we only have access to its samples:

$$H(s_i,p_j)\in\mathbb{C} \qquad \text{for} \quad i=1,\ldots,N \quad \text{and} \quad j=1,\ldots,M.$$

Our goal is, then, to find a rational function $\widetilde{H}(s,p)$ that is a *good* approximation of $H(s,p)$. We will specify later how we measure *goodness*. As mentioned in Chapter 1, even though our motivation is that $H(s,p)$ represents the transfer function of a parametric dynamical system and we consider the variable s as frequency and p as the parameter, this is not restrictive and the approach can be considered as rational approximation of a multivariate function from its samples. In order to make the derivations clear, we first review, in Section 2.1, three of the existing algorithms for data-driven rational approximation in the single variable case: the Loewner framework [5, 7], the vector fitting method [69], and the AAA algorithm [105]. We highlight the similarities and differences among these three approaches. In Section 2.2, we present the proposed method, the parametric AAA (p-AAA) algorithm for data-driven modeling of parametric dynamical systems, which extends the AAA algorithm [105] to the multivariate case. In Section 2.3 we show how to apply the proposed methodology to matrix-valued functions. Throughout Sections 2.2 and 2.3, we use various examples to illustrate the success of the new methodology.

Much of the material in this chapter has already appeared in the submitted manuscript [41].

2.1 Revisiting the Single Variable Problem

In this section, we briefly revisit three approaches for the single variable case that are pertinent to our work. The single variable function to be approximated can be considered as the transfer function of a non-parametric dynamical system, for example.

Consider a single variable scalar-valued function $H(s)$ and assume access to its samples

$$h_i = H(s_i), \qquad s_i \in \mathbb{C}, \qquad \text{for} \;\; i = 1, \ldots, N. \tag{2.1}$$

Assume we do not have repeated samples, i.e., $s_i \neq s_j$ for $i \neq j$. The three methods we discuss will build a rational function $\widetilde{H}(s)$ that approximates the given data by means of interpolation, least squares (LS) minimization, or a combination of both. A key component in each case is the barycentric representation [33] of a rational function, given by

$$\widetilde{H}(s) = \frac{n(s)}{d(s)} = \frac{\displaystyle\sum_{i=1}^{k} \frac{\beta_i}{s - \sigma_i}}{\displaystyle\sum_{i=1}^{k} \frac{\alpha_i}{s - \sigma_i}}, \tag{2.2}$$

where $\sigma_i \in \mathbb{C}$ are the support (interpolation) points, a subset of the sampling points $\{s_1, \ldots, s_N\}$, and $\beta_i, \alpha_i \in \mathbb{C}$ are the weights to be determined. The algorithms we describe will differ from each other in how they choose σ_i's, α_i's, and β_i's. Note that $\widetilde{H}(s)$ is a proper rational function of order $k - 1$.

2.1.1 The barycentric rational interpolant via Loewner matrices

Given the data (samples) in (2.1), the Loewner approach [5, 7] builds the rational approximant $\widetilde{H}(s)$ in (2.2) that interpolates the data (assuming a rational interpolant of degree $k-1$ exists). Partition the sampling points and the corresponding function values:

$$\{s_1, \ldots, s_N\} = \{\sigma_1, \ldots, \sigma_k\} \cup \{\hat{\sigma}_1, \ldots, \hat{\sigma}_{N-k}\} \text{ and}$$
$$\{h_1, \ldots, h_N\} = \{g_1, \ldots, g_k\} \cup \{\hat{g}_1, \ldots, \hat{g}_{N-k}\}.$$

Interpolation at $\{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ is attained by choosing

$$\beta_i = g_i \alpha_i, \tag{2.3}$$

provided the α_i's are nonzero. For interpolation at $\hat{\sigma}_i$, for $i = 1, 2, \ldots, N-k$, we set

$$H(\hat{\sigma}_i) - \widetilde{H}(\hat{\sigma}_i) = \hat{g}_i - \frac{n(\hat{\sigma}_i)}{d(\hat{\sigma}_i)} = \hat{g}_i - \sum_{j=1}^{k} \frac{g_j \alpha_j}{\hat{\sigma}_i - \sigma_j} \Bigg/ \sum_{j=1}^{k} \frac{\alpha_j}{\hat{\sigma}_i - \sigma_j} = 0.$$

Multiplying out with the denominator, we obtain

$$\hat{g}_i \sum_{j=1}^{k} \frac{\alpha_j}{\hat{\sigma}_i - \sigma_j} - \sum_{j=1}^{k} \frac{g_j \alpha_j}{\hat{\sigma}_i - \sigma_j} = \sum_{j=1}^{k} \frac{(\hat{g}_i - g_j)\alpha_j}{\hat{\sigma}_i - \sigma_j} = \mathbf{e}_i^\top \mathbb{L}\mathbf{a} = 0,$$

where $\mathbf{e}_i \in \mathbb{R}^{N-k}$ denotes the ith unit vector, $\mathbf{a}^\top = [\alpha_1 \cdots \alpha_k]$, and $\mathbb{L} \in \mathbb{C}^{(N-k)\times k}$ is the Loewner matrix given by

$$\mathbb{L} = \begin{bmatrix} \frac{\hat{g}_1 - g_1}{\hat{\sigma}_1 - \sigma_1} & \cdots & \frac{\hat{g}_1 - g_k}{\hat{\sigma}_1 - \sigma_k} \\ \vdots & \ddots & \vdots \\ \frac{\hat{g}_{N-k} - g_1}{\hat{\sigma}_{N-k} - \sigma_1} & \cdots & \frac{\hat{g}_{N-k} - g_k}{\hat{\sigma}_{N-k} - \sigma_k} \end{bmatrix}. \tag{2.4}$$

Hence to enforce interpolation at $\{\hat{\sigma}_1, \hat{\sigma}_2, \ldots, \hat{\sigma}_{N-k}\}$, the unknown coefficient vector

$$\mathbf{a} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_k \end{bmatrix}$$

is obtained by solving the linear system

$$\mathbb{L}\mathbf{a} = \mathbf{0}. \tag{2.5}$$

Here, we skip the details for the conditions on $\mathbb{L}$ and its null space to guarantee the existence and uniqueness of a degree $k-1$ rational interpolant of the form (2.2) and refer the reader to [7, 13] for details. A simple case to consider is when $N = 2k-1$. In this case, the Loewner matrix is $\mathbb{L} \in \mathbb{C}^{(k-1)\times k}$, with, at least, a one-dimensional nullspace. Considering the fact that a proper rational function of degree $k-1$ has $2k-1$ degrees of freedom (after normalization of the highest coefficient in the denominator), choosing $N = 2k-1$ will yield a unique rational interpolant (under certain conditions [7, 13]). By introducing the notion of the shifted Loewner matrix, in [96] the Loewner approach has been extended to a state-formulation, where the rational interpolant can be directly written in a state-space form, as in (1.3), without forming the barycentric form. However, for the parametric problems, the barycentric formulation is the key and we refer the reader to [12, 13, 96] and the references therein for the state-space based Loewner construction for modeling nonparametric dynamical systems.

Remark 2.1. The Loewner framework described above applies to linear approximations such as (1.3). Similar Loewner and shifted Loewner matrices can be built from measurements of subsystems' transfer functions to construct a bilinear system that interpolates the given measurements [11]. The Loewner framework has been further generalized to quadratic-bilinear systems [60]. We note that the data-driven frameworks we discuss here, both for

linear and nonlinear dynamical systems, only use input-output measurements in the form of transfer function measurements and do not require access to state measurement. See, for example, [19, 29, 38, 49, 86, 109, 128], and the references therein for some data-driven methods for nonlinear systems that work with the measurements of the internal state variable.

2.1.2 VF for rational LS approximation

Instead of constructing a rational approximation that interpolates the data, one can also consider fitting the data in a least-squares (LS) sense. Thus, given the samples (2.1), the goal is now to construct a rational function $\widetilde{H}(s)$ that minimizes the LS error

$$\sum_{i=1}^{N} |\widetilde{H}(s_i) - h_i|^2.$$

There are various approaches to solving rational LS approximation problems with measured data; see, e.g., [32, 48, 58, 59, 69, 78, 79, 91, 117] and the references therein. Due to its close connection to the barycentric form we consider here, we briefly review the vector fitting (VF) method of [69].

VF starts with a slightly revised version of $\widetilde{H}(s)$ with the form

$$\widetilde{H}(s) = \frac{n(s)}{d(s)} = \frac{\displaystyle\sum_{i=1}^{k} \frac{\beta_i}{s-\sigma_i}}{1 + \displaystyle\sum_{i=1}^{k} \frac{\alpha_i}{s-\sigma_i}} + d_1 + se_1. \tag{2.6}$$

A fundamental difference from the interpolation framework of Section 2.1 is that $\{\sigma_i\}$ in (2.6) are *not* a subset of sampling points; they are chosen independently, and in VF are updated at every step. The choice of $\{\sigma_i\}$ in (2.6) will be clarified later. The additional

"1" in the denominator guarantees that the first term in $\widetilde{H}(s)$ is strictly proper. The term $d_1 + se_1$, if needed, allows polynomial growth around $s = \infty$, which could be necessary in approximating transfer functions corresponding to differential algebraic equations [24, 67, 97]. These details are not fundamental to the focus of this chapter; therefore we skip those and assume $d_1 = e_1 = 0$. For details, we refer the reader to [65, 69].

Using (2.6), the LS error can be written as

$$\sum_{i=1}^{N} |\widetilde{H}(s_i) - h_i|^2 = \sum_{i=1}^{N} \frac{1}{|d(s_i)|^2} |n(s_i) - d(s_i)h_i|^2.$$

This is a nonlinear LS problem. Starting with an initial guess $d^{(0)}(s)$, Sanathanan and Koerner [117] convert this nonlinear LS problem into a sequence of weighted linear LS problems, which we will call the SK iteration:

$$\min_{n^{(j+1)}, d^{(j+1)}} \sum_{i=1}^{N} \left| \frac{n^{(j+1)}(s_i) - d^{(j+1)}(s_i)h_i}{d^{(j)}(s_i)} \right|^2, \quad j = 0, 1, 2, \ldots.$$

Note that the problem is now linear in the unknowns $n^{(j+1)}(s)$ and $d^{(j+1)}(s)$. The SK iteration uses the monomial basis for $n(s)$ and $d(s)$. VF, instead, uses the barycentric form (2.6), which proves to be the crucial step, since it allows for updating $\{\sigma_i\}$ in each step. VF updates $\{\sigma_i\}$ as the zeros of the denominator $d^{(j)}(s)$ from the previous iteration, i.e., $d^{(j)}(\sigma_i^{(j+1)}) = 0$. After a proper rescaling, this results in a sequence of unweighted linear LS minimization problems of the form

$$\min_{\mathbf{a}^{(j+1)}} \left\| \mathcal{A}^{(j)} \mathbf{a}^{(j+1)} - \mathbf{h} \right\|_2,$$

where $\mathbf{h} = [h_1 \ \cdots \ h_N]^\top$, $\mathbf{a} = [\beta_1 \ \cdots \ \beta_k \ \alpha_1 \ \cdots \ \alpha_k]^\top$, and $\mathcal{A}^{(j)}$ is given by

$$\mathcal{A}^{(j)} = \begin{bmatrix} \frac{1}{s_1-\sigma_1^{(j)}} & \frac{1}{s_1-\sigma_2^{(j)}} & \cdots & \frac{1}{s_1-\sigma_k^{(j)}} & \frac{-h_1}{s_1-\sigma_1^{(j)}} & \frac{-h_1}{s_1-\sigma_2^{(j)}} & \cdots & \frac{-h_1}{s_1-\sigma_k^{(j)}} \\ \frac{1}{s_2-\sigma_1^{(j)}} & \frac{1}{s_2-\sigma_2^{(j)}} & \cdots & \frac{1}{s_2-\sigma_k^{(j)}} & \frac{-h_2}{s_2-\sigma_1^{(j)}} & \frac{-h_2}{s_2-\sigma_2^{(j)}} & \cdots & \frac{-h_2}{s_2-\sigma_k^{(j)}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{1}{s_N-\sigma_1^{(j)}} & \frac{1}{s_N-\sigma_2^{(j)}} & \cdots & \frac{1}{s_N-\sigma_k^{(j)}} & \frac{-h_N}{s_N-\sigma_1^{(j)}} & \frac{-h_N}{s_N-\sigma_2^{(j)}} & \cdots & \frac{-h_N}{s_N-\sigma_k^{(j)}} \end{bmatrix} \in \mathbb{C}^{N\times 2k}.$$

Note that the Loewner matrix $\mathbb{L}$ appearing in the interpolation setting of Section 2.1.1 is now replaced with $\mathcal{A}^{(j)}$, which consists of a Cauchy and a diagonally-scaled Cauchy matrix. Despite dependence on the barycentric form, there is a fundamental difference from the Loewner framework of Section 2.1.1: The coefficients $\{\alpha_i\}$ and $\{\beta_i\}$ in the barycentric form are chosen independently to minimize the LS error. This is in contrast to the Loewner setting where one sets $\beta_i = h_i\alpha_i$ to enforce interpolation. Moreover, the points $\{\sigma_i\}$ are updated at every step.

Convergence of VF is an open question. Even though one can construct examples where the iteration does not converge [90], its behavior in practice is more robust. When the initial set $\{\sigma_i\}$ is chosen appropriately, the algorithm usually converges quickly. As VF converges, due to the updating scheme of $\{\sigma_i\}$, the denominator $d^{(k)}(s)$ converges to 1 and one obtains a pole-residue formulation for $\widetilde{H}(s)$. However, this is not needed. The algorithm can be terminated early with $\widetilde{H}(s)$ having the barycentric form, as in (2.6).

2.1.3 The AAA algorithm

Given the samples $\{H(s_i)\}_{i=1}^N$, we have seen two frameworks for constructing a rational approximant: the barycentric rational interpolation via Loewner matrices (Section 2.1.1) and the rational LS approximation via VF (Section 2.1.2). Both methods depend on the

barycentric form and differ in how they choose the variables in this representation. The Adaptive Anderson-Antoulas (AAA) algorithm developed by Nakatsukasa et al. [105] is an iterative algorithm that elegantly integrates these two frameworks (interpolation and LS), combining their strengths, leading to a powerful framework for rational approximation.

As in Section 2.1.1, we partition the sampling points $\{s_i\}$ and the samples $\{h_i\}$ into two disjoint data sets:

$$\begin{aligned} \text{sampling points: } \{s_1,\ldots,s_N\} &= \{\ \sigma_1,\ldots,\sigma_k\ \} \cup \{\ \hat{\sigma}_1,\ldots,\hat{\sigma}_{N-k}\ \} \stackrel{\text{def}}{=\!=} \boldsymbol{\sigma}\ \cup\ \hat{\boldsymbol{\sigma}} \quad \text{and} \\ \text{sampled values: } \{h_1,\ldots,h_N\} &= \{\ g_1,\ldots,g_k\ \} \cup \{\ \hat{g}_1,\ldots,\hat{g}_{N-k}\ \} \stackrel{\text{def}}{=\!=} \mathbf{g}\ \cup\ \hat{\mathbf{g}}\ . \end{aligned} \tag{2.7}$$

This partitioning will be clarified later. Assume the barycentric form for $\widetilde{H}(s)$ as in (2.2), which we repeat here:

$$\widetilde{H}(s) = \frac{n(s)}{d(s)} = \sum_{i=1}^{k} \frac{\beta_i}{s-\sigma_i} \Bigg/ \sum_{i=1}^{k} \frac{\alpha_i}{s-\sigma_i}. \tag{2.2}$$

Now assume that we want to enforce interpolation at the points in $\boldsymbol{\sigma}$. Therefore, in (2.2) we set $\beta_i = g_i\alpha_i$ for $i = 1, 2, \ldots, k$, as we did in Section 2.1.1. However, as opposed to enforcing interpolation on $\hat{\boldsymbol{\sigma}}$ as well, AAA chooses the coefficients $\{\alpha_i\}$ to minimize the LS error over the remaining sampling points $\hat{\boldsymbol{\sigma}}$.

As in Section 2.1.2, the LS problem over the sampling points $\hat{\boldsymbol{\sigma}}$ is nonlinear due to dependence on the denominator $d(s)$. The VF algorithm used the SK-iteration to convert this nonlinear LS problem to a sequence of linearized LS problems. AAA uses a different linearization. More

precisely, for the point $\hat{\sigma}_i \in \hat{\boldsymbol{\sigma}}$, AAA uses the linearization

$$H(\hat{\sigma}_i) - \widetilde{H}(\hat{\sigma}_i) = \hat{g}_i - \frac{n(\hat{\sigma}_i)}{d(\hat{\sigma}_i)} = \frac{1}{d(\hat{\sigma}_i)} \left(\hat{g}_i d(\hat{\sigma}_i) - n(\hat{\sigma}_i)\right) \tag{2.8}$$

$$\rightsquigarrow \hat{g}_i d(\hat{\sigma}_i) - n(\hat{\sigma}_i) = \sum_{j=1}^{k} \frac{(\hat{g}_i - g_j)\alpha_j}{\hat{\sigma}_i - \sigma_j} = \mathbf{e}_i^\top \mathbb{L}\mathbf{a}, \tag{2.9}$$

where $\mathbb{L}$ is the Loewner matrix defined as in (2.4) and $\mathbf{a} = [\alpha_1 \ \cdots \ \alpha_k]^\top$. Then, the linearized LS problem (over $\hat{\boldsymbol{\sigma}}$) to compute the coefficient vector $\mathbf{a}$ becomes

$$\min_{\|\mathbf{a}\|_2 = 1} \|\mathbb{L}\mathbf{a}\|_2 . \tag{2.10}$$

Before elaborating on how AAA partitions the data set for interpolation and LS, we point out the difference between (2.5) and (2.10) in determining $\mathbf{a}$. In the interpolation case, assuming that there exists an underlying degree $k-1$ rational interpolant, the Loewner matrix has a null space and thus we solve $\mathbb{L}\mathbf{a} = 0$. On the other hand, in the case of the linearized LS problem in AAA, such a rational interpolant does not generally exist (consider it as too many data points and not enough degrees of freedom), and one solves the minimization problem (2.10) by choosing $\mathbf{a}$ as the right singular vector corresponding to the smallest singular value of $\mathbb{L}$.

AAA iteratively partitions the data using a greedy search at each step. Let $\widetilde{H}(s)$ denote the AAA approximant at step k corresponding to the interpolation/LS data partitioning in (2.7). The next sampling point, σ_{k+1}, to be added to the interpolation set $\boldsymbol{\sigma}$, is determined by finding $\hat{\sigma}_i$ for which the current error is maximal, i.e.,

$$\sigma_{k+1} = \underset{i=1,\ldots,N-k}{\arg\max} \left| H(\hat{\sigma}_i) - \widetilde{H}(\hat{\sigma}_i) \right| .$$

Then, the algorithm proceeds by updating the interpolation and LS data partition (2.13),

setting $\beta_{k+1} = g_{k+1}\alpha_{k+1}$, and by solving (2.10) for the updated coefficient vector. AAA is terminated when either a pre-specified error tolerance or an order is achieved. We refer the reader to the original source [105] for details. We also note that a similar greedy search for computing interpolation points was proposed in [50, 54, 55] in projection-based interpolatory model reduction and in [89] in Loewner-based interpolatory modeling. As AAA proceeds, a new column is added to $\mathbb{L}$ at every step. Therefore, assuming a large number of data points N, the matrix $\mathbb{L}$ in AAA is tall and skinny, and thus generically does not have a null space. However, if $\mathbb{L}$ happens to have a nullspace after a certain iteration index, the AAA approximant will interpolate the full data set and coincide with the rational interpolant of Section 2.1.1, assuming a unique solution.

The AAA algorithm has proven very successful and has been employed in many applications, including nonlinear eigenvalue problems [92], linear dynamical systems with quadratic output [61], rational minimax approximation [56], and rational approximations over disconnected domains [105]. Our goal in the following sections is to extend AAA to approximating parametric (dynamical) systems from their samples.

2.1.4 AAA modifications

Since VF treats the LS problem differently than AAA, we analyze how incorporating tools from VF would impact the AAA algorithm. In particular, we study the following two modifications.

Enforcing strict properness. In the original AAA paper [105], the authors discuss various adaptations of the *core* AAA algorithm, among which is included the case where the degree of the numerator and denominator are not equal. Details can be found in [33] and [56] for the polynomial case and for the general rational case, respectively. Motivated by our dynamical systems background, we focus here on an option to guarantee that the ap-

proximation vanishes as $s \to \infty$. Consider the representation in (2.6) with the interpolation condition (2.3), to obtain an approximation of the form

$$\widetilde{H}(s) = \frac{\displaystyle\sum_{i=1}^{k} \frac{g_i \alpha_i}{s - \sigma_i}}{1 + \displaystyle\sum_{i=1}^{k} \frac{\alpha_i}{s - \sigma_i}},$$

and consider the simplification (2.9). Then the LS problem (2.10) becomes

$$\min_{\|\mathbf{a}\|_2 = 1} \|\mathbb{L}\mathbf{a} - \hat{\mathbf{g}}\|_2 \,,$$

where

$$\hat{\mathbf{g}} = \begin{bmatrix} \hat{g}_1 \\ \vdots \\ \hat{g}_{N-k} \end{bmatrix},$$

corresponding to the partition (2.7) resulting from the greedy search.

Adding $1/d(s)$ as a weight. It was pointed out in [105, §10] that one can introduce weighted norms in the LS problem in every step of AAA by scaling the rows of the Loewner matrix. Inspired by the SK iteration and VF, another type of weighting can be introduced by modifying the linearization step (2.9) in AAA as

$$H(\hat{\sigma}_i) - \widetilde{H}(\hat{\sigma}_i) = \frac{1}{d(\hat{\sigma}_i)} \left(\hat{g}_i d(\hat{\sigma}_i) - n(\hat{\sigma}_i)\right) \rightsquigarrow \frac{1}{d^{-}(\hat{\sigma}_i)} \left(\hat{g}_i d(\hat{\sigma}_i) - n(\hat{\sigma}_i)\right),$$

where $d^{-}(s)$ denotes the denominator of the AAA approximation from the previous step, thus keeping the error still linear in the variables $n(s)$ and $d(s)$ to be computed. Then the

coefficient vector $\mathbf{a}$ can be found by solving the weighted linear LS problem

$$\min_{\|\mathbf{a}\|_2=1} \|\Delta \mathbb{L}\mathbf{a}\|_2 ,$$

where Δ is a $(N-k)\times(N-k)$ diagonal matrix with the diagonal elements

$$\Delta_{ii} = 1/d^{-}(\hat{\sigma}_i).$$

Note that this weighting strategy by $1/d(s)$ focuses on adding weighting during the AAA iteration. In two recent works [56, 104] in the setting of rational minimax approximation, AAA is followed by the Lawson algorithm [88], an iteratively weighed LS iteration, yielding the AAA-Lawson method. The weighting in AAA-Lawson appears in the Lawson step, not in AAA.

In our numerical experiments, this revised implementation applied to various examples did not result in a significant advantage. The only improvement we observed, and only in some cases, was a reduction by one unit in the order of the rational approximation corresponding to the same error tolerance. As an illustrative representation, we show in Figure 2.1 the results of applying the AAA algorithm and these modifications to the Penzl example in Section 2.2.2.3 with $p = 10$, $N = 100$ logarithmically spaced samples, and stopping tolerance `1e-4`. Note that the modification that affects the approximation the most is enforcing strict properness. This results in an approximation of order $k = 7$ instead of $k = 8$ for the original AAA method, and an approximation that does not stagnate for large frequency values. Even though our weighted AAA method does not seem to have an impact, it does reduce the order of the approximation in some cases, when paired with our first modification. Due to these numerical observations, we do not investigate this further here or in the multivariate case below.

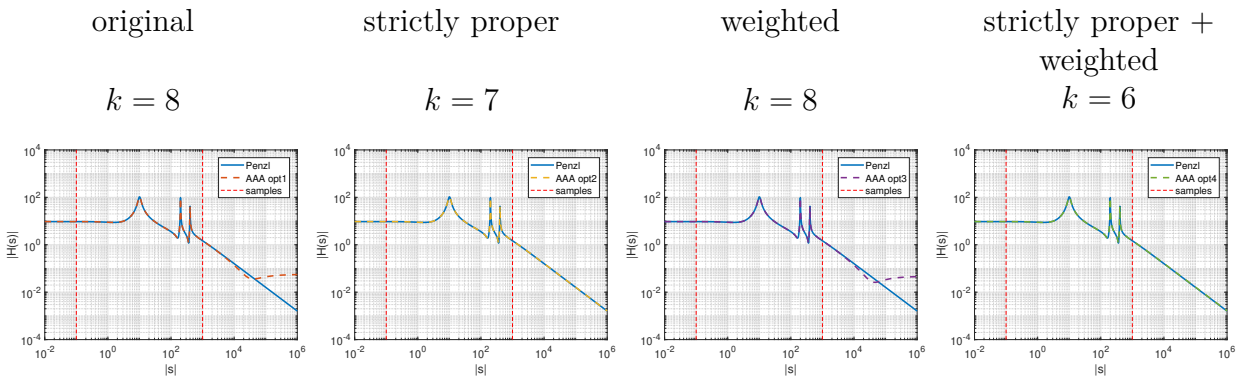


Figure 2.1: AAA modifications for the Penzl model

2.2 p-AAA: AAA for Parametric Dynamical Systems

In this section, we introduce the parametric AAA (p-AAA) algorithm, which extends AAA to multivariable problems appearing in the modeling of (the transfer function of) parametric dynamical systems. We start in Section 2.2.1 with the two-variable case first, illustrate its performance on various examples in Section 2.2.2, and discuss the effect of variable scaling in Section 2.2.3. Then, we briefly discuss how p-AAA can be applied to functions with more than two variables in Section 2.2.4, followed by an application to such an example. In this section, to simplify the initial discussion, we only focus on scalar-valued functions. The p-AAA method for matrix valued functions is discussed in Section 2.3.

2.2.1 p-AAA for the two-parameter case

We consider the problem of rational approximation of a multivariate function $H(s, p)$ from data. We assume only access to samples of $H(s, p)$, i.e., we have

$$h_{ij} = H(s_i, p_j) \in \mathbb{C} \qquad \text{for} \quad i = 1, \ldots, N \quad \text{and} \quad j = 1, \ldots, M. \tag{2.11}$$

Analogously to the single-variable case, we express the rational approximant $\widetilde{H}(s,p)$ in its *two-variable* barycentric form

$$\widetilde{H}(s,p) = \frac{n(s,p)}{d(s,p)} = \frac{\displaystyle\sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\beta_{ij}}{(s-\sigma_i)(p-\pi_j)}}{\displaystyle\sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij}}{(s-\sigma_i)(p-\pi_j)}}, \tag{2.12}$$

where $\{\sigma_i\}$ and $\{\pi_j\}$ are to-be-determined points, subsets of $\{s_i\}$ and $\{p_j\}$, respectively; and β_{ij} and α_{ij} are scalar coefficients to be chosen based on the interpolation and LS conditions to be enforced on the data (2.11). The number of points, k in the variable s, and q in the variable p, will be automatically determined by the algorithm.

We start by partitioning the data (2.11):

$$\begin{aligned}
\{s_1,\ldots,s_N\} = \{\sigma_1,\ldots,\sigma_k\} \cup \{\hat{\sigma}_1,\ldots,\hat{\sigma}_{N-k}\} &\stackrel{\text{def}}{=\!=} \boldsymbol{\sigma} \ \cup \ \hat{\boldsymbol{\sigma}}, \\
\{p_1,\ldots,p_M\} = \{\pi_1,\ldots,\pi_q\} \cup \{\hat{\pi}_1,\ldots,\hat{\pi}_{M-q}\} &\stackrel{\text{def}}{=\!=} \boldsymbol{\pi} \ \cup \ \hat{\boldsymbol{\pi}}, \text{ and} \\
\left[\begin{array}{c|c} [H(\sigma_i,\pi_j)] & [H(\sigma_i,\hat{\pi}_j)] \\ \hline [H(\hat{\sigma}_i,\pi_j)] & [H(\hat{\sigma}_i,\hat{\pi}_j)] \end{array}\right] &\stackrel{\text{def}}{=\!=} \left[\begin{array}{c|c} \mathbf{D}_{\boldsymbol{\sigma\pi}} & \mathbf{D}_{\boldsymbol{\sigma\hat{\pi}}} \\ \hline \mathbf{D}_{\boldsymbol{\hat{\sigma}\pi}} & \mathbf{D}_{\boldsymbol{\hat{\sigma}\hat{\pi}}} \end{array}\right],
\end{aligned} \tag{2.13}$$

where $[H(\sigma_i,\pi_j)] = \mathbf{D}_{\boldsymbol{\sigma\pi}}$ denotes the $k \times q$ matrix whose (i,j)th entry is $H(\sigma_i,\pi_j)$, and similarly for other quantities such as $[H(\sigma_i,\hat{\pi}_j)] = \mathbf{D}_{\boldsymbol{\sigma\hat{\pi}}}$. We use $\mathbf{D}_{\boldsymbol{\sigma\pi}}$ to denote the sampled data corresponding to the sampling points $(\boldsymbol{\sigma},\boldsymbol{\pi})$ (and similarly for other samples) instead of $\mathbf{H}_{\boldsymbol{\sigma\pi}}$, since $\mathbf{H}(s,p)$ will be used in Section 2.3 to denote matrix-valued (transfer) functions. How data is partitioned as in (2.13) will be clarified later. For a visual representation of partition (2.13) see Figure 2.2, where the outside labels refer to the variables' partitions and the colored rectangles symbolize the corresponding data partition matrices $\mathbf{D}_{\boldsymbol{\sigma\pi}}$, $\mathbf{D}_{\boldsymbol{\sigma\hat{\pi}}}$, $\mathbf{D}_{\boldsymbol{\hat{\sigma}\pi}}$, and $\mathbf{D}_{\boldsymbol{\hat{\sigma}\hat{\pi}}}$. Also in this figure we color code the two approaches that we will apply to our

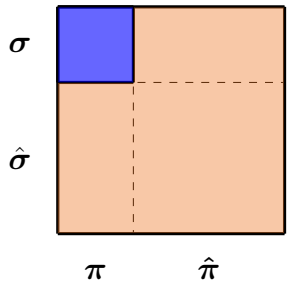


Figure 2.2: Illustration of the p-AAA data partition matrix for $[H(s_i, p_j)]$ corresponding to (2.13). In blue, the data to interpolate; In orange, the data to perform LS minimization.

data: we color in blue the part of the data that we will interpolate, and we color in orange the parts of the data that we approximate via LS. We discuss this in detail in the following paragraphs.

Interpolation of the sampled data $\mathbf{D}_{\boldsymbol{\sigma\pi}}$

In accordance with the partitioning of the data in (2.13), first we enforce interpolation at $(\boldsymbol{\sigma}, \boldsymbol{\pi})$, i.e., on the (1,1) block $\mathbf{D}_{\boldsymbol{\sigma\pi}}$, of the sampled data. This is achieved by setting, in (2.12),

$$\beta_{ij} = H(\sigma_i, \pi_j)\alpha_{ij}, \tag{2.14}$$

assuming $\alpha_{ij} \neq 0$. This follows from the fact that, as in the single variable case, the barycentric form $\widetilde{H}(s, p)$ in (2.12) has a removable singularity at (σ_i, π_j) with

$$\widetilde{H}(\sigma_i, \pi_j) = \frac{\beta_{ij}}{\alpha_{ij}},$$

and the choice (2.14) leads to interpolation of the data in $\mathbf{D}_{\boldsymbol{\sigma\pi}}$. This determines β_{ij}, and what remains to fully specify $\widetilde{H}(s, p)$ is the choice of α_{ij}.

LS fit for the uninterpolated data

The rational approximant $\widetilde{H}(s,p)$ in (2.12), with the choice (2.14), interpolates the data $\mathbf{D}_{\boldsymbol{\sigma\pi}}$. Next, we show how to choose α_{ij} so that $\widetilde{H}(s,p)$ minimizes the LS error in the remaining sampled data set in $\mathbf{D}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}}$, $\mathbf{D}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}}$, and $\mathbf{D}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}}$, i.e., to minimize

$$\|\boldsymbol{\varepsilon}\|_2 = \left\| \begin{bmatrix} \boldsymbol{\varepsilon}_1 \\ \boldsymbol{\varepsilon}_2 \\ \boldsymbol{\varepsilon}_3 \end{bmatrix} \right\|_2 \stackrel{\text{def}}{=\!=} \left\| \begin{bmatrix} \mathsf{vec}(\mathbf{D}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}}) \\ \mathsf{vec}(\mathbf{D}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}}) \\ \mathsf{vec}(\mathbf{D}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}}) \end{bmatrix} - \begin{bmatrix} \mathsf{vec}(\widetilde{H}(\boldsymbol{\sigma},\hat{\boldsymbol{\pi}})) \\ \mathsf{vec}(\widetilde{H}(\hat{\boldsymbol{\sigma}},\boldsymbol{\pi})) \\ \mathsf{vec}(\widetilde{H}(\hat{\boldsymbol{\sigma}},\hat{\boldsymbol{\pi}})) \end{bmatrix} \right\|_2. \tag{2.15}$$

As in the single variable case, the resulting LS problem is nonlinear, and we will linearize it similarly. To illustrate this more clearly, we rewrite the error for a sample $(\hat{\sigma},\hat{\pi})$ in the set $(\hat{\boldsymbol{\sigma}},\hat{\boldsymbol{\pi}})$ corresponding to a component in $\boldsymbol{\varepsilon}_3$ in (2.15) as

$$\begin{aligned} H(\hat{\sigma},\hat{\pi}) - \widetilde{H}(\hat{\sigma},\hat{\pi}) &= H(\hat{\sigma},\hat{\pi}) - \frac{n(\hat{\sigma},\hat{\pi})}{d(\hat{\sigma},\hat{\pi})} \\ &= \frac{1}{d(\hat{\sigma},\hat{\pi})}\left(H(\hat{\sigma},\hat{\pi})d(\hat{\sigma},\hat{\pi}) - n(\hat{\sigma},\hat{\pi})\right) \\ &\rightsquigarrow H(\hat{\sigma},\hat{\pi})d(\hat{\sigma},\hat{\pi}) - n(\hat{\sigma},\hat{\pi}) \quad \text{(linearization)} \\ &= H(\hat{\sigma},\hat{\pi})\sum_{i=1}^{k}\sum_{j=1}^{q}\frac{\alpha_{ij}}{(\hat{\sigma}-\sigma_i)(\hat{\pi}-\pi_j)} - \sum_{i=1}^{k}\sum_{j=1}^{q}\frac{H(\sigma_i,\pi_j)\alpha_{ij}}{(\hat{\sigma}-\sigma_i)(\hat{\pi}-\pi_j)} \\ &= \sum_{i=1}^{k}\sum_{j=1}^{q}\frac{(H(\hat{\sigma},\hat{\pi}) - H(\sigma_i,\pi_j))\alpha_{ij}}{(\hat{\sigma}-\sigma_i)(\hat{\pi}-\pi_j)} \\ &= \mathbf{e}_{\hat{\sigma}\hat{\pi}}^{\top}\,\mathbb{L}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}}\,\mathbf{a}, \end{aligned}$$

where

$$\mathbf{a}^{\top} = [\alpha_{11}\cdots\alpha_{1q} \mid \cdots \mid \alpha_{k1}\cdots\alpha_{kq}] \in \mathbb{C}^{kq}, \tag{2.16}$$

$\mathbb{L}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}} \in \mathbb{C}^{(N-k)(M-q)\times(kq)}$ is the 2D Loewner matrix[1] defined by

$$\mathbb{L}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}} = \left[\begin{array}{ccc|c} \frac{H(\hat{\sigma}_1,\hat{\pi}_1)-H(\sigma_1,\pi_1)}{(\hat{\sigma}_1-\sigma_1)(\hat{\pi}_1-\pi_1)} & \cdots & \frac{H(\hat{\sigma}_1,\hat{\pi}_1)-H(\sigma_1,\pi_q)}{(\hat{\sigma}_1-\sigma_1)(\hat{\pi}_1-\pi_q)} & \cdots \\ & \vdots & & \\ \frac{H(\hat{\sigma}_{N-k},\hat{\pi}_{M-q})-H(\sigma_1,\pi_1)}{(\hat{\sigma}_{N-k}-\sigma_1)(\hat{\pi}_{M-q}-\pi_1)} & \cdots & \frac{H(\hat{\sigma}_{N-k},\hat{\pi}_{M-q})-H(\sigma_1,\pi_q)}{(\hat{\sigma}_{N-k}-\sigma_1)(\hat{\pi}_{M-q}-\pi_q)} & \cdots \end{array}\right.$$
$$\left.\begin{array}{c|ccc} \cdots & \frac{H(\hat{\sigma}_1,\hat{\pi}_1)-H(\sigma_k,\pi_1)}{(\hat{\sigma}_1-\sigma_k)(\hat{\pi}_1-\pi_1)} & \cdots & \frac{H(\hat{\sigma}_1,\hat{\pi}_1)-H(\sigma_k,\pi_q)}{(\hat{\sigma}_1-\sigma_k)(\hat{\pi}_1-\pi_q)} \\ & \vdots & & \\ \cdots & \frac{H(\hat{\sigma}_{N-k},\hat{\pi}_{M-q})-H(\sigma_k,\pi_1)}{(\hat{\sigma}_{N-k}-\sigma_k)(\hat{\pi}_{M-q}-\pi_1)} & \cdots & \frac{H(\hat{\sigma}_{N-k},\hat{\pi}_{M-q})-H(\sigma_k,\pi_q)}{(\hat{\sigma}_{N-k}-\sigma_k)(\hat{\pi}_{M-q}-\pi_q)} \end{array}\right], \tag{2.17}$$

and $\mathbf{e}_{\hat{\sigma}\hat{\pi}} \in \mathbb{R}^{(N-k)(M-q)}$ is the unit vector with 1 in the entry corresponding to the sample $(\hat{\sigma}, \hat{\pi})$. Therefore, the linearized error $\boldsymbol{\varepsilon}_3$ is given by $\mathbb{L}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}}\,\mathbf{a}$.

The procedure follows similarly for the other blocks in (2.15). We rewrite the error corresponding to a sample $(\sigma_i, \hat{\pi}_\ell)$ in $\boldsymbol{\varepsilon}_1$ in (2.15) as

$$\begin{aligned} H(\sigma_i,\hat{\pi}_\ell) - \widetilde{H}(\sigma_i,\hat{\pi}_\ell) &= \left(\sum_{j=1}^{q} \frac{H(\sigma_i,\hat{\pi}_\ell) - H(\sigma_i,\pi_j)}{\hat{\pi}_\ell - \pi_j}\alpha_{ij}\right) \Bigg/ \sum_{j=1}^{q} \frac{\alpha_{ij}}{\hat{\pi}_\ell - \pi_j} \\ &\rightsquigarrow \sum_{j=1}^{q} \frac{H(\sigma_i,\hat{\pi}_\ell) - H(\sigma_i,\pi_j)}{\hat{\pi}_\ell - \pi_j}\alpha_{ij} \qquad \text{(linearization)} \\ &= \mathbf{e}_\ell^\top \mathbb{L}_{\sigma_i}\mathbf{a}_i, \end{aligned}$$

where $\mathbf{a}_i^\top = [\alpha_{i1} \ \cdots \ \alpha_{iq}] \in \mathbb{C}^q$ is the ith row block of $\mathbf{a}$, $\mathbf{e}_\ell \in \mathbb{C}^{M-q}$ is the ℓth unit vector, and $\mathbb{L}_{\sigma_i} \in \mathbb{C}^{(M-q)\times q}$ is the regular (1D) Loewner matrix corresponding to the data in the ith row of $[\mathbf{D}_{\boldsymbol{\sigma}\boldsymbol{\pi}} \ \mathbf{D}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}}]$, i.e.,

$$(\mathbb{L}_{\sigma_i})_{\ell,j} = \frac{H(\sigma_i,\hat{\pi}_\ell) - H(\sigma_i,\pi_j)}{\hat{\pi}_\ell - \pi_j} \quad \text{for} \quad \ell = 1,2,\ldots,M-q \quad \text{and} \quad j = 1,2,\ldots,q. \tag{2.18}$$

[1] Similar to the single-variable case, the Loewner matrices appearing in p-AAA here also appear in the parametric Loewner framework [10, 80] where one aims to interpolate the full data set. We revisit these connections in Remark 2.4.

Similar to [80], define

$$\mathbb{L}_{\boldsymbol{\sigma\hat{\pi}}} = \mathsf{diag}(\mathbb{L}_{\sigma_1}, \ldots, \mathbb{L}_{\sigma_k}) \in \mathbb{C}^{(k(M-q))\times(kq)}. \tag{2.19}$$

Then, the linearized error corresponding to $\boldsymbol{\varepsilon}_1$ in (2.15) is given by $\mathbb{L}_{\boldsymbol{\sigma\hat{\pi}}}\mathbf{a}$. Similarly, we can linearize and rewrite the error for the $\boldsymbol{\varepsilon}_2$-block in (2.15) as $\mathbb{L}_{\boldsymbol{\hat{\sigma}\pi}}\mathbf{a}$ where $\mathbb{L}_{\boldsymbol{\hat{\sigma}\pi}}$ is an assembly of all 1D Loewner matrices $\mathbb{L}_{\pi_j}$ corresponding to the data in each column of $\begin{bmatrix} \mathbf{D}_{\boldsymbol{\sigma\pi}} \\ \mathbf{D}_{\boldsymbol{\hat{\sigma}\pi}} \end{bmatrix}$. Putting all three together, after linearization, minimizing the LS error (2.15) in p-AAA becomes

$$\min_{\|\mathbf{a}\|_2=1} \|\mathbb{L}_2\mathbf{a}\|_2 \quad \text{where} \quad \mathbb{L}_2 = \begin{bmatrix} \mathbb{L}_{\boldsymbol{\sigma\hat{\pi}}}^\top & \mathbb{L}_{\boldsymbol{\hat{\sigma}\pi}}^\top & \mathbb{L}_{\boldsymbol{\hat{\sigma}\hat{\pi}}}^\top \end{bmatrix}^\top \in \mathbb{C}^{(MN-kq)\times kq}. \tag{2.20}$$

We summarize this analysis in a corollary.

Corollary 2.2. *Consider the data (2.13) and let the corresponding barycentric rational approximant $\widetilde{H}(s,p)$ have the form in (2.12).*

(a) If (2.14) holds, then

$$\widetilde{H}(\sigma_i, \pi_j) = H(\sigma_i, \pi_j), \quad i = 1, \ldots, k, \ j = 1, \ldots, q.$$

(b) Assume (2.14) holds. Choose the indices α_{ij} using

$$[\alpha_{11} \cdots \alpha_{1q} \mid \cdots \mid \alpha_{k1} \cdots \alpha_{kq}] = \mathbf{a}^\star, \quad \textit{where} \quad \mathbf{a}^\star = \operatorname*{arg\,min}_{\|\mathbf{a}\|_2=1} \|\mathbb{L}_2\mathbf{a}\|_2, \tag{2.21}$$

where $\mathbb{L}_2$ is as defined in (2.20), with $\mathbb{L}_{\boldsymbol{\hat{\sigma}\hat{\pi}}}$ as given by (2.17), $\mathbb{L}_{\boldsymbol{\sigma\hat{\pi}}}$ by (2.19) and (2.18),

and $\mathbb{L}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}}$ *as defined by*

$$\mathbb{L}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}} = \left[\begin{array}{ccc|c|ccc} \mathbb{L}_{\pi_1}\mathbf{e}_1 & & & \cdots & \mathbb{L}_{\pi_1}\mathbf{e}_k & & \\ & \ddots & & & & \ddots & \\ & & \mathbb{L}_{\pi_q}\mathbf{e}_1 & & & & \mathbb{L}_{\pi_q}\mathbf{e}_k \end{array}\right] \in \mathbb{C}^{(q(N-k))\times(kq)},$$

where

$$\mathbb{L}_{\pi_j}(\hat{\imath}, i) = \frac{H(\hat{\sigma}_{\hat{\imath}}, \pi_j) - H(\sigma_i, \pi_j)}{\hat{\sigma}_{\hat{\imath}} - \sigma_i}, \quad \hat{\imath} = 1, \ldots, N-k, \quad i = 1, \ldots, k,$$

and $\mathbf{e}_i \in \mathbb{C}^k$ *is the* i*th unit vector. Then, the two-variable barycentric approximant minimizes the linearized* LS *error*

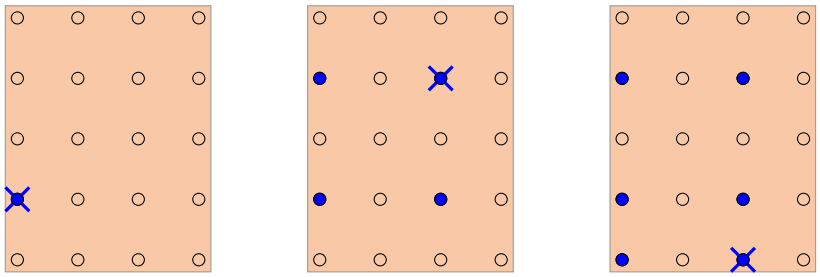
$$\widetilde{H} = \underset{\hat{H}=n/d}{\arg\min} \sum_{i,j} |H(s_i, p_j)d(s_i, p_j) - n(s_i, p_j)|^2,$$

with n *and* d *as in* (2.12)*, for the samples* (s_i, p_j) *corresponding to the error* $\boldsymbol{\varepsilon}$ *in* (2.15)*, i.e., for the data in* $\{\mathbf{D}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}}, \mathbf{D}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}}, \mathbf{D}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}}\}$.

Choosing the interpolated vs LS-fitted data

The last component of p-AAA is determining how to choose the data to be interpolated and the data to be fitted in the LS sense. Let $\widetilde{H}(s, p)$ in (2.12) be the current p-AAA approximant corresponding to the interpolation/LS partitioning in (2.13). Note that the order of the current approximation is $(k-1, q-1)$ and these orders need not be equal. Then, we select the next frequency-parameter tuple $(\sigma_{k+1}, \pi_{q+1})$ by means of the greedy search

$$(s_{\hat{\imath}}, p_{\hat{\jmath}}) = \underset{(s_i, p_j)}{\arg\max} |H(s_i, p_j) - \widetilde{H}(s_i, p_j)|. \tag{2.22}$$



iteration 1 iteration 2 iteration 3

Figure 2.3: Evolution of the p-AAA partition over iterations. In blue, the samples where interpolation is guaranteed by (2.14). A blue cross represents the sample selected by the greedy search in that iteration. The hollow circles correspond to the samples for which LS is performed.

We do not simply set $(\sigma_{k+1}, \pi_{q+1}) = (s_{\hat{\imath}}, p_{\hat{\jmath}})$ since one of the entries might already be in the previous interpolation data. In other words, $s_{\hat{\imath}}$ might already be in the set $\boldsymbol{\sigma}$ or $p_{\hat{\jmath}}$ might already be in the set $\boldsymbol{\pi}$ in (2.13). (This cannot occur for $s_{\hat{\imath}}$ and $p_{\hat{\jmath}}$ simultaneously, since we impose interpolation on the selected tuples. In other words, if the tuple $(s_{\hat{\imath}}, p_{\hat{\jmath}})$ was already in the interpolated data, we would have had $H(\sigma_{\hat{\imath}}, \pi_{\hat{\jmath}}) - \widetilde{H}(\sigma_{\hat{\imath}}, \pi_{\hat{\jmath}}) = 0$, which means the whole data set is interpolated.) If the point $\pi_{\hat{\jmath}}$ is already in the set $\boldsymbol{\pi}$ in (2.13), then the order in the variable p remains unchanged, as $q - 1$ and the set $\boldsymbol{\pi}$ is not altered. On the other hand, the point $s_{\hat{\imath}}$ is added to the set $\boldsymbol{\sigma}$ in (2.13) and the order in the variable s is increased to k. The operation is reversed if the point $\sigma_{\hat{\imath}}$ is already in the set $\boldsymbol{\sigma}$ instead. This allows updating the orders in each variable independently, giving the algorithm flexibility to make the decision automatically. We show an example of how the partition may evolve over iterations in Figure 2.3. In the first iteration, the first selection occurs, hence both (s_1, p_1) are added to the $\sigma, \boldsymbol{\pi}$ parts of the partition, making $(k, q) = (1, 1)$. In the second iteration, a completely different tuple is selected by the greedy search, hence making $(k, q) = (2, 2)$. Note that interpolation is not only attained at the tuples selected by the greedy search, but also at all the combination tuples. In the third iteration, one of the selected tuple values was already in the second iteration, and therefore only one variables' partition is updated,

making $(k, q) = (3, 2)$. Once the data partitioning (2.13) (and the orders) are updated, p-AAA computes the new coefficients β_{ij} as in (2.14), and then solves the LS problem (2.20) for the updated coefficient vector $\mathbf{a}$. The process is repeated until either a pre-specified error tolerance or desired orders in (s, p) are achieved. We give a brief sketch of p-AAA in Algorithm 2.3. We use the notation $[x_{ij}]$ to denote a matrix whose (i, j)th entry is x_{ij} and vec$(\cdot)$ to denote the vectorization of a matrix.

Algorithm 2.3. The p-AAA algorithm.

Given: $\{s_i\}$, $\{p_j\}$, and $\{h_{ij}\} = \{H(s_i, p_j)\}$ for

$i = 1, \ldots, N$ and $j = 1, \ldots, M$

Initialize: $k = 0$ and $q = 0$

Define $\widetilde{H} = average(h_{ij})$ and set error $\leftarrow \frac{\|\mathrm{vec}(h_{ij}) - \mathrm{vec}(\widetilde{H}(s_i,p_j))\|_\infty}{\|\mathrm{vec}(h_{ij})\|_\infty}$

while error $>$ desired tolerance **do**

Select $(s_{\hat{\imath}}, p_{\hat{\jmath}})$ by the greedy search (2.22)

Update the data partitioning (2.13):

if $s_{\hat{\imath}}$ was not selected at a previous iteration **then**

$k \leftarrow k + 1$

$\sigma_k \leftarrow s_{\hat{\imath}}$

end if

if $p_{\hat{\jmath}}$ was not selected at a previous iteration **then**

$q \leftarrow q + 1$

$\pi_q \leftarrow p_{\hat{\jmath}}$

end if

Build $\mathbb{L}_2$ as in (2.20)

Solve $\min \|\mathbb{L}_2\mathbf{a}\|_2$ s.t. $\|\mathbf{a}\|_2 = 1$

Use $\mathbf{a}$ to update the rational approximant $\widetilde{H}(s, p)$ with (2.12)–(2.14)

error $\leftarrow \frac{\|\mathrm{vec}(h_{ij}) - \mathrm{vec}(\widetilde{H}(s_i,p_j))\|_\infty}{\|\mathrm{vec}(h_{ij})\|_\infty}$

end while

return $\widetilde{H}$

Remark 2.4. *Parametric Loewner framework.* As in the single-variable case discussed in Section 2.1.1, one can choose to construct an approximation that interpolates the full-data (2.11), as done in [10, 80]. In this case, based on the ranks of Loewner matrices, the orders k

and q are chosen large enough so that, unlike in p-AAA, the matrix $\mathbb{L}_2$ has a null space, and thus one chooses the coefficient vector $\mathbf{a}$ by solving the linear system $\mathbb{L}_2\mathbf{a} = \mathbf{0}$. Therefore, the parametric Loewner framework [10, 80] interpolates the full data, in contrast to p-AAA, which greedily chooses a subset of data to interpolate and performs LS fit on the rest. When the orders k and q are not chosen *large enough*, the parametric Loewner framework no longer yields an interpolant, and instead an *approximate* interpolant is obtained. For details we refer the reader to [10, 12, 13, 80]. Even though this situation is more similar to the case of p-AAA, the major difference lies in the fact that p-AAA is an iterative algorithm and chooses the interpolation data with a greedy search while performing an LS fit on the rest. In other words, p-AAA decides the data-partitioning (2.13) automatically using a greedy search with an appropriately defined criterion. On the other hand, the parametric Loewner framework is a one-step algorithm, and how to partition the data is not yet fully understood. Even though there have been recent efforts in this direction for the single-variable case [53, 82, 83], this is still an open question, especially in the multivariate case. It will be worthwhile to investigate how the final data partitioning from p-AAA affects the parametric Loewner construction and whether it improves the conditioning-issues, appearing, at times, in the (one-step) Loewner framework.

2.2.2 Numerical examples

Next, we illustrate the performance of p-AAA on three numerical examples.

2.2.2.1 Synthetic transfer function

We use a simple model from [80], which is a low-order rational function in two variables. Consider

$$H(s,p) = \frac{1}{1+25(s+p)^2} + \frac{0.5}{1+25(s-0.5)^2} + \frac{0.1}{p+25}.$$

We sample this transfer function at $H(s_i, p_j)$ for $N = M = 21$ frequency and parameter points linearly spaced, $s_i \in [-1, 1]$ and $p_j \in [0, 1]$. This is a rational function with order $(4, 3)$. p-AAA, with tolerance 10^{-3}, terminates after 7 iterations. Table 2.1 shows the greedy search selection in each iteration step.

Table 2.1: Example 2.2.2.1: p-AAA samples selected in each iteration

p-AAA partition evolution				
iter.	greedy selection	update		size
#	(s_i, p_j)	σ_k	π_q	(k, q)
1	$(0, 0)$	0	0	$(1, 1)$
2	$(-1, 0)$	-1		$(2, 1)$
3	$(0.1, 0)$	0.1		$(3, 1)$
4	$(0, 1)$		1	$(3, 2)$
5	$(-1, 0.6)$		0.6	$(3, 3)$
6	$(-0.6, 0.1)$	-0.6	0.1	$(4, 4)$
7	$(0.6, 0.55)$	0.6	0.55	$(5, 5)$

Note that the p-AAA approximation $\widetilde{H}$ has order $(k-1, q-1) = (4, 4)$, as opposed to $(4, 3)$ of the original model. This is due to the greedy search selecting frequencies and parameters to interpolate as tuples, hence allowing for repetition. In Table 2.1 we see exactly how this happened for this example. During iterations 2 and 3, no parameters are added for interpolation, while during iterations 4 and 5, no frequencies are added for interpolation. Upon convergence, for this simple example where the underlying function is a low-order rational function itself, p-AAA exactly recovers it. In other words, after step 7, all the data is interpolated. This shows another flexibility of p-AAA: If the underlying order is

low enough, the LS component is automatically converted to full interpolation, thus, in this special example, giving the same approximant as the parametric Loewner approach [80].

We present in Figure 2.4 the evolution of the p-AAA approximant at various iterations: first, third, and last (seventh). As Figure 2.4 shows, upon convergence, the proposed algorithm captures the full model exactly.

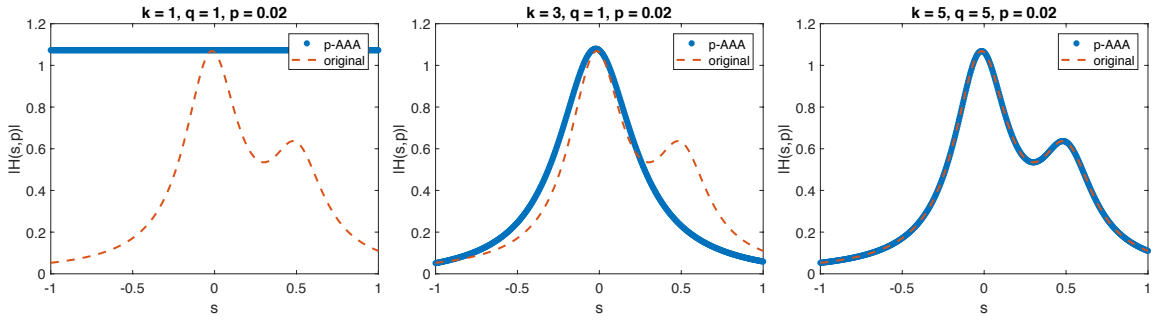


Figure 2.4: Example 2.2.2.1: p-AAA approximation at various iterations

2.2.2.2 A beam model with a string attached

In this example, we consider the finite element model of a one-dimensional Euler-Bernoulli beam with a string attached near its left boundary and an input force applied at its right boundary, as shown in Figure 2.5. As the output $y(t)$, we measure the displacement at the right boundary where the forcing is applied. We take the stiffness coefficient of the spring as the parameter and obtain the parametric dynamical system

$$\mathbf{M}\ddot{\mathbf{x}}(t,p) + \mathbf{G}\dot{\mathbf{x}}(t,p) + \mathbf{K}(p)\mathbf{x}(t,p) = \mathbf{b}u(t), \quad y(t,p) = \mathbf{c}^\top \mathbf{x}(t,p),$$

with the corresponding transfer function

$$H(s,p) = \mathbf{c}^\top (s^2\mathbf{M} + s\mathbf{G} + \mathbf{K}(p))^{-1}\mathbf{b},$$

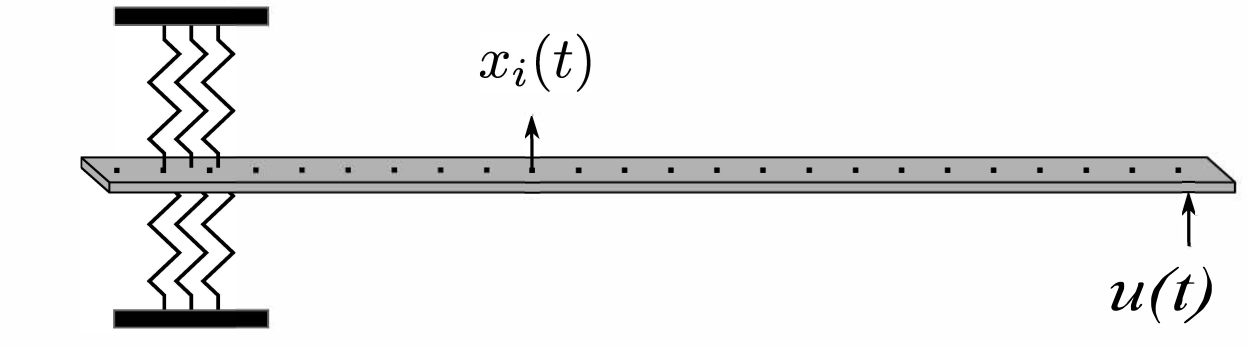


Figure 2.5: Example 2.2.2.2

where $\mathbf{M}$ and $\mathbf{G}$ are, respectively, the mass and damping matrices; $\mathbf{K}(p)$ is the parametric stiffness matrix; and $\mathbf{b}$ and $\mathbf{c}$ are, respectively, the input-to-state and the state-to-output mappings. We measure the transfer function at $H(s_i, p_j)$ for $N = 3000$ frequency points $\{s_i\}$ in the interval $[0, 2\pi \times 10^3]\imath$, where $\imath^2 = -1$ and for $M = 3$ parameter values $p_1 = 0.2$, $p_2 = 0.4$, and $p_3 = 1$. p-AAA yields an approximant with orders $(k, q) = (12, 2)$. Out of three parameter samples, p-AAA chooses $p_1 = 0.2$ and $p_2 = 0.4$ for interpolation. Using the same parameter and frequency samples, we also construct the parametric Loewner approximant [80]. Figure 2.6 shows the amplitude frequency responses of the original transfer function $H(s, p)$, and the p-AAA and parametric Loewner approximants for various parameter values, including values that did not enter into the p-AAA or parametric Loewner construction (second row in Figure 2.6). Both p-AAA and parametric Loewner yield highly accurate approximations, capturing the peaks in the frequency response accurately. We note that $p = 0$ and $p = 15$ are outside the parameter range that was sampled. To check the accuracy of the p-AAA and parametric Loewner approximants further, we perform an exhaustive search over the parameter domain by computing, for every $\hat{p} \in [0, 1]$, the worst-case frequency domain error, i.e., $\max\limits_{\omega \in [0, 2\pi \times 10^3]} \left| H(\imath\omega, \hat{p}) - \widetilde{H}(\imath\omega, \hat{p}) \right|$. The results in Figure 2.7 show that p-AAA is accurate throughout the full parameter domain and, for this example, outperforms the parametric Loewner approach.

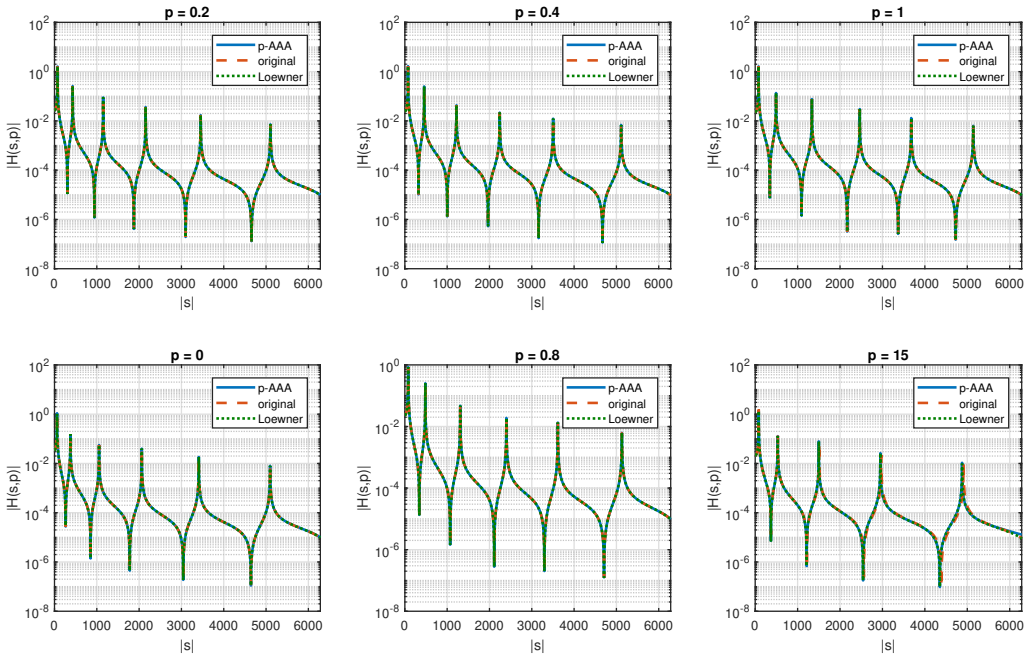


Figure 2.6: Example 2.2.2.2. p-AAA approximation for various parameter values and Loewner approximation with the same order as p-AAA.

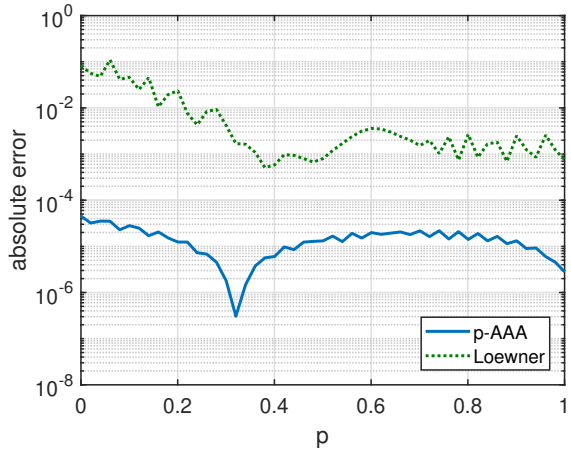
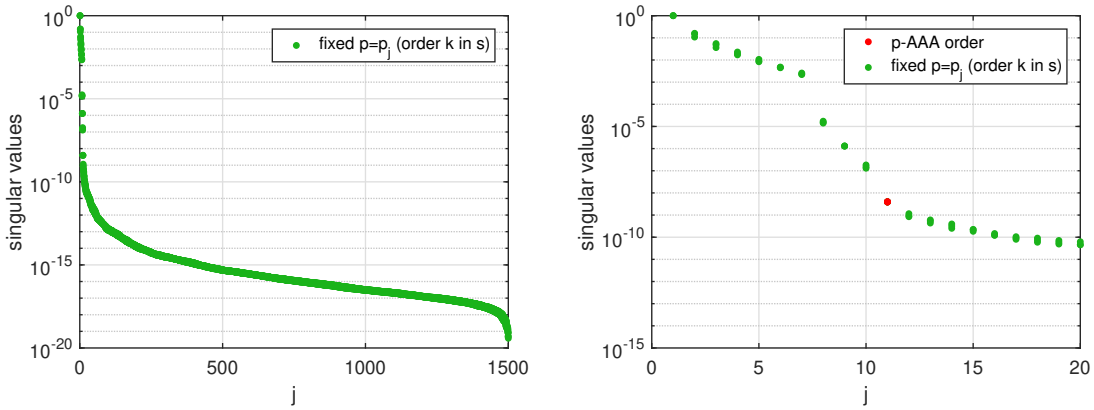


Figure 2.7: Example 2.2.2.2. Absolute infinite error in the s-interval sampled.



(a) Three decays (one for each fixed p). (b) First 20 values in Figure 2.8a.

Figure 2.8: Example 2.2.2.2: Singular value decay for the parametric Loewner method.

As a reference for the reader, we include in Figure 2.8 the first step of the Loewner method: using the decay of the singular values of the 1D Loewner matrices to pick the size of the Loewner partition. Figure 2.8a shows the singular value decay of the 1D Loewner matrices corresponding to the data when fixing each of the three parameter values sampled. Figure 2.8b shows the first 20 singular values in Figure 2.8a and highlights the order $k-1$ in s that our p-AAA algorithm selected automatically (without looking at these singular values). Note that we do not show the corresponding Loewner analysis to determine the order q in p. This is because there is not really a choice to be made in this particular example:

- The minimum order to have a rational approximation that depends on p is $q-1=1$. This is, $q \geq 2$.

- Since we only have $M=3$ parameter samples and q must be less than M in order to have a partition, then $q<3$.

Putting the two together only leaves one option: $q=2$. Note that this is indeed the order that the p-AAA algorithm picked automatically.

2.2.2.3 Penzl model

Consider the linear dynamical system

$$\dot{\mathbf{x}}(t,p) = \mathbf{A}(p)\mathbf{x}(t,p) + \mathbf{b}f(t), \quad y(t,p) = \mathbf{c}^\top \mathbf{x}(t,p),$$

where $\mathbf{b} = \mathbf{c} = [\ \overbrace{10\cdots10}^{6}\overbrace{1\cdots1}^{1000}\ ]^\top$ and

$$\mathbf{A}(p) = \mathsf{diag}\left(\mathbf{A}_1(p), \mathbf{A}_2, \mathbf{A}_3, \mathbf{A}_4\right)$$

with

$$\mathbf{A}_1(p) = \begin{bmatrix} -1 & p \\ -p & -1 \end{bmatrix},\ \mathbf{A}_2 = \begin{bmatrix} -1 & 200 \\ -200 & -1 \end{bmatrix},\ \mathbf{A}_3 = \begin{bmatrix} -1 & 400 \\ -400 & -1 \end{bmatrix},$$

and

$$\mathbf{A}_4 = -\mathsf{diag}(1, 2, \ldots, 1000).$$

The transfer function corresponding to this model is

$$H(s,p) = \mathbf{c}^\top \left(s\mathbf{I} - \mathbf{A}(p)\right)^{-1} \mathbf{b},$$

where the variable p affects the location of the transfer function's left peak in the frequency domain. The model, taken from [80], is a modification of the nonparametric Penzl model [112].

We sample this transfer function at $H(s_i, p_j)$ for $N = 100$ frequency points $\{s_i\}$ logarithmically spaced in $[0.1, 1000]\imath$ and $M = 30$ parameter points $\{p_j\}$ linearly spaced in $[10, 100]$, and run p-AAA with a stopping tolerance of 10^{-3}. After 9 iterations, we obtain the p-AAA approximation with $(k, q) = (9, 9)$; thus out of 3000 sampling pairs (s_i, p_j), p-AAA chooses to

enforce interpolation in 81 pairs and an LS fit in the rest. We show in Figure 2.9 the approximation quality for four representative parameter values: $p = 5$, $p = 10$, $p = 11.5$, and $p = 110$. Note that two of these parameter points are outside the sampled interval and, as in the previous example, p-AAA provides a high-fidelity approximation even at those parameter values.

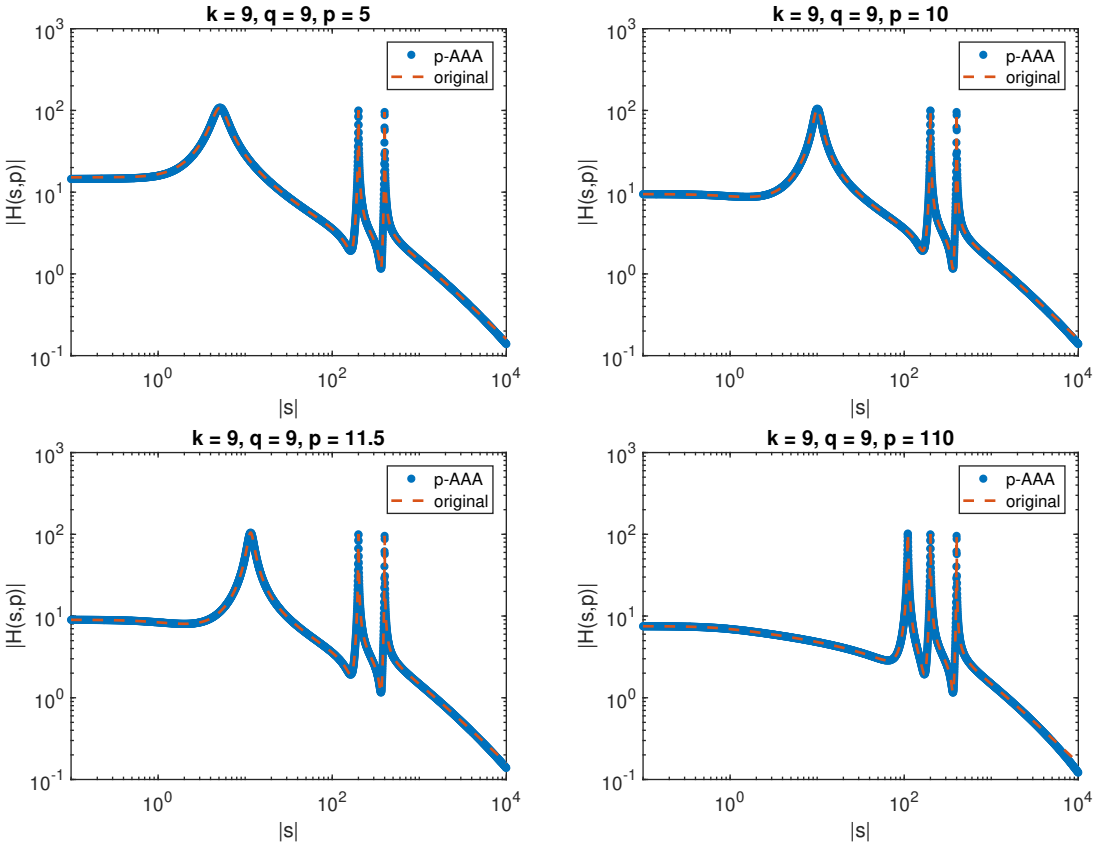


Figure 2.9: Example 2.2.2.3: p-AAA approximation for various parameter values

2.2.3 Variable scaling in p-AAA

In this section we discuss how variable scaling may affect our method. Depending on the application, dynamical systems representing the same phenomenon may be expressed in different ways by, for example, choosing different units of measurement. Ideally, this choice

should not affect the accuracy of the models used to approximate that system. In that case, the sampling of such systems could be made unit-independent, and a global sampling strategy could be applied to all cases. We will show that this is not the case for the p-AAA algorithm, and one should take advantage of the *intuitive* knowledge of the variable spaces to choose *good* samples.

Consider a scaling in the frequencies and parameters

$$s^{new} = f(s) \qquad \text{and} \qquad p^{new} = g(p).$$

The corresponding system $H^{new}(s^{new}, p^{new})$ models the same phenomenon, hence function values do not depend on scaling, i.e.,

$$H^{new}(s^{new}, p^{new}) = H(s, p) \qquad \text{for } (s^{new}, p^{new}) = (f(s), g(p)). \tag{2.23}$$

Therefore the choice $\beta_{ij} = H(\sigma_i, \pi_j)\alpha_{ij}$ for the interpolation condition on our p-AAA algorithm is not affected by scaling. However, scaling may affect the Loewner matrix $\mathbb{L}_2$ that defines the LS problem in p-AAA. The nonzero entries of $\mathbb{L}_2$ have one of the following forms:

$$\frac{H(\sigma, \hat{\pi}) - H(\sigma, \pi)}{\hat{\pi} - \pi}, \qquad \frac{H(\hat{\sigma}, \pi) - H(\sigma, \pi)}{\hat{\sigma} - \sigma}, \qquad \text{and} \qquad \frac{H(\hat{\sigma}, \hat{\pi}) - H(\sigma, \pi)}{(\hat{\sigma} - \sigma)(\hat{\pi} - \pi)}.$$

Note that the numerators remain constant under scaling due to (2.23). If scaling preserves distances, then

$$\hat{\sigma}^{new} - \sigma^{new} = f(\hat{\sigma}) - f(\sigma) = \hat{\sigma} - \sigma,$$
$$\hat{\pi}^{new} - \pi^{new} = g(\hat{\pi}) - g(\pi) = \hat{\pi} - \pi,$$

and so $\mathbb{L}_2$ is unaltered. In general, if scaling does not preserve distances, each $\mathbb{L}_2$ entry can

be rewritten as

$$\frac{H^{new}(\sigma^{new},\hat{\pi}^{new})-H^{new}(\sigma^{new},\pi^{new})}{\hat{\pi}^{new}-\pi^{new}} = \frac{H(\sigma,\hat{\pi})-H(\sigma,\pi)}{\hat{\pi}-\pi}\cdot\frac{\hat{\pi}-\pi}{g(\hat{\pi})-g(\pi)},$$

$$\frac{H^{new}(\hat{\sigma}^{new},\pi^{new})-H^{new}(\sigma^{new},\pi^{new})}{\hat{\sigma}^{new}-\sigma^{new}} = \frac{H(\hat{\sigma},\pi)-H(\sigma,\pi)}{\hat{\sigma}-\sigma}\cdot\frac{\hat{\sigma}-\sigma}{f(\hat{\sigma})-f(\sigma)}, \text{ and}$$

$$\frac{H^{new}(\hat{\sigma}^{new},\hat{\pi}^{new})-H^{new}(\sigma^{new},\pi^{new})}{(\hat{\sigma}^{new}-\sigma^{new})(\hat{\pi}^{new}-\pi^{new})} = \frac{H(\hat{\sigma},\hat{\pi})-H(\sigma,\pi)}{(\hat{\sigma}-\sigma)(\hat{\pi}-\pi)}\cdot\frac{(\hat{\sigma}-\sigma)(\hat{\pi}-\pi)}{(f(\hat{\sigma})-f(\sigma))(g(\hat{\pi})-g(\pi))}.$$

With compact notation, this means

$$\mathbb{L}_2^{new} = \begin{bmatrix} \mathbb{L}^{new}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}} \\ \mathbb{L}^{new}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}} \\ \mathbb{L}^{new}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}} \end{bmatrix} = \begin{bmatrix} (\mathbf{I}_k \otimes G)\odot\mathbb{L}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}} \\ [\mathbf{I}_q\otimes F\mathbf{e}_1 \mid \cdots \mid \mathbf{I}_q\otimes F\mathbf{e}_k]\odot\mathbb{L}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}} \\ \left(F\otimes\mathbf{1}_{(M-q)\times q}\right)\odot\left(\mathbf{1}_{(N-k)\times k}\otimes G\right)\odot\mathbb{L}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}} \end{bmatrix},$$

where $\otimes$ denotes the Kronecker product, $\odot$ denotes the Hadamard product, $\mathbf{1}_{m\times n}$ denotes the matrix of dimension $m\times n$ with all entries equal to one, and the matrices F and G contain all the weights corresponding to the variables scaling, $f(s)$ and $g(p)$:

$$F = \begin{bmatrix} \frac{\hat{\sigma}_1-\sigma_1}{f(\hat{\sigma}_1)-f(\sigma_1)} & \cdots & \frac{\hat{\sigma}_1-\sigma_k}{f(\hat{\sigma}_1)-f(\sigma_k)} \\ \vdots & \ddots & \vdots \\ \frac{\hat{\sigma}_{N-k}-\sigma_1}{f(\hat{\sigma}_{N-k})-f(\sigma_1)} & \cdots & \frac{\hat{\sigma}_{N-k}-\sigma_k}{f(\hat{\sigma}_{N-k})-f(\sigma_k)} \end{bmatrix}\in\mathbb{C}^{(N-k)\times k} \qquad \text{and}$$

$$G = \begin{bmatrix} \frac{\hat{\pi}_1-\pi_1}{g(\hat{\pi}_1)-g(\pi_1)} & \cdots & \frac{\hat{\pi}_1-\pi_q}{g(\hat{\pi}_1)-g(\pi_q)} \\ \vdots & \ddots & \vdots \\ \frac{\hat{\pi}_{M-q}-\pi_1}{g(\hat{\pi}_{M-q})-g(\pi_1)} & \cdots & \frac{\hat{\pi}_{M-q}-\pi_q}{g(\hat{\pi}_{M-q})-g(\pi_q)} \end{bmatrix}\in\mathbb{C}^{(M-q)\times q}.$$

Note that these matrices are 1D Loewner matrices for the inverse scaling, i.e., for $f^{-1}(s)$ and $g^{-1}(p)$. To summarize, we present the following lemma.

Lemma 2.5. *Consider the partitioned data (2.13)*

$$\{s_1,\dots,s_N\} = \{\sigma_1,\dots,\sigma_k\} \cup \{\hat{\sigma}_1,\dots,\hat{\sigma}_{N-k}\} \stackrel{def}{=\!=} \boldsymbol{\sigma} \ \cup \ \hat{\boldsymbol{\sigma}},$$
$$\{p_1,\dots,p_M\} = \{\pi_1,\dots,\pi_q\} \cup \{\hat{\pi}_1,\dots,\hat{\pi}_{M-q}\} \stackrel{def}{=\!=} \boldsymbol{\pi} \ \cup \ \hat{\boldsymbol{\pi}}, \ \textit{and}$$
$$\left[\begin{array}{c|c} [H(\sigma_i,\pi_j)] & [H(\sigma_i,\hat{\pi}_j)] \\ \hline [H(\hat{\sigma}_i,\pi_j)] & [H(\hat{\sigma}_i,\hat{\pi}_j)] \end{array}\right] \stackrel{def}{=\!=} \left[\begin{array}{c|c} \mathbf{D}_{\boldsymbol{\sigma\pi}} & \mathbf{D}_{\boldsymbol{\sigma\hat{\pi}}} \\ \hline \mathbf{D}_{\boldsymbol{\hat{\sigma}\pi}} & \mathbf{D}_{\boldsymbol{\hat{\sigma}\hat{\pi}}} \end{array}\right],$$

and the Loewner matrix $\mathbb{L}_2$ *that defines the* LS *problem (2.20). Let* $f(s)$ *and* $g(p)$ *be two invertible scalings of the frequency and parameter variables, respectively. Let* $\sigma_i^{new} = f(\sigma_i)$, $\hat{\sigma}_{\hat{\imath}}^{new} = f(\hat{\sigma}_{\hat{\imath}})$, $\pi_j^{new} = g(\pi_j)$, *and* $\hat{\pi}_{\hat{\jmath}}^{new} = g(\hat{\pi}_{\hat{\jmath}})$ *for any* i, $\hat{\imath}$, j, *and* $\hat{\jmath}$. *Then, for the scaled data*

$$\{\sigma_1^{new},\dots,\sigma_k^{new}\} \cup \{\hat{\sigma}_1^{new},\dots,\hat{\sigma}_{N-k}^{new}\}, \ \textit{and}$$
$$\{\pi_1^{new},\dots,\pi_q^{new}\} \cup \{\hat{\pi}_1^{new},\dots,\hat{\pi}_{M-q}^{new}\},$$

the Loewner matrix $\mathbb{L}_2^{new}$ *for the corresponding* LS *problem can be written as*

$$\mathbb{L}_2^{new} = \begin{bmatrix} \mathbf{I}_k \otimes G \\ [\mathbf{I}_q \otimes F\mathbf{e}_1 \ | \ \cdots \ | \ \mathbf{I}_q \otimes F\mathbf{e}_k] \\ \left(F \otimes \mathbf{1}_{(M-q)\times q}\right) \odot \left(\mathbf{1}_{(N-k)\times k} \otimes G\right) \end{bmatrix} \odot \mathbb{L}_2. \tag{2.24}$$

In general, the singular vectors of $\mathbb{L}_2^{new}$ are not the same as those of $\mathbb{L}_2$, hence the p-AAA approximation for the scaled problem may differ from the p-AAA approximation for the original problem. We illustrate this for a simple example with linear scaling.

2.2.3.1 Linear scaling

Consider the case where variables are linearly scaled. Then

$$s^{new} = f(s) = as + b \implies \hat{\sigma}^{new} - \sigma^{new} = f(\hat{\sigma}) - f(\sigma) = a\hat{\sigma} + b - (a\sigma + b) = a(\hat{\sigma} - \sigma),$$
$$p^{new} = g(p) = cp + d \implies \hat{\pi}^{new} - \pi^{new} = g(\hat{\pi}) - g(\pi) = c\hat{\pi} + d - (c\pi + d) = c(\hat{\pi} - \pi),$$

and thus the entries in the matrices F and G are constant:

$$\frac{\hat{\sigma} - \sigma}{f(\hat{\sigma}) - f(\sigma)} = \frac{\hat{\sigma} - \sigma}{a(\hat{\sigma} - \sigma)} = \frac{1}{a} \quad \text{and}$$
$$\frac{\hat{\pi} - \pi}{g(\hat{\pi}) - g(\pi)} = \frac{\hat{\pi} - \pi}{c(\hat{\pi} - \pi)} = \frac{1}{c}.$$

Therefore the LS problem for the scaled model can be written as

$$\min_{\|\mathbf{a}^{new}\|_2 = 1} \|\mathbb{L}_2^{new}\mathbf{a}^{new}\|_2 ,$$

where

$$\mathbb{L}_2^{new} = \begin{bmatrix} \frac{1}{c}\,\mathbb{L}_{\boldsymbol{\sigma\hat{\pi}}} \\ \frac{1}{a}\,\mathbb{L}_{\boldsymbol{\hat{\sigma}\pi}} \\ \frac{1}{ac}\mathbb{L}_{\boldsymbol{\hat{\sigma}\hat{\pi}}} \end{bmatrix} = \begin{bmatrix} \frac{1}{c}\,\mathbf{1}_{k(M-q)\times(kq)} \\ \frac{1}{a}\,\mathbf{1}_{q(N-k)\times(kq)} \\ \frac{1}{ac}\mathbf{1}_{(N-k)(M-q)\times(kq)} \end{bmatrix} \odot \mathbb{L}_2.$$

In order to illustrate our analysis, we implement three p-AAA approximations of the Penzl model from Section 2.2.2.3 with $N = 50$ frequency samples in $s_i \in [0.1, 1000]\iota$ and $M = 10$ parameter samples in $p_j \in [10, 100]$, both linearly spaced. We scale each variable so that

both sets of samples lie in the interval $[0, 1]$:

$$s_i^{new} = f(s_i) = \frac{1}{999.9}(s_i - 0.1) \qquad \text{and} \qquad p_j^{new} = g(p_j) = \frac{1}{90}(p_j - 10).$$

The three approximations are built as follows:

(a) (thick dashed orange line in Figure 2.10) p-AAA with samples (s_i, p_j), which yields an approximation $\widetilde{H}(s, p)$.

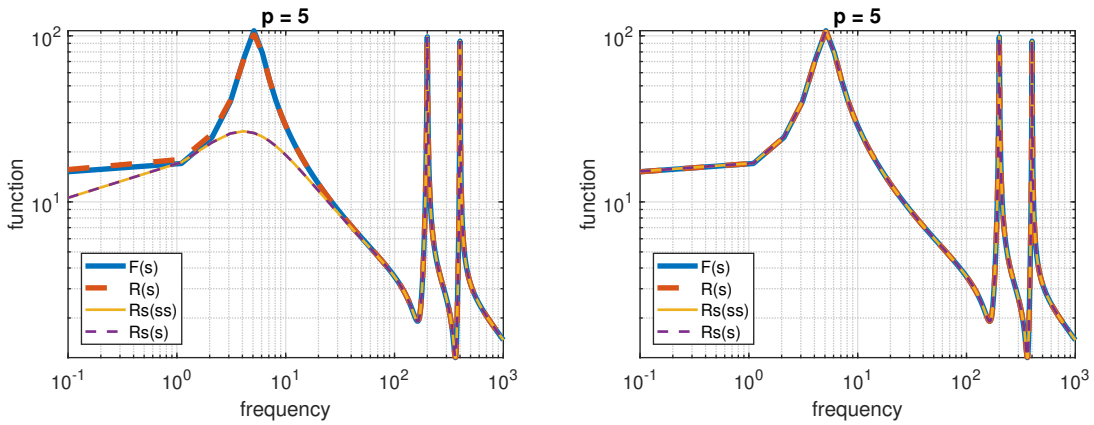
(b) (dashed yellow line in Figure 2.10) p-AAA with samples (s_i^{new}, p_j^{new}), which yields an approximation $\widetilde{H}^{new}(s^{new}, p^{new})$.

(c) (dotted purple line in Figure 2.10) p-AAA with samples (s_i, p_j) and $\mathbb{L}_2^{new}$, which yields an approximation $\widetilde{H}^{new\mathbb{L}}(s, p)$.

Note that the function values are the same for all three cases (a)-(c): $h_{ij} = H(s_i, p_j)$. Also note that even though (b) and (c) may seem like different cases, they are not. They yield the same approximation, i.e.,

$$\widetilde{H}^{new}(s^{new}, p^{new}) = \widetilde{H}^{new\mathbb{L}}(s, p), \qquad \text{for } (s^{new}, p^{new}) = (f(s), g(p)).$$

The difference is that in (b) we incorporate the scaling in the sample values directly hence we are able to run our p-AAA algorithm unaltered, while (c) works with the original samples and needs the algorithm to incorporate the scaling into the Loewner matrices. We see that the approximations are indeed the same in Figure 2.10a. We also see that the scaling does affect the approximation: the scaled approximation misses the first peak of the transfer function and selects different orders: while for the non-scaled data $(k, q) = (8, 7)$, for the scaled data we obtain $(k, q) = (7, 9)$. We also observe that considering more samples $N = 100$ and



(a) $N = 50$ and $M = 10$. (b) $N = 100$ and $M = 30$.

Figure 2.10: Comparing p-AAA approximations for the (scaled) Penzl model.

$M = 30$ results in a match for the scaled data as well, as shown in Figure 2.10b, where the orders are $(k, q) = (9, 6)$ and $(k, q) = (9, 25)$ for the non-scaled data and for the scaled data, respectively.

2.2.3.2 1D case

Our analysis follows analogously for AAA. We include our summary for the one-variable algorithm since notation is simpler and we are able to draw further conclusions in the Loewner case. The choice for interpolation remains invariant

$$\beta_i = H^{new}(\sigma_i^{new}) = H^{new}(f(\sigma_i))\alpha_i = H(\sigma_i)\alpha_i,$$

while the LS problem is scaled as follows:

$$\mathbb{L}_{ij}^{new} = \frac{H^{new}(\hat{\sigma}_i^{new}) - H^{new}(\sigma_j^{new})}{\hat{\sigma}_i^{new} - \sigma_j^{new}} = \frac{H(\hat{\sigma}_i) - H(\sigma_j)}{\hat{\sigma}_i - \sigma_j} \frac{\hat{\sigma}_i - \sigma_j}{f(\hat{\sigma}_i) - f(\sigma_j)},$$

$$\implies \mathbb{L}^{new} = F \odot \mathbb{L}.$$

In particular, for linear scaling, we have $\mathbb{L}^{new} = \frac{1}{a}\mathbb{L}$. Therefore in the Loewner framework, where the data partition is chosen so that $\mathbb{L}$ has a non-trivial null space, a linear scaling does not affect the approximation:

$$\mathbb{L}^{new}\mathbf{a}^{new} = \mathbf{0} \implies \frac{1}{a}\mathbb{L}\mathbf{a}^{new} = \mathbf{0} \implies \mathbb{L}\mathbf{a}^{new} = \mathbf{0} \implies \mathbf{a}^{new} = \mathbf{a},$$

assuming a unique solution.

2.2.4 p-AAA for more than two parameters

The p-AAA algorithm extends analogously to the cases with more than two variables. To keep the discussion concise, we briefly highlight the three-variable case.

In this case, the underlying (transfer) function to approximate, $H(s, p, z)$, is a function of three variables, s, p, and z, where z is another parameter in the model, and we assume access to the sampling data

$$h_{ij\ell} = H(s_i, p_j, z_\ell) \in \mathbb{C} \quad \text{for} \quad i = 1, \ldots, N, \quad j = 1, \ldots, M, \quad \text{and} \quad \ell = 1, \ldots, O. \tag{2.25}$$

The approximant $\widetilde{H}(s, p, z)$ is represented in the barycentric form given by

$$\widetilde{H}(s, p, z) = \sum_{i=1}^{k}\sum_{j=1}^{q}\sum_{\ell=1}^{o} \frac{\beta_{ij\ell}}{(s-\sigma_i)(p-\pi_j)(z-\zeta_\ell)} \Big/ \sum_{i=1}^{k}\sum_{j=1}^{q}\sum_{\ell=1}^{o} \frac{\alpha_{ij\ell}}{(s-\sigma_i)(p-\pi_j)(z-\zeta_\ell)}, \tag{2.26}$$

where $\{\sigma_i\}$, $\{\pi_j\}$, and $\{\zeta_\ell\}$ are to-be-determined sampling points, subsets of $\{s_i\}$, $\{p_j\}$, and $\{z_\ell\}$, respectively. As in the two-variable case, $\beta_{ij\ell}$ will be chosen to enforce interpolation in a subset of the data and $\alpha_{ij\ell}$ to minimize a linearized LS error in the remaining data.

In accordance with the data (2.25) and the approximant $\widetilde{H}(s, p, z)$, partition the sampling

points:

$$\begin{aligned}[s_1,\dots,s_N] &= [\sigma_1,\dots,\sigma_k]\cup[\hat{\sigma}_1,\dots,\hat{\sigma}_{N-k}] = [\boldsymbol{\sigma}\mid\hat{\boldsymbol{\sigma}}],\\ [p_1,\dots,p_M] &= [\pi_1,\dots,\pi_q]\cup[\hat{\pi}_1,\dots,\hat{\pi}_{M-q}] = [\boldsymbol{\pi}\mid\hat{\boldsymbol{\pi}}],\ \text{and}\\ [z_1,\dots,z_O] &= [\zeta_1,\dots,\zeta_o]\ \cup\ [\hat{\zeta}_1,\dots,\hat{\zeta}_{O-o}] = [\boldsymbol{\zeta}\mid\hat{\boldsymbol{\zeta}}].\end{aligned} \tag{2.27}$$

Then, p-AAA imposes interpolation on the samples $\{\boldsymbol{\sigma},\boldsymbol{\pi},\boldsymbol{\zeta}\}$ by setting

$$\beta_{ij\ell} = H(\sigma_i,\pi_j,\zeta_\ell)\alpha_{ij\ell},\quad \text{for } i=1,\dots,k,\ j=1,\dots,q,\ \text{and},\ \ell=1,\dots,o. \tag{2.28}$$

Based on the partitioning (2.27), consider the data as a three-dimensional tensor. Figure 2.11 shows a visualization of this tensor paired with the separation of the data into the part that will be interpolated and the part that LS will be applied to. We enforce interpolation in

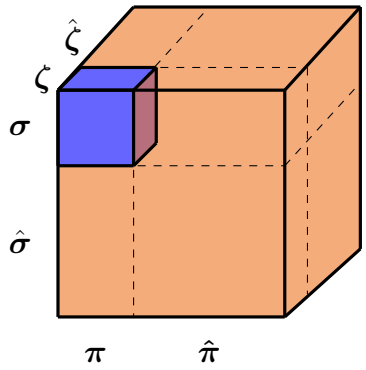


Figure 2.11: Illustration of the p-AAA data partition tensor for $\mathbf{H}(s_i,p_j,z_k)$ corresponding to (2.27). In blue, data to interpolate. In orange, data to perform LS minimization.

the $(1,1,1)$ block of this tensor (blue block in Figure 2.11) with the choice in (2.28). Then, p-AAA minimizes the linearized LS error in the rest of the data (orange blocks in Figure 2.11) by choosing the remaining coefficients

$$\mathbf{a} = [\alpha_{111}\cdots\alpha_{11o}|\alpha_{121}\cdots\alpha_{12o}|\cdots|\alpha_{kq1}\cdots\alpha_{kqo}]^\top.$$

These coefficients should solve the linear LS problem

$$\min_{\|\mathbf{a}\|_2=1} \|\mathbb{L}_3\mathbf{a}\|_2,$$

where $\mathbb{L}_3$ is the 3D Loewner matrix, which plays the same role the 2D Loewner matrix $\mathbb{L}_2$ played in Section 2.2.1. The partitioning of the data in (2.27) is automatically established via the greedy search in every step.

Regarding variable scaling, we will only mention that $\mathbb{L}_3^{new}$ would have an analogous expression to $\mathbb{L}_2^{new}$ in (2.24), with the scaling of each variable affecting the corresponding blocks in $\mathbb{L}_3$.

Generalization to functions of more than three variables follows analogously. We skip those details due to cumbersome notation. However the potential computational difficulties with the increasing number of variables is worth elaborating. Assume that at the current step of p-AAA, we have the approximant $\widetilde{H}(s,p,z)$ as in (2.26). Given the sampling data in (2.25), this will result in $\mathbb{L}_3$ having $NMO - kqo$ rows and kqo columns. Therefore computing the coefficient vector $\mathbf{a}$ becomes more expensive as the number of variables (and the orders in each variable) increase. For functions with many variables, if the coefficient matrix becomes prohibitively large to compute $\mathbf{a}$ via direct methods, one might revert to well-established iterative approaches. For the numerical examples we considered in this chapter, these computational complications did not arise and direct methods were suitable.

2.2.4.1 A three-variable example

We reconsider the parametric system in Section 2.2.2.3 and include one more parameter in defining $\mathbf{A}_2$ and $\mathbf{A}_3$:

$$\mathbf{A}_2(z) = \begin{bmatrix} -1 & z \\ -z & -1 \end{bmatrix} \quad \text{and} \quad \mathbf{A}_3(z) = \begin{bmatrix} -1 & 2z \\ -2z & -1 \end{bmatrix}.$$

This leads to the three-variable (transfer) function $H(s,p,z) = \mathbf{c}^\top(sI - \mathbf{A}(p,z))^{-1}\mathbf{b}$ to approximate. With the addition of the new variable, now all three peaks in the frequency response move as p and z vary, making it a harder system to approximate. We sample the transfer function at $N = 100$ logarithmically spaced frequencies $\{s_i\}$ in $[1, 2000]\imath$, and $M = O = 10$ linearly spaced parameter samples $\{p_j\}$ in $[10, 100]$ and $\{z_\ell\}$ in $[150, 250]$. Applying the three-variable p-AAA with relative error tolerance 10^{-3} yields a rational approximation with orders $(k, q, o) = (12, 2, 4)$. We plot $H(s,p,z)$ and $\widetilde{H}(s,p,z)$ for two representative p and z samples in Figure 2.12. Note that all the peaks in the frequency response plot have moved and $\widetilde{H}(s,p,z)$ accurately captures this behavior. To further show the approximation quality of p-AAA, we perform a search for the worst-case relative error over the full frequency interval and parameter domain of interest and obtain the worst case error of 3.8×10^{-3}, illustrating the success of p-AAA for this three-variable example.

2.3 p-AAA for Matrix-valued Functions

So far, we have considered approximating scalar-valued functions $H(s,p)$. In this section, we discuss p-AAA for approximating matrix-valued functions. This is a common situation, especially arising in the case of dynamical systems where the underlying system has multiple

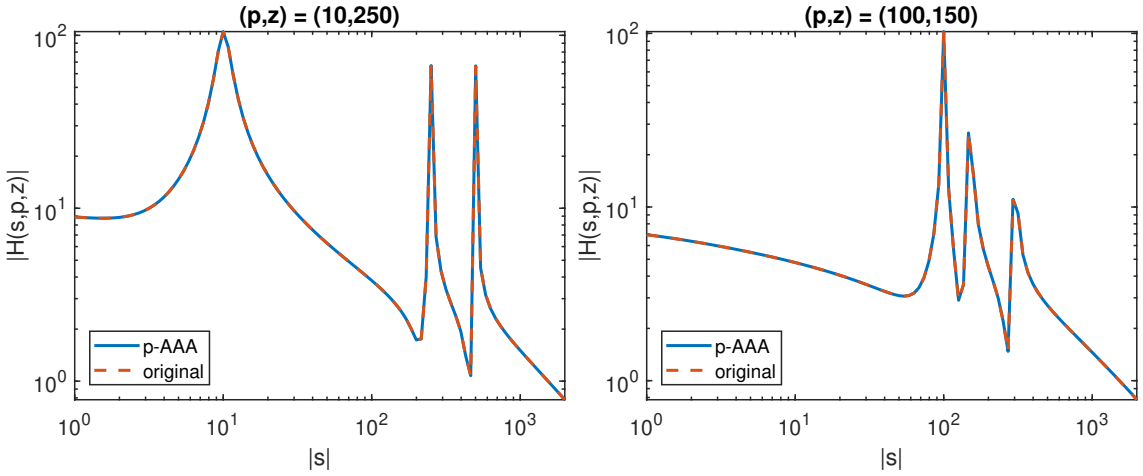


Figure 2.12: p-AAA for 2 parameters (and 1 frequency)

inputs and multiple outputs (MIMO), leading to matrix-valued transfer functions. Motivated by our interest in approximating dynamical systems, we will call the resulting method MIMO p-AAA. To keep the notation concise, we will present the discussion for the two-variable case. But as in Section 2.2.4, the results similarly extend to higher-dimensional parametric problems.

Let $\mathbf{H}(s,p)$ denote the underlying MIMO (transfer) function with n_{in} inputs and n_{out} outputs. Therefore, for the sampling points $\{s_i\}_{i=1}^N$ and $\{p_j\}_{j=1}^M$, we have access to matrix-valued sampling data:

$$\mathbf{H}_{ij} = \mathbf{H}(s_i, p_j) \in \mathbb{C}^{n_{in} \times n_{out}} \quad \text{for } i = 1, \ldots, N \quad \text{and} \quad j = 1, \ldots, M. \tag{2.29}$$

From the data (2.29), the goal is to construct a high-fidelity, matrix-valued approximant $\widetilde{\mathbf{H}}(s,p)$ to $\mathbf{H}(s,p)$. The method we describe below consists of three main steps: transform the data from matrix to scalar values; apply p-AAA; and transform the p-AAA approximation from scalar- to matrix-valued. However recent work in the nonparametric case has been done to tackle this MIMO problem by using matrix-valued weights in the barycentric

representation; see block-AAA [62] for details. This could potentially lead to a different implementation of MIMO p-AAA.

2.3.1 Transformation to scalar-valued data

For the single-variable (nonparametric) case, one solution to handle the matrix-valued data in AAA is to vectorize every sample and replace the scalar data forming the Loewner matrix $\mathbb{L}$ with the vectorized data. This is closely related to the approach proposed in Lietaert *et al.* [92] for using AAA in nonlinear eigenvalue problems. It is also analogous to how VF handles MIMO problems. One potential disadvantage of this approach is that, in the case of many inputs and outputs, the resulting Loewner matrix will have large dimensions, leading to a computationally expensive LS step. Exploiting the fact that only certain rows and columns of the underlying Loewner matrix change in every step, [92] partially alleviates this computational complexity. However, for the parametric problems we consider here, dimension growth due to vectorization is more prominent and we will adopt another approach introduced by [52] for the nonparametric case, which transforms the MIMO data to a scalar one and applies AAA to this scalar-valued data. We will extend this approach to parametric problems and establish what it means, for MIMO p-AAA, in terms of interpolation and LS minimization.

As in the scalar case, assume the partitioning of the data in (2.29) as follows:

$$\begin{aligned}
\{s_1,\dots,s_N\} &= \{\sigma_1,\dots,\sigma_k\} \cup \{\hat{\sigma}_1,\dots,\hat{\sigma}_{N-k}\} \stackrel{\text{def}}{=\!=} \boldsymbol{\sigma} \ \cup \ \hat{\boldsymbol{\sigma}}, \\
\{p_1,\dots,p_M\} &= \{\pi_1,\dots,\pi_q\} \cup \{\hat{\pi}_1,\dots,\hat{\pi}_{M-q}\} \stackrel{\text{def}}{=\!=} \boldsymbol{\pi} \ \cup \ \hat{\boldsymbol{\pi}}, \text{ and} \\
&\left[\begin{array}{c|c} [\mathbf{H}(\sigma_i,\pi_j)] & [\mathbf{H}(\sigma_i,\hat{\pi}_j)] \\ \hline [\mathbf{H}(\hat{\sigma}_i,\pi_j)] & [\mathbf{H}(\hat{\sigma}_i,\hat{\pi}_j)] \end{array}\right] \stackrel{\text{def}}{=\!=} \left[\begin{array}{c|c} \mathbf{D}_{\boldsymbol{\sigma}\boldsymbol{\pi}} & \mathbf{D}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}} \\ \hline \mathbf{D}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}} & \mathbf{D}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}} \end{array}\right].
\end{aligned} \tag{2.30}$$

This partitioning will be determined by applying p-AAA to a scalar data set described below. In accordance with this partitioning, we want to construct $\widetilde{\mathbf{H}}(s,p)$ with the matrix-valued barycentric form

$$\widetilde{\mathbf{H}}(s,p) = \frac{\mathbf{N}(s,p)}{d(s,p)} = \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\mathbf{B}_{ij}}{(s-\sigma_i)(p-\pi_j)} \Big/ \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\tilde{\alpha}_{ij}}{(s-\sigma_i)(p-\pi_j)}, \tag{2.31}$$

where $\mathbf{B}_{ij} \in \mathbb{C}^{n_{in}\times n_{out}}$ and $\tilde{\alpha}_{ij} \in \mathbb{C}$ are to be determined.

Motivated by [52] for the nonparametric case, we convert the matrix-valued data (2.29) to scalar data by picking two random unit vectors $\mathbf{w} \in \mathbb{C}^{n_{out}}$ and $\mathbf{v} \in \mathbb{C}^{n_{in}}$, and computing

$$h_{ij} = \mathbf{w}^{\top}\mathbf{H}(s_i,p_j)\mathbf{v} \quad \text{for} \quad i = 1,\dots,N \quad \text{and} \quad j = 1,\dots,M. \tag{2.32}$$

We apply p-AAA to the scalar data (2.32) to obtain the scalar-valued rational approximation, as in (2.12):

$$\widetilde{H}(s,p) = \frac{n(s,p)}{d(s,p)} = \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\left(\mathbf{w}^{\top}\mathbf{H}(\sigma_i,\pi_j)\mathbf{v}\right)\alpha_{ij}}{(s-\sigma_i)(p-\pi_j)} \Big/ \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij}}{(s-\sigma_i)(p-\pi_j)}. \tag{2.33}$$

Note that $\beta_{ij} = \mathbf{w}^{\top}\mathbf{H}(\sigma_i,\pi_j)\mathbf{v}\alpha_{ij}$. Then, the final matrix-valued approximant $\widetilde{\mathbf{H}}(s,p)$ is

obtained by setting $\tilde{\alpha}_{ij} = \alpha_{ij}$ and $\mathbf{B}_{ij} = \alpha_{ij}\mathbf{H}(\sigma_i, \pi_j)$ in (2.31), resulting in

$$\widetilde{\mathbf{H}}(s,p) = \frac{\mathbf{N}(s,p)}{d(s,p)} = \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\mathbf{H}_{ij}\alpha_{ij}}{(s-\sigma_i)(p-\pi_j)} \Big/ \sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij}}{(s-\sigma_i)(p-\pi_j)}. \tag{2.34}$$

As in the scalar p-AAA case, by construction, our choice of $\mathbf{B}_{ij}$ guarantees interpolation of the data for the samples $\{\boldsymbol{\sigma}, \boldsymbol{\pi}\}$ in (2.30). However, the (linearized) LS minimization is different. We summarize these results next.

Proposition 2.6. *Given the sampling data (2.29), let $\widetilde{\mathbf{H}}(s,p)$ in (2.34) be the resulting approximant obtained via MIMO p-AAA with $\alpha_{ij} \neq 0$ and with the corresponding data partitioning (2.30). Then, $\widetilde{\mathbf{H}}(s,p)$ interpolates the data in $\mathbf{D}_{\boldsymbol{\sigma}\boldsymbol{\pi}}$ corresponding to the samples $\{\boldsymbol{\sigma}, \boldsymbol{\pi}\}$, i.e.,*

$$\widetilde{\mathbf{H}}(\sigma_i, \pi_j) = \mathbf{H}(\sigma_i, \pi_j) \quad \text{for} \quad i = 1, \ldots, k \quad \text{and} \quad j = 1, \ldots, q. \tag{2.35}$$

Furthermore, $\widetilde{\mathbf{H}}(s,p)$ minimizes an input/output weighted linearized LS measure, namely

$$\widetilde{\mathbf{H}} = \underset{\hat{\mathbf{H}}=\mathbf{N}/d}{\arg\min} \sum_{i,j} \left| \mathbf{w}^{\top} \big(\mathbf{H}(s_i, p_j) d(s_i, p_j) - \mathbf{N}(s_i, p_j) \big) \mathbf{v} \right|^2 \tag{2.36}$$

for the data in $\{\mathbf{D}_{\boldsymbol{\sigma}\hat{\boldsymbol{\pi}}}, \mathbf{D}_{\hat{\boldsymbol{\sigma}}\boldsymbol{\pi}}, \mathbf{D}_{\hat{\boldsymbol{\sigma}}\hat{\boldsymbol{\pi}}}\}$, not selected by the greedy search, i.e., corresponding to the sampling pairs $(s_i, p_j) \in \{\hat{\boldsymbol{\sigma}} \times \boldsymbol{\pi}\} \cup \{\boldsymbol{\sigma} \times \hat{\boldsymbol{\pi}}\} \cup \{\hat{\boldsymbol{\sigma}} \times \hat{\boldsymbol{\pi}}\}$.

Proof. The interpolation property (2.35) follows analogously to the scalar case, by observing that for $\alpha_{ij} \neq 0$, $\widetilde{\mathbf{H}}(s,p)$ has a removable pole at each (σ_i, π_j) with

$$\widetilde{\mathbf{H}}(\sigma_i, \pi_j) = \frac{\mathbf{B}_{ij}}{\alpha_{ij}}.$$

Then, the choice $\mathbf{B}_{ij} = \alpha_{ij}\mathbf{H}_{ij}$ proves (2.35).

To prove (2.36), first recall that $\widetilde{H}(s,p)$ in (2.32) is obtained by applying (scalar-valued)

p-AAA to the data (2.32). Therefore, by Corollary 2.2,

$$\widetilde{H} = \underset{\hat{H}=d/n}{\arg\min} \sum_{i,j} \mid \mathbf{w}^\top \mathbf{H}(s_i, p_j)\mathbf{v} d(s_i, p_j) - n(s_i, p_j) \mid^2 . \tag{2.37}$$

Using (2.33) and (2.34), we have

$$\widetilde{H}(s,p) = \frac{n(s,p)}{d(s,p)} = \mathbf{w}^\top \widetilde{\mathbf{H}}(s,p)\mathbf{v} = \frac{\mathbf{w}^\top \mathbf{N}(s,p)\mathbf{v}}{d(s,p)}.$$

Therefore,

$$\mathbf{w}^\top \mathbf{H}(s_i, p_j)\mathbf{v} d(s_i, p_j) - n(s_i, p_j) = \mathbf{w}^\top \big(\mathbf{H}(s_i, p_j) d(s_i, p_j) - \mathbf{N}(s_i, p_j)\big)\mathbf{v}.$$

Inserting this last equality into (2.37) proves (2.36). □

Remark 2.7. Proposition 2.6 states that for MIMO p-AAA, interpolation holds analogously to the scalar case. However, the LS minimization differs from the scalar case in that what is minimized is a weighted LS measure. More precisely, in terms of the LS aspect of MIMO p-AAA, the linearization is performed on the weighted error $\mathbf{w}^\top(\mathbf{H}(s,p) - \widetilde{\mathbf{H}}(s,p))\mathbf{v}$.

Remark 2.8. When the internal description of the underlying (transfer) function is available, as in (1.1) and (1.3), projection-based approaches are commonly used to construct interpolatory parametric approximants [13, 16, 26]. In this setting, for MIMO systems, one usually does not enforce full matrix interpolation. Instead, interpolation is enforced along selected *tangential directions*. In other words, one picks vectors $\mathbf{w}_i \in \mathbb{C}^{n_{out}}$ and $\mathbf{v}_i \in \mathbb{C}^{n_{in}}$ such that $\mathbf{H}(\sigma_i, \pi_j)\mathbf{v}_i = \widetilde{\mathbf{H}}(\sigma_i, \pi_j)\mathbf{v}_i$ and/or $\mathbf{w}_i^\top \mathbf{H}(\sigma_i, \pi_j) = \mathbf{w}_i^\top \widetilde{\mathbf{H}}(\sigma_i, \pi_j)$. This is called tangential interpolation. Tangential vectors usually vary with the sampling points. At this point, it is not clear, at least to us, how to achieve tangential interpolation using the barycentric form (2.31). However, inspired by this concept, instead of choosing two fixed vectors $\mathbf{w}$ and $\mathbf{v}$,

one could pick different vectors $\mathbf{w}_i$, and $\mathbf{v}_i$ for each sample σ_i, for example and apply MIMO p-AAA to the data $\mathbf{w}_i^\top \mathbf{H}_{ij}\mathbf{v}_i$ to build the MIMO approximation (2.34) as above. The resulting model $\widetilde{\mathbf{H}}(s,p)$ would still interpolate the data $\mathbf{D}_{\boldsymbol{\sigma\pi}}$ and minimize the LS error along varying weighted directions. In our experiments (see Section 2.3.2), fixed vectors $\mathbf{w}$ and $\mathbf{v}$ provided accurate approximations and therefore we do not pursue the idea of choosing different vectors here. The interpolatory parametric-Loewner approach [80] handles vector-valued problems, i.e., $\mathbf{H}(s_i,p_j) \in \mathbb{C}^{n_{out}\times 1}$, in a similar manner by choosing $\mathbf{w}$ as vector of ones (and $\mathbf{v} = 1$ since $n_{in} = 1$). Recent work [62] bypasses this by choosing matrix-valued weights in the barycentric form.

Remark 2.9. The choice of random directions $\mathbf{v}$ and $\mathbf{w}$ allows MIMO p-AAA to use the information relating all inputs with all outputs. As in tangential interpolation, if, for example, one were to use the canonical vectors $\mathbf{e}_i$ and $\mathbf{e}_j$ as directions, then the algorithm would only have access to information on the relation between the ith input and jth output.

2.3.2 Numerical examples: two stationary PDEs

We consider two examples from [44]. The first is the stationary PDE

$$w_{xx} + pw_{yy} + zw = 10\sin(8x(y-1)) \quad \text{on } \Omega = [-1,1]\times[-1,1],$$

with homogeneous Dirichlet boundary conditions. The solution $w(x,y)$ depends on the two parameters (p,z) and is independent of time. Therefore, the model is not a dynamical system, unlike our previous examples, yet this does not matter for our formulation, since we simply view the solution as a function of two-variables. The *truth model* is the spectral Chebyshev collocation with 50 nodes in each direction from [44]. We choose to approximate $w(x,y)$ on the whole domain Ω; thus the output is the full solution, leading to a two-variable

vector-valued function to sample $\mathbf{H}(p, z) \in \mathbb{R}^{2401\times 1}$. In our MIMO p-AAA terminology, we interpret this as a model with $n_{in} = 1$ and $n_{out} = 2401$. We take $N = M = 10$ linearly spaced measurements of $\mathbf{H}(p, z)$ in the parameter space $[0.1, 4] \times [0, 2]$. The usual projection-based approaches to PMOR would form a global basis from these samples and project the truth model into a low-dimensional space. However, we do not assume access to the truth model, but only its samples via black-box simulation; we construct our approximation directly from these samples. MIMO p-AAA leads to an approximation with orders $q = 3$ in p and $o = 3$ in z. To judge the quality of the approximation, we perform a parameter sweep in the full parameter domain and find the worst case scenario in terms of the maximum error between the truth model and the MIMO p-AAA approximation over Ω. The worst-case approximation occurs for $p = 1.7545$ and $z = 2$, with an error of 3.11×10^{-2}, showing that the MIMO p-AAA approximant is accurate even in the worst-case. This worst case scenario is depicted in the *left pane* of Figure 2.13, where the top plot shows the truth model, the middle one the MIMO p-AAA approximation, and the bottom one the error plot. As the figure illustrates, MIMO p-AAA is able to recover the solution on the whole domain accurately.

We also apply MIMO p-AAA to a slightly revised PDE from [44]:

$$(1 + px)w_{xx} + (1 + zy)w_{yy} = e^{4xy} \qquad \text{on } \Omega = [-1, 1] \times [-1, 1].$$

The set-up is the same as above: we impose Dirichlet boundary conditions, and obtain the *truth model* via Chebyshev collocation, with 49 nodes in each direction, leading to a two-variable vector-valued function to sample $\mathbf{H}(p, z) \in \mathbb{R}^{2401\times 1}$. We sample $\mathbf{H}(p, z)$ at $N = M = 10$ linearly spaced points in the parameter domain $(p, z) \in [-0.99, 0.99] \times [-0.99, 0.99]$ and apply MIMO p-AAA. As stated in [44], this problem is harder to approximate than the first one due to near singularities at the corners of the parameter domain. This is automatically reflected in the approximation orders MIMO p-AAA chooses: $q = 5$ in p and $o = 6$ in z.

As for the first PDE, we perform a parameter sweep in the full parameter domain to find the worst-case performance. In this case, the worst approximation occurs for $p = 0.95$ and $z = 0.99$, with an error of 7.28×10^{-2}, an accurate approximation even in the worst case. We show the results from this worst case in the *right pane* of Figure 2.13, where the top plot shows the truth model, the middle one the MIMO p-AAA approximation, and the bottom one the error plot. As in the previous case, MIMO p-AAA accurately captures the full solution.

2.4 Summary

We have presented a data-driven modeling framework for approximating parametric (dynamical) systems by extending the AAA algorithm to multivariate problems. The method does not require access to an internal state-space description, and works with function evaluations. We have discussed the scalar-valued problem as well as the matrix-valued case. Seven numerical examples have been used to illustrate the effectiveness of the proposed approach.

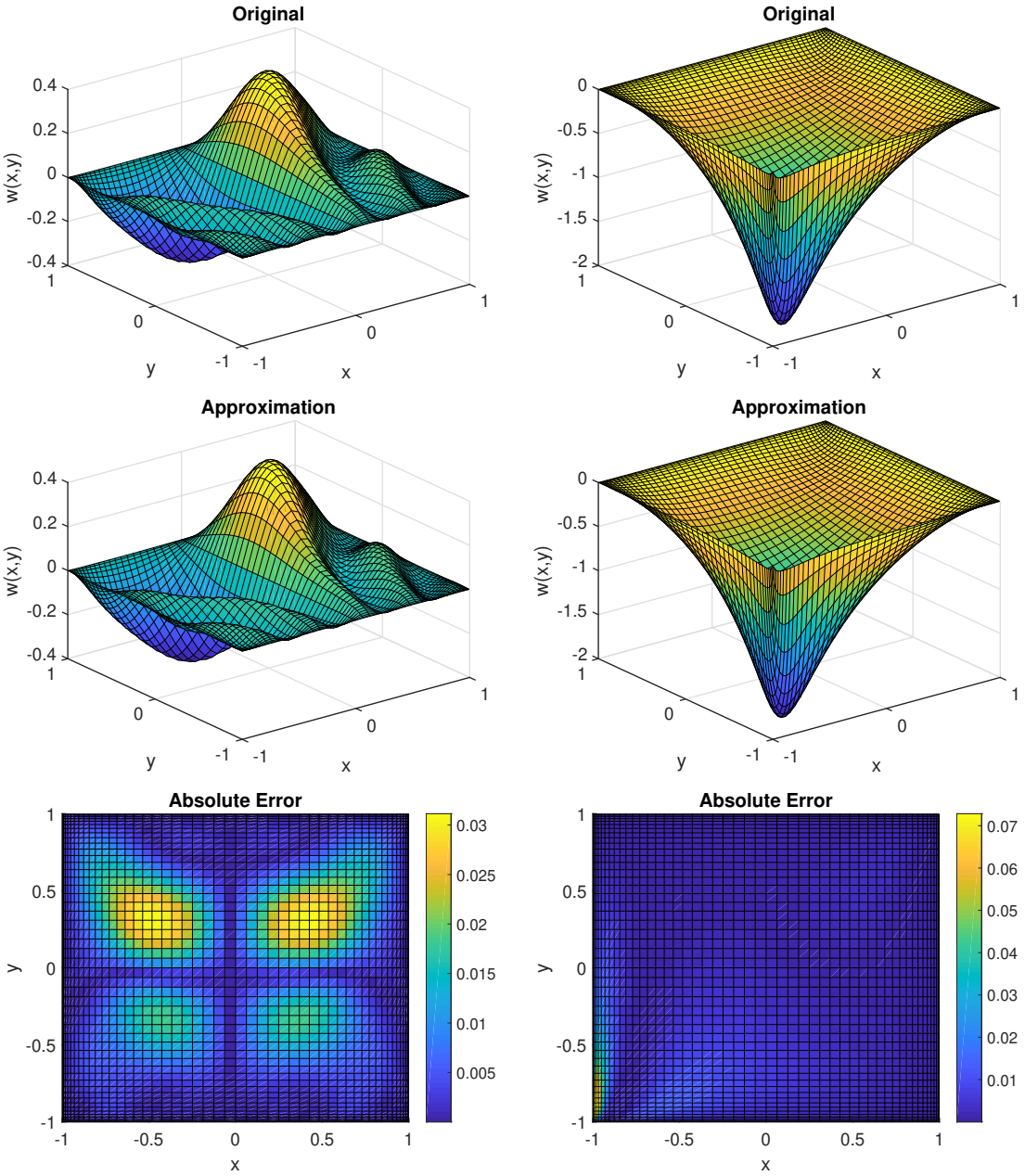


Figure 2.13: Example Section 2.3.2. MIMO p-AAA approximations for two two-variable PDEs

Chapter 3

p-AAA for Nonlinear Parametric Eigenvalue Problems

In this chapter, we first review a method from [92] that employs the AAA algorithm to solve nonlinear eigenvalue problems (NLEVP). We then extend this approach by applying p-AAA from Chapter 2 to approximate parametric nonlinear eigenvalue problems (PNLEVP). We illustrate our method with three examples.

A common example where NLEVPs arise is in the stability analysis of delay differential equations [70], such as

$$\dot{\mathbf{v}}(t) + \mathbf{A}\mathbf{v}(t) + \mathbf{B}\mathbf{v}(t-1) = 0,$$

where $\mathbf{v}(t) \in \mathbb{C}^n$ is known for $t \in [-1, 0]$. Then solutions have the form

$$\mathbf{v}(t) = e^{\lambda t}\mathbf{x},$$

where $\mathbf{x} \neq \mathbf{0}$ is a solution to the NLEVP

$$\mathbf{T}(\lambda)\mathbf{x} = (\lambda\mathbf{I} + \mathbf{A} + \mathbf{B}e^{-\lambda})\mathbf{x} = \mathbf{0}.$$

Here $\mathbf{x}$ is the eigenvector corresponding to the eigenvalue λ. For this example, the delay is set to be 1. However one may only have a rough estimate of the delay. Then parametrizing

the delay

$$\dot{\mathbf{v}}(t) + \mathbf{A}\mathbf{v}(t) + \mathbf{B}\mathbf{v}(t-p) = 0$$

and studying the corresponding PNLEVP

$$\mathbf{T}(\lambda, p)\mathbf{x} = (\lambda\mathbf{I} + \mathbf{A} + \mathbf{B}e^{-\lambda p})\mathbf{x} = \mathbf{0}$$

will give insight on how the delay affects stability.

Our approach consists of three steps: finding a rational approximation to the NLEVP, linearizing this rational EVP, and finding the eigenvalues for a specific parameter value. The method in [92] consists on the same three steps for standard NLEVPs. Hence one could apply it to PNLEVP for each parameter value of interest. This means creating a linearization for each parameter. Our method creates a parametric linearization once, hence saving computational effort when reevaluating new parameters. This may not be a significant advantage if one is only interested on few parameter values, but it certainly will be if many parameter values are of interest, e.g., when studying how the eigenvalues change for a continuous range of parameters or when studying the sensitivity of the eigenvalues to changes in the parameter.

3.1 Reviewing the nonparametric case

We start this section by providing the definition of the generalized linear eigenvalue problem.

Definition 3.1. Given $\mathbf{A}, \mathbf{B} \in \mathbb{C}^{n\times n}$, a *(linear) eigenvalue problem* (EVP) consists of finding scalars $\lambda \in \mathbb{C}$ and nonzero vectors $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$ such that

$$(\mathbf{A} - \lambda\mathbf{B})\mathbf{x} = \mathbf{0} \qquad \text{and} \qquad \mathbf{y}^*(\mathbf{A} - \lambda\mathbf{B}) = \mathbf{0}.$$

The scalar λ and the vectors $\mathbf{x}$ and $\mathbf{y}$ are called an *eigenvalue*, a *right eigenvector*, and a *left eigenvector*, respectively.

EVPs arise in countless theoretical and applied problems, and hence are thoroughly studied. Here we focus on a generalization of this problem to include nonlinearities.

Definition 3.2. Let $\mathbf{T} : \mathbb{C} \to \mathbb{C}^{n\times n}$ be a nonlinear matrix valued function. A *nonlinear eigenvalue problem* (NLEVP) consists of finding scalars $\lambda \in \mathbb{C}$ and nonzero vectors $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$ such that

$$\mathbf{T}(\lambda)\mathbf{x} = 0 \quad \text{and} \quad \mathbf{y}^*\mathbf{T}(\lambda) = 0.$$

The field of NLEVP has seen much development in recent years: see [98] for a review of NLEVP applications and [70] for a general review of NLEVP methods. For specific methods exploiting specific nonlinear structures, such as quadratic, see [124]; and for the general polynomial case, see [94, 95] and the references therein. Common approaches for the general nonlinear case are: applying Newton-like iterations for the approximation of a single eigenvalue [85, 87]; expressing the eigenvalues in a region of interest as the poles of a resolvent and evaluating them with contour integration methods [35, 51]; approximating the NLEVP with a polynomial or rational function and then finding the eigenvalues of a linearization of the approximation by Krylov methods. Some popular methods for rational approximation are: NLEIGS [71], infinite Arnoldi [81], Padé approximation [123], and CORK [125].

We describe below the approach in [92] for NLEVPs, which combines the AAA algorithm [105] with the CORK algorithm [125]. It considers a representation that separates the polynomial and rational parts of $\mathbf{T}(\lambda)$ from general nonlinearities. Then it replaces the nonlinear part with a rational approximation and approximates the eigenvalues of the resulting polynomial-rational approximation. We do not include any proofs here, since the original source [92] provides those details and we will prove our generalizations in the next section.

Rational EVPs can be solved exactly by *linearizing* the equation into a generalized EVP of larger dimension. The general philosophy behind CORK is to separate the rational and nonlinear parts

$$\mathbf{T}(\lambda) = \mathbf{P}(\lambda) + \mathbf{G}(\lambda), \tag{3.1}$$

where the rational part $\mathbf{P}(\lambda)$ can be linearized and the nonlinear part $\mathbf{G}(\lambda)$ will be replaced by a AAA rational approximation, then linearized. Specifically, assume $\mathbf{T}(\lambda)$ has the form as in (3.1) where

$$\mathbf{P}(\lambda) = \sum_{\ell=0}^{d-1} (\mathbf{A}_\ell - \lambda \mathbf{B}_\ell) f_\ell(\lambda), \tag{3.2}$$

and

$$\mathbf{G}(\lambda) = \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) g_\ell(\lambda), \tag{3.3}$$

where $\mathbf{A}_\ell, \mathbf{B}_\ell, \mathbf{C}_\ell, \mathbf{D}_\ell \in \mathbb{C}^{n\times n}$ are constant matrices, $g_\ell : \mathbb{C} \to \mathbb{C}$ are nonlinear functions, and $f_\ell : \mathbb{C} \to \mathbb{C}$ are polynomial or rational functions that satisfy

$$(\mathbf{M} - \lambda \mathbf{N})\mathbf{f}(\lambda) = 0,$$

for $\mathbf{f}(\lambda) = [f_0(\lambda)\ f_1(\lambda)\ \cdots\ f_{d-1}(\lambda)]^\top$ and for some $\mathbf{M}, \mathbf{N} \in \mathbb{C}^{(d-1)\times d}$ with

$$\operatorname{rank}(\mathbf{M} - \lambda \mathbf{N}) = d - 1 \qquad \forall \lambda \in \mathbb{C}.$$

If $L = 0$, then $\mathbf{T}(\lambda) = \mathbf{P}(\lambda)$ and its eigenvalues are the same as those of the matrix pencil

$$\mathcal{L}_{\mathbf{P}}(\lambda) = \left[\begin{array}{c|c} \mathbf{A}_0 - \lambda\mathbf{B}_0 \ \cdots \ \mathbf{A}_{d-1} - \lambda\mathbf{B}_{d-1} \\ \hline (\mathbf{M} - \lambda\mathbf{N}) \otimes \mathbf{I}_n \end{array}\right].$$

The eigenvalues of $\mathcal{L}_{\mathbf{P}}(\lambda) \in \mathbb{C}^{dn \times dn}$ are computed efficiently via the compact rational Krylov (CORK) method described in [125]. Hence $\mathcal{L}_{\mathbf{P}}(\lambda)$ is called the CORK linearization of $\mathbf{P}(\lambda)$. In [92], the authors present a CORK-like linearization for the case $L \neq 0$. First, [92] applies AAA to $g_\ell(\lambda)$ to obtain the rational approximant

$$g_\ell(\lambda) \approx r_\ell(\lambda) = \sum_{i=1}^{k} \frac{g_\ell(\sigma_i)\alpha_i}{(\lambda - \sigma_i)} \Big/ \sum_{i=1}^{k} \frac{\alpha_i}{(\lambda - \sigma_i)}. \tag{3.4}$$

Using the AAA approximation (3.4), define

$$\mathbf{a}_\ell = \begin{bmatrix} g_\ell(\sigma_1)\alpha_1 \\ \vdots \\ g_\ell(\sigma_k)\alpha_k \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{R}^k,$$

$$\mathbf{E} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_{k-1} & \alpha_k \\ -\sigma_1 & \sigma_2 & & & \\ & -\sigma_2 & \ddots & & \\ & & \ddots & \sigma_{k-1} & \\ & & & -\sigma_{k-1} & \sigma_k \end{bmatrix}, \quad \text{and} \quad \mathbf{F} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ -1 & 1 & & & \\ & -1 & \ddots & & \\ & & \ddots & 1 & \\ & & & -1 & 1 \end{bmatrix}.$$

With these, $\mathbf{G}(\lambda)$ in (3.3) is approximated by

$$\mathbf{G}(\lambda) \approx \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) r_\ell(\lambda) \tag{3.5}$$

$$= \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) \mathbf{a}_\ell^\top (\mathbf{E} - \lambda \mathbf{F})^{-1} \mathbf{b}. \tag{3.6}$$

Using this rational approximation for $\mathbf{G}(\lambda)$ in (3.6), we approximate the NLEVP with

$$\mathbf{T}(\lambda) \approx \mathbf{R}(\lambda) = \mathbf{P}(\lambda) + \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) r_\ell(\lambda) \tag{3.7}$$

$$= \mathbf{P}(\lambda) + \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) \mathbf{a}_\ell^\top (\mathbf{E} - \lambda \mathbf{F})^{-1} \mathbf{b}. \tag{3.8}$$

Then the CORK linearization for $\mathbf{R}(\lambda)$ is

$$\mathcal{L}_{\mathbf{R}}(\lambda) = \left[\begin{array}{c|c} \mathbf{A}_0 - \lambda \mathbf{B}_0 \ \cdots \ \mathbf{A}_{d-1} - \lambda \mathbf{B}_{d-1} & \sum_{\ell=1}^{L} \mathbf{a}_\ell^\top \otimes (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) \\ \hline \left[\begin{array}{cc} \mathbf{M} - \lambda \mathbf{N} & 0 \\ -\mathbf{b} \ 0 & \mathbf{E} - \lambda \mathbf{F} \end{array}\right] \otimes \mathbf{I}_n & \end{array}\right], \tag{3.9}$$

which is of size $(d+k)n \times (d+k)n$. The eigenvalues of $\mathcal{L}_{\mathbf{R}}(\lambda)$ and $\mathbf{R}(\lambda)$ are the same. Hence, if $\mathbf{R}(\lambda)$ is a good approximation to $\mathbf{T}(\lambda)$, then the eigenvalues of $\mathcal{L}_{\mathbf{R}}$ are a good approximation of the eigenvalues of $\mathbf{T}(\lambda)$. Note that $\mathbf{T}(\lambda)$ might have infinitely many eigenvalues while $\mathcal{L}_{\mathbf{R}}(\lambda)$ has only finitely many. The eigenvalues of $\mathcal{L}_{\mathbf{R}}(\lambda)$ are used to approximate the eigenvalues of $\mathbf{T}(\lambda)$ that are not poles of $\mathbf{R}(\lambda)$ and are inside the region sampled for the AAA approximation. The authors in [92] apply the CORK algorithm to this linearization of various NLEVPs to illustrate its accuracy. The compact rational Krylov (CORK) algorithm [125] is a generalization of Arnoldi decomposition and takes advantage of the Kronecker structure in (3.9) to save memory and orthogonalization cost.

To summarize the method in [92]:

- Step 1: Approximate the nonlinear $\mathbf{T}(\lambda)$ with the rational $\mathbf{R}(\lambda)$ via AAA.
- Step 2: Approximate the eigenvalues of $\mathbf{R}(\lambda)$ via CORK.

In the next section, we generalize this method to problems that depend on a parameter.

3.2 Parametric case

Consider now a PNLEVP.

Definition 3.3. A *parametric nonlinear eigenvalue problem* (PNLEVP) consists of finding scalars $\lambda \in \mathbb{C}$ and nonzero vectors $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$ such that

$$\mathbf{T}(\lambda, p)\mathbf{x} = 0 \quad \text{and} \quad \mathbf{y}^*\mathbf{T}(\lambda, p) = 0,$$

for some $p \in \mathcal{P} \subset \mathbb{R}$ where $\mathbf{T} : \mathbb{C} \times \mathcal{P} \to \mathbb{C}^{n\times n}$ is a nonlinear matrix function.

We present here analogous steps to those in Section 3.1 to approximate the solution of a PNLEVP. We first separate the polynomial/rational parts that do not depend on the parameter p from the nonlinearities and the p dependencies. This is, we assume $\mathbf{T}(\lambda, p)$ has the form

$$\mathbf{T}(\lambda, p) = \mathbf{P}(\lambda) + \mathbf{G}(\lambda, p), \tag{3.10}$$

with $\mathbf{P}(\lambda)$ as in (3.2) and

$$\mathbf{G}(\lambda, p) = \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) g_\ell(\lambda, p)$$

where $g_\ell : \mathbb{C} \times \mathcal{P} \to \mathbb{C}$ is a multivariate nonlinear function, for $\ell = 1, \ldots, L$. For example, consider the Hadeler problem from Section 3.3.2:

$$\mathbf{T}(\lambda, p) = (e^{p\lambda} - 1)\mathbf{A}_2 + \lambda^2 \mathbf{A}_1 - \alpha \mathbf{A}_0,$$

where $\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2 \in \mathbb{R}^{n \times n}$ and $\alpha \in \mathbb{R}$. This problem has the form in (3.10) with

$$\mathbf{P}(\lambda) = \lambda^2 \mathbf{A}_1 - \alpha \mathbf{A}_0 \qquad \text{and}$$
$$\mathbf{G}(\lambda) = (e^{p\lambda} - 1)\mathbf{A}_2.$$

The key steps in defining the CORK linearization $\mathcal{L}_{\mathbf{R}}(\lambda)$ (3.9) for the non-parametric problem are (3.7), which approximates the nonlinearities $g_\ell(\lambda)$ with AAA,

$$g_\ell(\lambda) \approx r_\ell(\lambda),$$

and (3.8), which rewrites the AAA approximation as

$$r_\ell(\lambda) = \mathbf{a}_\ell^\top (\mathbf{E} - \lambda \mathbf{F})^{-1} \mathbf{b}.$$

Inspired by this, for the PNLEVP

$$\mathbf{T}(\lambda, p) = \mathbf{P}(\lambda) + \sum_{\ell=1}^{L} (\mathbf{C}_\ell - \lambda \mathbf{D}_\ell) g_\ell(\lambda, p),$$

we apply the p-AAA algorithm from Chapter 2 to approximate the nonlinearities $g_\ell(\lambda, p)$ viewing the variable λ here as the frequency variable s from Chapter 2. Then we obtain the

rational approximation

$$g_\ell(\lambda,p) \approx r_\ell(\lambda,p) = \frac{\displaystyle\sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij} g_\ell(\sigma_i,\pi_j)}{(\lambda-\sigma_i)(p-\pi_j)}}{\displaystyle\sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij}}{(\lambda-\sigma_i)(p-\pi_j)}}$$

and we use it to define

$$\mathbf{b} = [1\ 0\ \cdots\ 0]^\top \in \mathbb{R}^{kq},$$

$$\mathbf{a}_\ell^\top = [g_\ell(\sigma_1,\pi_1)\alpha_{11}\ \cdots\ g_\ell(\sigma_1,\pi_q)\alpha_{1,q} \mid \cdots \mid g_\ell(\sigma_k,\pi_1)\alpha_{k,1}\ \cdots\ g_\ell(\sigma_k,\pi_q)\alpha_{k,q}],$$

$$\mathbf{E}(p) = \operatorname{diag}\left(\sigma_1\mathbf{f}_1(p), \sigma_2\mathbf{f}_2(p), \ldots, \sigma_k\mathbf{f}_2(p)\right) + \begin{bmatrix} \alpha_{11}\ \cdots\ \alpha_{1q} \mid \cdots \mid \alpha_{k1}\ \cdots\ \alpha_{kq} \\ \operatorname{diag}\left(\sigma_1\mathbf{f}_2(p), \ldots, \sigma_{k-1}\mathbf{f}_2(p), \sigma_k\mathbf{f}_3(p)\right) \end{bmatrix}, \text{ and}$$

$$\mathbf{F}(p) = \operatorname{diag}\left(\mathbf{f}_1(p), \underbrace{\mathbf{f}_2(p), \ldots, \mathbf{f}_2(p)}_{k-1 \text{ times}}\right) + \begin{bmatrix} 0\ \cdots\ 0 \\ \operatorname{diag}\left(\underbrace{-\mathbf{f}_2(p), \ldots, -\mathbf{f}_2(p)}_{k-1 \text{ times}}, -\mathbf{f}_3(p)\right) \end{bmatrix},$$

where

$$\mathbf{f}_1(p) = [0\ p-\pi_2\ \cdots\ p-\pi_q],$$

$$\mathbf{f}_2(p) = [p-\pi_1\ \cdots\ p-\pi_q], \text{and}$$

$$\mathbf{f}_3(p) = [p-\pi_1\ \cdots\ p-\pi_{q-1}].$$

Then we can rewrite the p-AAA rational approximation $r_\ell(\lambda,p)$ analogously to (3.8):

$$r_\ell(\lambda,p) = \mathbf{a}_\ell^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}. \tag{3.11}$$

We prove (3.11) in two steps: Lemma 3.4 and Proposition 3.5. We pick specific small orders $k = q = 2$ for simplicity of the presentation. The proof is analogous for higher orders.

Lemma 3.4. *Let $r_\ell(\lambda, p)$ be a two-variable rational function with barycentric representation*

$$r_\ell(\lambda, p) = \frac{n(\lambda, p)}{d(\lambda, p)} = \frac{\displaystyle\sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij} g_\ell(\sigma_i, \pi_j)}{(\lambda - \sigma_i)(p - \pi_j)}}{\displaystyle\sum_{i=1}^{k}\sum_{j=1}^{q} \frac{\alpha_{ij}}{(\lambda - \sigma_i)(p - \pi_j)}},$$

where $k = q = 2$. Consider $\mathbf{E}(p)$, $\mathbf{F}(p)$, and $\mathbf{b}$ as in (3.11). Then

$$(\mathbf{E}(p) - \lambda \mathbf{F}(p)) \frac{1}{d(\lambda, p)} \begin{bmatrix} (\lambda - \sigma_1)^{-1}(p - \pi_1)^{-1} \\ (\lambda - \sigma_1)^{-1}(p - \pi_2)^{-1} \\ (\lambda - \sigma_2)^{-1}(p - \pi_1)^{-1} \\ (\lambda - \sigma_2)^{-1}(p - \pi_2)^{-1} \end{bmatrix} = \mathbf{b}. \tag{3.12}$$

Proof. First note that for $k = q = 2$, $\mathbf{E}(p)$ and $\mathbf{F}(p)$ are as follows:

$$\mathbf{E}(p) = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{21} & \alpha_{22} \\ -\sigma_1(p - \pi_1) & \sigma_1(p - \pi_2) & & \\ & -\sigma_1(p - \pi_2) & \sigma_2(p - \pi_1) & \\ & & -\sigma_2(p - \pi_1) & \sigma_2(p - \pi_2) \end{bmatrix}$$

and

$$\mathbf{F}(p) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -(p - \pi_1) & (p - \pi_2) & & \\ & -(p - \pi_2) & (p - \pi_1) & \\ & & -(p - \pi_1) & (p - \pi_2) \end{bmatrix}.$$

Then

$$\mathbf{E}(p)-\lambda\mathbf{F}(p)=\begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{21} & \alpha_{22} \\ (\lambda-\sigma_1)(p-\pi_1) & -(\lambda-\sigma_1)(p-\pi_2) & & \\ & (\lambda-\sigma_1)(p-\pi_2) & -(\lambda-\sigma_2)(p-\pi_1) & \\ & & (\lambda-\sigma_2)(p-\pi_1) & -(\lambda-\sigma_2)(p-\pi_2) \end{bmatrix}$$

and we have (3.12):

$$(\mathbf{E}(p)-\lambda\mathbf{F}(p))\,\frac{1}{d(\lambda,p)}\begin{bmatrix} (\lambda-\sigma_1)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_1)^{-1}(p-\pi_2)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_2)^{-1} \end{bmatrix}$$

$$=\frac{1}{d(\lambda,p)}\begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{21} & \alpha_{22} \\ (\lambda-\sigma_1)(p-\pi_1) & -(\lambda-\sigma_1)(p-\pi_2) & & \\ & (\lambda-\sigma_1)(p-\pi_2) & -(\lambda-\sigma_2)(p-\pi_1) & \\ & & (\lambda-\sigma_2)(p-\pi_1) & -(\lambda-\sigma_2)(p-\pi_2) \end{bmatrix}\begin{bmatrix} (\lambda-\sigma_1)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_1)^{-1}(p-\pi_2)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_2)^{-1} \end{bmatrix}$$

$$=\frac{1}{d(\lambda,p)}\begin{bmatrix} \alpha_{11}(\lambda-\sigma_1)^{-1}(p-\pi_1)^{-1}+\alpha_{12}(\lambda-\sigma_1)^{-1}(p-\pi_2)^{-1}+\alpha_{21}(\lambda-\sigma_2)^{-1}(p-\pi_1)^{-1}+\alpha_{22}(\lambda-\sigma_2)^{-1}(p-\pi_2)^{-1} \\ 1-1+0+0 \\ 0+1-1+0 \\ 0+0+1-1 \end{bmatrix}$$

$$=\frac{1}{d(\lambda,p)}\begin{bmatrix} d(\lambda,p) \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$=\mathbf{b}$.

□

Proposition 3.5. *Let $r_\ell(\lambda,p)$ be as in Lemma 3.4. Then (3.11) is satisfied.*

Proof. From Lemma 3.4, we know (3.12) holds:

$$(\mathbf{E}(p) - \lambda\mathbf{F}(p))\,\frac{1}{d(\lambda,p)}\begin{bmatrix} (\lambda-\sigma_1)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_1)^{-1}(p-\pi_2)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_2)^{-1} \end{bmatrix} = \mathbf{b}.$$

Recall that, for $k = q = 2$,

$$r_\ell(\lambda,p) = \frac{n(\lambda,p)}{d(\lambda,p)} = \frac{\displaystyle\sum_{i=1}^{2}\sum_{j=1}^{2}\frac{\alpha_{ij}g_\ell(\sigma_i,\pi_j)}{(\lambda-\sigma_i)(p-\pi_j)}}{\displaystyle\sum_{i=1}^{2}\sum_{j=1}^{2}\frac{\alpha_{ij}}{(\lambda-\sigma_i)(p-\pi_j)}}.$$

Then

$$\begin{aligned}
&\mathbf{a}_\ell^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b} \\
&= \mathbf{a}_\ell^\top\frac{1}{d(\lambda,p)}\begin{bmatrix} (\lambda-\sigma_1)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_1)^{-1}(p-\pi_2)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_2)^{-1} \end{bmatrix} \\
&= \frac{1}{d(\lambda,p)}\begin{bmatrix} g_\ell(\sigma_1,\pi_1)\alpha_{11} & g_\ell(\sigma_1,\pi_2)\alpha_{12} & g_\ell(\sigma_2,\pi_1)\alpha_{21} & g_\ell(\sigma_2,\pi_2)\alpha_{22} \end{bmatrix}\begin{bmatrix} (\lambda-\sigma_1)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_1)^{-1}(p-\pi_2)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_1)^{-1} \\ (\lambda-\sigma_2)^{-1}(p-\pi_2)^{-1} \end{bmatrix} \\
&= \frac{1}{d(\lambda,p)}\,n(\lambda,p) \\
&= r_\ell(\lambda,p).
\end{aligned}$$

□

Then the linearization $\mathcal{L}_{\mathbf{R}}(\lambda)$ in (3.9) for the parametric NLEVP in (3.1) becomes

$$\mathcal{L}_{\mathbf{R}}(\lambda,p)=\left[\begin{array}{c|c}\mathbf{A}_0-\lambda\mathbf{B}_0\ \cdots\ \mathbf{A}_{d-1}-\lambda\mathbf{B}_{d-1} & \sum_{\ell=1}^{L}\mathbf{a}_\ell^\top\otimes(\mathbf{C}_\ell-\lambda\mathbf{D}_\ell)\\ \hline \begin{bmatrix}\mathbf{M}-\lambda\mathbf{N} & 0\\ -\mathbf{b}\ 0 & \mathbf{E}(p)-\lambda\mathbf{F}(p)\end{bmatrix}\otimes\mathbf{I}_n\end{array}\right]. \tag{3.13}$$

To summarize:

$$\begin{aligned}\mathbf{T}(\lambda,p)&=\mathbf{P}(\lambda)+\mathbf{G}(\lambda,p)\\&=\sum_{\ell=0}^{d-1}(\mathbf{A}_\ell-\lambda\mathbf{B}_\ell)f_\ell(\lambda)+\sum_{\ell=0}^{L}(\mathbf{C}_\ell-\lambda\mathbf{D}_\ell)g_\ell(\lambda,p)\\&\approx\sum_{\ell=0}^{d-1}(\mathbf{A}_\ell-\lambda\mathbf{B}_\ell)f_\ell(\lambda)+\sum_{\ell=0}^{L}(\mathbf{C}_\ell-\lambda\mathbf{D}_\ell)r_\ell(\lambda,p)\qquad\text{(via p-AAA on } g_\ell)\\&=:\mathbf{R}(\lambda,p).\end{aligned} \tag{3.14}$$

The eigenvalues of $\mathcal{L}_{\mathbf{R}}(\lambda,p)$ in (3.13) are the same as the eigenvalues of $\mathbf{R}(\lambda,p)$ in (3.14) and hence approximate the eigenvalues of the PNLEVP $\mathbf{T}(\lambda,p)\mathbf{x}=\mathbf{0}$. We prove this for the simple case where there is only one polynomial term and one nonlinear term. This will simplify derivations significantly. More general cases follow analogously to our derivation below and the derivation for the nonparametric case in [92]. Consider then

$$\mathbf{T}(\lambda,p)=\mathbf{A}-\lambda\mathbf{B}+(\mathbf{C}-\lambda\mathbf{D})g(\lambda,p)$$

with $\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D}\in\mathbb{C}^{n\times n}$ and $g:\mathbb{C}\times\mathcal{P}\to\mathbb{C}$ is nonlinear. Then the corresponding $\mathbf{R}(\lambda,p)$

in (3.14) becomes

$$\mathbf{R}(\lambda, p) = \mathbf{A} - \lambda\mathbf{B} + (\mathbf{C} - \lambda\mathbf{D})r(\lambda, p)$$

and the linearization (3.13) becomes

$$\mathcal{L}_{\mathbf{R}}(\lambda, p) = \begin{bmatrix} \mathbf{A} - \lambda\mathbf{B} & | & \mathbf{a}^\top \otimes (\mathbf{C} - \lambda\mathbf{D}) \\ [-\mathbf{b} & \mathbf{E}(p) - \lambda\mathbf{F}(p)] \otimes \mathbf{I}_n \end{bmatrix}. \tag{3.15}$$

Now we will rewrite $\mathbf{R}(\lambda, p)$ with a Kronecker structure that will simplify the proofs of Propositions 3.6 and 3.7. First, by (3.11) we have

$$\mathbf{R}(\lambda, p) = \mathbf{A} - \lambda\mathbf{B} + (\mathbf{C} - \lambda\mathbf{D})\mathbf{a}^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}.$$

Note that $\mathbf{a}^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}$ is a scalar. Then, applying the Kronecker product property

$$(\mathbf{M}_1 \otimes \mathbf{M}_2)(\mathbf{M}_3 \otimes \mathbf{M}_4) = (\mathbf{M}_1\mathbf{M}_3) \otimes (\mathbf{M}_2\mathbf{M}_4) \tag{3.16}$$

for $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, and $\mathbf{M}_4$ matrices of conforming dimension, we obtain

$$\mathbf{R}(\lambda, p) = \mathbf{A} - \lambda\mathbf{B} + (\mathbf{a}^\top \otimes (\mathbf{C} - \lambda\mathbf{D}))(((\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}) \otimes \mathbf{I}_n).$$

To make the identity above clear, we rework it backwards:

$$\begin{aligned}
&(\mathbf{a}^\top \otimes (\mathbf{C} - \lambda\mathbf{D}))(((\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}) \otimes \mathbf{I}_n) \\
&= (\mathbf{a}^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}) \otimes ((\mathbf{C} - \lambda\mathbf{D})\mathbf{I}_n) \qquad \text{by (3.16)} \\
&= (\mathbf{a}^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}) \otimes (\mathbf{C} - \lambda\mathbf{D}) \\
&= (\mathbf{a}^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b})(\mathbf{C} - \lambda\mathbf{D}) \\
&= (\mathbf{C} - \lambda\mathbf{D})\mathbf{a}^\top(\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}.
\end{aligned}$$

Hence we have

$$\mathbf{R}(\lambda, p) = \mathbf{A} - \lambda\mathbf{B} + (\mathbf{a}^\top \otimes (\mathbf{C} - \lambda\mathbf{D}))(((\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b}) \otimes \mathbf{I}_n)$$

and, by block matrix multiplication

$$\mathbf{R}(\lambda, p) = [\mathbf{A} - \lambda\mathbf{B} \;\mid\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda\mathbf{D})] \begin{bmatrix} \mathbf{I}_n \\ (\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b} \otimes \mathbf{I}_n \end{bmatrix}.$$

Finally, defining

$$\Psi(\lambda, p) := \begin{bmatrix} \mathbf{I}_n \\ (\mathbf{E}(p) - \lambda\mathbf{F}(p))^{-1}\mathbf{b} \otimes \mathbf{I}_n \end{bmatrix}, \tag{3.17}$$

we obtain

$$\mathbf{R}(\lambda, p) = [\mathbf{A} - \lambda\mathbf{B} \;\mid\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda\mathbf{D})]\Psi(\lambda, p). \tag{3.18}$$

Note that the first factor of $\mathbf{R}(\lambda, p)$ in (3.18) is the first row block of $\mathcal{L}_{\mathbf{R}}(\lambda, p)$ in (3.15) and

that $\Psi(\lambda, p)$ can also be written as

$$\Psi(\lambda, p) = \begin{bmatrix} 1 \\ (\mathbf{E}(p) - \lambda \mathbf{F}(p))^{-1}\mathbf{b} \end{bmatrix} \otimes \mathbf{I}_n. \tag{3.19}$$

Proposition 3.6. *Fix $p = \pi$. If λ^* is an eigenvalue of $\mathbf{R}(\lambda, \pi)$, then λ^* is an eigenvalue of $\mathcal{L}_{\mathbf{R}}(\lambda, \pi)$.*

Proof. Suppose λ^* is an eigenvalue of $\mathbf{R}(\lambda, \pi)$. Then there exists a nonzero $\mathbf{x} \in \mathbb{C}^n$ such that

$$\mathbf{R}(\lambda^*, \pi)\mathbf{x} = \mathbf{0}. \tag{3.20}$$

Let

$$\mathbf{z} = \Psi(\lambda^*, \pi)\mathbf{x}, \tag{3.21}$$

with $\Psi(\lambda, p)$ in (3.17). Then

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ ((\mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi))^{-1}\mathbf{b} \otimes \mathbf{I}_n)\mathbf{x} \end{bmatrix}$$

and $\mathbf{z} \neq \mathbf{0}$ since $\mathbf{x} \neq \mathbf{0}$. Furthermore $(\lambda^*, \mathbf{z})$ is an eigenpair of $\mathcal{L}_{\mathbf{R}}(\lambda, \pi)$, i.e.,

$$\mathcal{L}_{\mathbf{R}}(\lambda^*, \pi)\mathbf{z} = \mathbf{0}$$

or, equivalently by (3.15),

$$\begin{bmatrix} \mathbf{A} - \lambda^* \mathbf{B} \;\; | \;\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda^* \mathbf{D}) \\ [-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)] \otimes \mathbf{I}_n \end{bmatrix} \mathbf{z} = \mathbf{0}.$$

We will show this in two steps by showing

$$[\mathbf{A} - \lambda^* \mathbf{B} \;\; | \;\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda^* \mathbf{D})]\mathbf{z} = \mathbf{0} \quad \text{and} \tag{3.22}$$

$$([-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)] \otimes \mathbf{I}_n)\mathbf{z} = \mathbf{0}. \tag{3.23}$$

First, we show (3.22) holds:

$$\begin{aligned} &[\mathbf{A} - \lambda^* \mathbf{B} \;\; | \;\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda^* \mathbf{D})]\mathbf{z} \\ &= [\mathbf{A} - \lambda^* \mathbf{B} \;\; | \;\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda^* \mathbf{D})]\Psi(\lambda^*, \pi)\mathbf{x} && \text{(by (3.21))} \\ &= \mathbf{R}(\lambda^*, \pi)\mathbf{x} && \text{(by (3.18))} \\ &= \mathbf{0}. && \text{(by (3.20))} \end{aligned}$$

Finally, we show (3.23):

$$\begin{aligned} &([-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)] \otimes \mathbf{I}_n)\mathbf{z} \\ &= ([-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)] \otimes \mathbf{I}_n)\Psi(\lambda^*, \pi)\mathbf{x} && \text{(by (3.21))} \\ &= ([-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)] \otimes \mathbf{I}_n) \left(\begin{bmatrix} 1 \\ (\mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi))^{-1}\mathbf{b} \end{bmatrix} \otimes \mathbf{I}_n \right) \mathbf{x} && \text{(by (3.19))} \\ &= \left(\left([-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)] \begin{bmatrix} 1 \\ (\mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi))^{-1}\mathbf{b} \end{bmatrix} \right) \otimes \mathbf{I}_n \right) \mathbf{x} && \text{(by } \otimes \text{ property)} \\ &= \mathbf{0}, \end{aligned}$$

since

$$[-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi)] \begin{bmatrix} 1 \\ (\mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi))^{-1}\mathbf{b} \end{bmatrix} = -\mathbf{b} + (\mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi))(\mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi))^{-1}\mathbf{b}$$
$$= -\mathbf{b} + \mathbf{b}$$
$$= \mathbf{0}.$$

□

Proposition 3.7. *Fix $p = \pi$. If λ^* is an eigenvalue of $\mathcal{L}_{\mathbf{R}}(\lambda, \pi)$, then λ^* is an eigenvalue of $\mathbf{R}(\lambda, \pi)$.*

Proof. Suppose λ^* is an eigenvalue of $\mathcal{L}_{\mathbf{R}}(\lambda, \pi)$. Let $\mathbf{z} \in \mathbb{C}^{n+kqn} \setminus \{\mathbf{0}\}$ denote an eigenvector corresponding to λ^* and let $\mathbf{z} = [\mathbf{z_{AB}} \;\; \mathbf{z_{CD}}]^\top$, where $\mathbf{z_{AB}} \in \mathbb{C}^n$ and $\mathbf{z_{CD}} \in \mathbb{C}^{kqn}$. Then

$$\begin{aligned} \mathbf{0} &= \mathcal{L}_{\mathbf{R}}(\lambda^*, \pi)\mathbf{z} \\ &= \begin{bmatrix} \mathbf{A} - \lambda^*\mathbf{B} \;\; | \;\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda^*\mathbf{D}) \\ [-\mathbf{b} \;\; \mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi)] \otimes \mathbf{I}_n \end{bmatrix} \begin{bmatrix} \mathbf{z_{AB}} \\ \mathbf{z_{CD}} \end{bmatrix}. \end{aligned} \tag{3.24}$$

From the second line in (3.24), we have

$$([\mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi)] \otimes \mathbf{I}_n)\mathbf{z_{CD}} = (\mathbf{b} \otimes \mathbf{I}_n)\mathbf{z_{AB}}.$$

Hence

$$\begin{aligned} \mathbf{z_{CD}} &= (\mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi) \otimes \mathbf{I}_n)^{-1}(\mathbf{b} \otimes \mathbf{I}_n)\mathbf{z_{AB}} \\ &= ([\mathbf{E}(\pi) - \lambda^*\mathbf{F}(\pi)]^{-1}\mathbf{b} \otimes \mathbf{I}_n)\mathbf{z_{AB}}, \end{aligned}$$

and so

$$\mathbf{z} = \begin{bmatrix} \mathbf{I}_n \\ [\mathbf{E}(\pi) - \lambda^* \mathbf{F}(\pi)]^{-1} \mathbf{b} \otimes \mathbf{I}_n \end{bmatrix} = \Psi(\lambda^*, \pi)\mathbf{z}_{\mathbf{AB}}.$$

Note that $\mathbf{z}_{\mathbf{AB}} \neq \mathbf{0}$ since $\mathbf{z} \neq \mathbf{0}$ and $\mathbf{z} = \Psi(\lambda^*, \pi)\mathbf{z}_{\mathbf{AB}}$. To conclude, we need to show $(\lambda^*, \mathbf{z}_{\mathbf{AB}})$ is an eigenpair of $\mathbf{R}(\lambda, \pi)$: From the first line in (3.24), we have

$$\mathbf{0} = [\mathbf{A} - \lambda^* \mathbf{B} \;\; | \;\; \mathbf{a}^\top \otimes (\mathbf{C} - \lambda^* \mathbf{D})]\Psi(\lambda^*, \pi)\mathbf{z}_{\mathbf{AB}} = \mathbf{R}(\lambda, \pi)\mathbf{z}_{\mathbf{AB}}.$$

□

We have now all the ingredients (the rational approximation of the NLEVP via p-AAA, and its linearization, which has the same eigenvalues and a CORK-like structure) to describe our method:

- Step 1: Approximate the nonlinear $\mathbf{T}(\lambda, p)$ in (3.10) with the rational $\mathbf{R}(\lambda, p)$ in (3.14) via p-AAA.
- Step 2: For a parameter value of interest $p = \pi$, find the eigenvalues of $\mathbf{R}(\lambda, \pi)$ via CORK.

3.3 Examples

In this section we present three examples to show the performance of our method. The examples in Sections 3.3.1 and 3.3.2 are of small dimension and hence do not require large-scale eigenvalue solvers. Therefore we build (3.13) and use `eig` in MATLAB. The example in Section 3.3.3 is of high dimension, making (3.13) also large and hence we apply the CORK algorithm.

3.3.1 Loaded String

We parametrize a rational eigenvalue problem from the NLEVP collection [34] to use as a toy example since the nonlinearity $g(\lambda, p)$ is already rational. The problem is the result of a finite element discretization of a string with a load attached to one end, and is defined by

$$R(\lambda, p) = \mathbf{A} - \lambda\mathbf{B} + \frac{\lambda}{\lambda - p/m}\mathbf{C}$$

where the parameter p represents the stiffness of the string, $m = 1$ is the mass of the load, and the $n \times n$ matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$ are given by

$$\mathbf{A} = \frac{1}{h}\begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & 2 & -1 \\ & & -1 & 1 \end{bmatrix}, \quad \mathbf{B} = \frac{h}{6}\begin{bmatrix} 4 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & 4 & 1 \\ & & 1 & 2 \end{bmatrix}, \quad \text{and} \quad \mathbf{C} = p\mathbf{e}_n\mathbf{e}_n^\top,$$

with $h = 1/n$. We choose $n = 100$. We run our p-AAA algorithm to build an approximation of the nonlinear part of $R(\lambda, p)$: $g(\lambda, p) := \dfrac{\lambda}{\lambda - p/m}$. We sampled for λ in $[3, 300]$ with $N = 100$ linearly spaced samples and for the parameter p at 0.1, 0.5, 1.5, and 2. For a tolerance of 1e-3, p-AAA builds an approximant with $(k, q) = (2, 2)$. We build the CORK linearization (3.13), evaluated at $p = 1$ and find its eigenvalues using `eig` in MATLAB. The five smallest eigenvalues for $p = 1$ are given in Table 3.1. In Figure 3.1 we show these eigenvalues in blue

Table 3.1: Values (from [120]) of the five smallest eigenvalues for $p = 1$.

λ_1	λ_2	λ_3	λ_4	λ_5
4.482176546	24.223572113	63.723821142	123.031221068	202.200899143

circles and our approximation to these eigenvalues in orange dots. This figure shows that

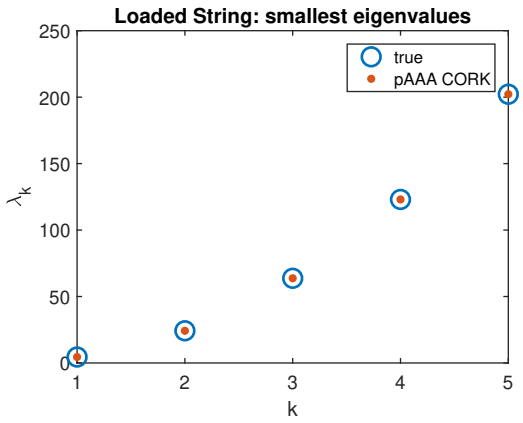


Figure 3.1: Example 3.3.1: Five smallest eigenvalues.

the CORK linearization that we propose for the parametric problem does indeed capture the eigenvalues accurately. We corroborate this accuracy further in Table 3.2, where we provide the absolute and relative error in the approximation of each eigenvalue. All the error values in Table 3.2 are at least in the order of 10^{-7}. Note that the values of interest (λ, p) for

Table 3.2: Accuracy of the approximation of the eigenvalues for $p = 1$ in Example 3.3.1.

Eigenvalue	Absolute Error	Relative Error
λ_1	5.5298e-10	1.2337e-10
λ_2	9.9956e-07	4.1264e-08
λ_3	5.6104e-11	8.8043e-13
λ_4	3.8774e-10	3.1516e-12
λ_5	5.5616e-10	2.7505e-12

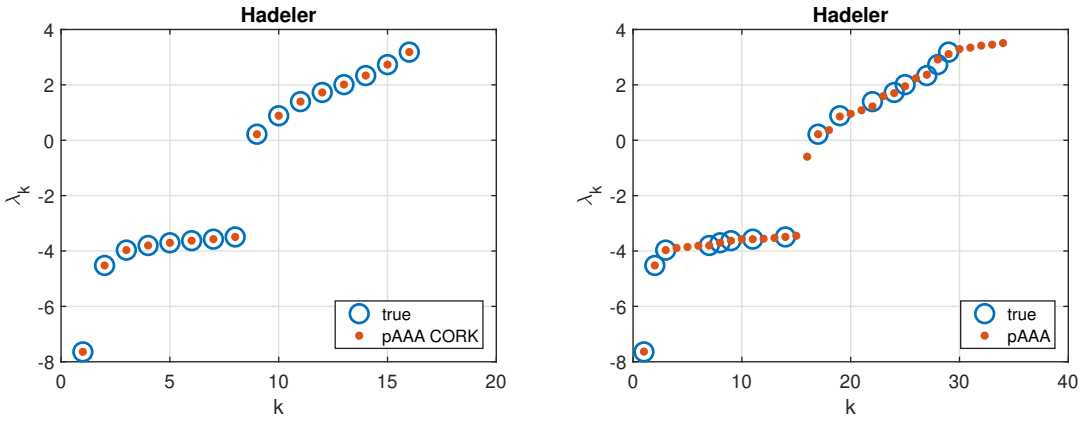
$p = 1$ and $\lambda = \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$ are not part of the p-AAA samples but they do fall inside the intervals sampled. This is crucial for the performance of this method. This method does not aim to find all (potentially infinitely many) eigenvalues. Instead our method aims to find the eigenvalues in a certain region (the same way as the nonparametric method in [92]). Hence the samples used to build the p-AAA approximation need to be in the intervals of interest where one wants to find eigenvalues. This will be our sampling approach in the following examples as well where the nonlinearities $g(\lambda, p)$ are not rational.

3.3.2 Hadeler model

Consider the following problem from the NLEVP collection [34, 74]:

$$\mathbf{T}(\lambda, p) = \left(e^{p\lambda} - 1\right)\mathbf{A}_2 + \lambda^2\mathbf{A}_1 - \alpha\mathbf{A}_0,$$

where $\mathbf{A}_2, \mathbf{A}_1, \mathbf{A}_0 \in \mathbb{R}^{n\times n}$ are symmetric, $n = 100$, $\alpha = 100$ is a scalar, and we introduce the parameter p (whose original value is $p = 1$). We apply p-AAA to approximate $g(\lambda, p) = e^{p\lambda} - 1$ using $N = 100$ linearly spaced samples in $\lambda \in [-4, 4]$ and $M = 6$ linearly spaced samples in $p \in [0.1, 2]$, with tolerance `1e-5`. The result is an approximation of order $(k, q) = (6, 3)$. For the eigenvalue approximation, we evaluate the p-AAA approximation at $p = 1$ and apply the CORK method from [92] for the corresponding linearization (3.9). In Figure 3.2a, we compare our approximation (in orange dots) with the standard CORK method [92] for the nonparametric problem with $p = 1$ in the interval $\lambda \in [-8, 4]$. We see that the approximation is accurate even for the values slightly outside $[-8, -4]$ of the interval sampled $[-4, 4]$. Both the maximum absolute and relative error is in hte order of 10^{-3}.



(a) p-AAA with CORK. (b) p-AAA.

Figure 3.2: Eigenvalues for the Hadeler model.

Note that one could apply our p-AAA algorithm to solve the PNLEVP in a way different

from the method we have discussed so far. Since the eigenvalues of $\mathbf{T}(\lambda, p)$ are the poles of $\mathbf{T}(\lambda, p)^{-1}$, consider approximating $\mathbf{T}(\lambda, p)^{-1}$ via our MIMO p-AAA in Section 2.3

$$\mathbf{T}(\lambda, p)^{-1} \approx \widetilde{\mathbf{H}}(\lambda, p).$$

Then one might expect that, for a specific $p = \pi$, the poles of $\widetilde{\mathbf{H}}(\lambda, \pi)$ are a good approximation of the eigenvalues of $\mathbf{T}(\lambda, p)$. We show that this may not be the case. In Figure 3.2b, we see that some of the eigenvalues are captured. However there are also poles that do not correspond to any eigenvalues, hence it would be a mistake to consider them as approximations to eigenvalues. Note also that this PNLEVP has infinitely many eigenvalues, while any rational approximation such as p-AAA has finitely many poles that depend greatly on where the samples are taken.

3.3.3 Sandwich beam

We parametrize an example from [92] that models a beam with a damping layer between two steel layers. The model has the form

$$\mathbf{T}(\lambda, p) = \mathbf{K} - \lambda^2 \mathbf{M} + \underbrace{\frac{G_0 + G_\infty (\iota \lambda p)^\alpha}{1 + (\iota \lambda p)^\alpha}}_{g(\lambda, p)} \mathbf{C},$$

where $\mathbf{K}, \mathbf{M}, \mathbf{C} \in \mathbb{R}^{168 \times 168}$, $G_0 = 3.504 \times 10^5$, $G_\infty = 3.062 \times 10^9$, and $\alpha = 0.675$. We approximate $g(\lambda, p)$ with p-AAA using $N = 10000$ and $M = 3$ linearly spaced samples in $\lambda \in [200, 30000]$ and $p \in [7, 9] \times 10^{-9}$, and stopping tolerance 10^{-12}. The result is an approximation of order $(k, q) = (12, 2)$. We evaluate the approximation at $p = 8.23 \times 10^{-9}$ and apply the CORK algorithm. Figure 3.3 shows our approximations (orange dots) of the eigenvalues to the approximation in [92] (blue circles). The maximum absolute error is

4.5174×10^{-1} and the maximum relative error is 7.2826×10^{-5}. Note that for this example

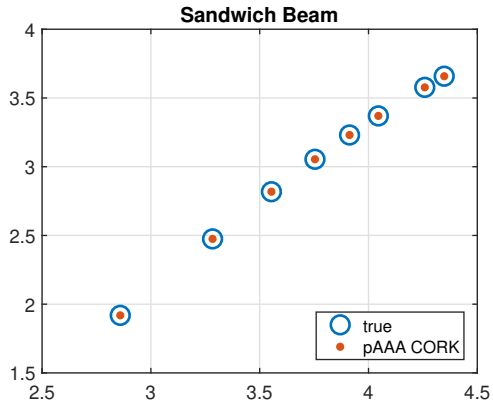


Figure 3.3: Eigenvalues of Example 3.3.3

the parameter interval sampled is very small. This is indeed necessary, since the parameter p is the relaxation time measured in nano-seconds, and hence its values are of order 10^{-9}. When this is overlooked, our method does not provide good eigenvalue approximations. We check that our approach provides good eigenvalue approximation for other parameter values. For the same p-AAA approximation described above and shown in Figure 3.3, we find the eigenvalues in the same region for various parameter values, and we check the condition number of the corresponding $\mathbf{T}(\lambda, p)$. For $p = 6 \times 10^{-9}, 7.5 \times 10^{-9}, 8.5 \times 10^{-9}, 10 \times 10^{-9}$, the condition number is in the order of 10^{11}, meaning $\mathbf{T}(\lambda, p)$ is close to being rank deficient, as desired, and the residual norms (using the definition in [92])

$$\frac{\|\mathbf{T}(\lambda, p)\mathbf{x}\|_2}{\|\mathbf{T}(\lambda, p)\|_1 \|\mathbf{x}\|_2}$$

are in the order of 10^{-11}.

3.4 Summary

We have extended the method in [92] from NLEVP to PNLEVP. We have defined an analogous CORK linearization to include the p-AAA approximation of the nonlinearities in the eigenvalue problem. We have shown the accuracy of this linearization in three numerical examples. Our method adds to the efficiency of [92]: instead of running AAA for each parameter of interest, we find a parametric approximation with p-AAA and only repeat the CORK approximation for parameter reevaluations.

Chapter 4

Parametric Bilinear Interpolatory Model Reduction

In this chapter, we present the final contribution of this dissertation: the extension of interpolatory model reduction to parametric bilinear dynamical systems. For the nonparametric case, interpolatory projection methods have been already extended from linear dynamical systems to bilinear dynamical systems. We develop here the analysis and methodology for the parametric case. We introduce necessary conditions that the projection subspaces must satisfy to obtain parametric tangential interpolation of each subsystem transfer function, their parameter sensitivities (Jacobian), and their parameter Hessian. As in the parametric linear case, the basis construction does not require computing the Jacobian or the Hessian.

The chapter is organized as follows. In Section 4.1, we define parametric bilinear dynamical systems and introduce the problem description. In Section 4.2, we present our main theoretical results. Section 4.3 illustrates the theory using three numerical examples. This chapter follows very closely our published work [42]. Recent work [23] has extended these results to the more general case of polynomial parametric systems.

4.1 Problem Description

In this chapter, we will focus on large-scale bilinear systems parametrized with the parameter vector $\mathbf{p} \in \mathcal{P} \subseteq \mathbb{R}^{n_{par}}$ and represented in the state-space form

$$\left\{ \begin{array}{rcl} \mathbf{E}(\mathbf{p})\dot{\mathbf{x}}(t;\mathbf{p}) & = & \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \displaystyle\sum_{j=1}^{n_{in}} \mathbf{N}_j(\mathbf{p})\mathbf{x}(t;\mathbf{p})u_j(t) + \mathbf{B}(\mathbf{p})\mathbf{u}(t), \\ \mathbf{y}(t;\mathbf{p}) & = & \mathbf{C}(\mathbf{p})\mathbf{x}(t;\mathbf{p}), \end{array} \right. \tag{4.1}$$

where $\mathbf{x}(t;\mathbf{p}) \in \mathbb{R}^{n_{st}}$, $\mathbf{y}(t;\mathbf{p}) \in \mathbb{R}^{n_{out}}$, and $\mathbf{u}(t) = [u_1(t),\ u_2(t), \ldots,\ u_m(t)]^\top \in \mathbb{R}^{n_{in}}$ denote the states, outputs (measurements/quantities of interest), and inputs (excitation/forcing) of the bilinear dynamical system, respectively. Thus the corresponding state-space matrices have the dimensions $\mathbf{E}(\mathbf{p}), \mathbf{A}(\mathbf{p}), \mathbf{N}_j(\mathbf{p}) \in \mathbb{R}^{n_{st} \times n_{st}}$ for $j = 1, \ldots, n_{in}$, $\mathbf{B}(\mathbf{p}) \in \mathbb{R}^{n_{st} \times n_{in}}$, and $\mathbf{C}(\mathbf{p}) \in \mathbb{R}^{n_{out} \times n_{st}}$. We assume that the matrix $\mathbf{E}(\mathbf{p})$ is nonsingular for every parameter value $\mathbf{p} \in \mathcal{P}$. Bilinear systems of the form (4.1) appear in a variety of applications, such as the study of biological species and nuclear fission, are used in the context of stochastic control problems, and frequently appear in modeling nonlinear phenomena of small magnitude; see, for instance, [21, 28, 75, 99, 101, 116, 127]. We are interested in large-scale settings where simulating/solving (4.1) for a wide variety of inputs $\mathbf{u}(t)$ and parameters $\mathbf{p}$ solely to determine the output $\mathbf{y}(t;\mathbf{p})$ is too expensive. Therefore, our goal is to construct a reduced parametric bilinear system of order $\widetilde{n}_{st} \ll n_{st}$ in state-space form

$$\left\{ \begin{array}{rcl} \widetilde{\mathbf{E}}(\mathbf{p})\dot{\widetilde{\mathbf{x}}}(t;\mathbf{p}) & = & \widetilde{\mathbf{A}}(\mathbf{p})\widetilde{\mathbf{x}}(t;\mathbf{p}) + \displaystyle\sum_{j=1}^{n_{in}} \widetilde{\mathbf{N}}_j(\mathbf{p})\widetilde{\mathbf{x}}(t;\mathbf{p})u_j(t) + \widetilde{\mathbf{B}}(\mathbf{p})\mathbf{u}(t), \\ \widetilde{\mathbf{y}}(t;\mathbf{p}) & = & \widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{x}}(t;\mathbf{p}), \end{array} \right. \tag{4.2}$$

where $\widetilde{\mathbf{E}}(\mathbf{p}), \widetilde{\mathbf{A}}(\mathbf{p}), \widetilde{\mathbf{N}}_j(\mathbf{p}) \in \mathbb{R}^{\widetilde{n}_{st} \times \widetilde{n}_{st}}$, for $j = 1, \ldots, n_{in}$, $\widetilde{\mathbf{B}}(\mathbf{p}) \in \mathbb{R}^{\widetilde{n}_{st} \times n_{in}}$, and $\widetilde{\mathbf{C}}(\mathbf{p}) \in \mathbb{R}^{n_{out} \times \widetilde{n}_{st}}$ such that the reduced output $\widetilde{\mathbf{y}}(t;\mathbf{p})$ provides a good approximation to the original

output $\mathbf{y}(t;\mathbf{p})$ for a variety of inputs $\mathbf{u}(t)$ and a range of parameters $\mathbf{p}$.

Non-parametric bilinear systems (4.1) where the state-space matrices $\mathbf{E}$, $\mathbf{A}$, $\{\mathbf{N}_j\}_{j=1}^{n_{in}}$, $\mathbf{B}$, and $\mathbf{C}$ are constant, have been studied thoroughly, and input-independent optimal model reduction techniques from the linear case ($\mathbf{N}_j = 0$ for $j = 1, \ldots, n_{in}$) have been successfully generalized to non-parametric bilinear systems. For example, [1, 14, 25, 37, 113] have extended model reduction via rational interpolation [9, 18] from linear to non-parametric bilinear systems. The optimal model reduction of linear dynamical systems in the $\mathcal{H}_2$ norm[1] via the iterative rational Krylov algorithm (IRKA) [66] has been generalized to bilinear systems via bilinear IRKA (B-IRKA) [20]. Later, [57] showed that, as with IRKA and $\mathcal{H}_2$ model reduction in the linear case, the reduced model via B-IRKA also satisfies Hermite interpolation in this case, but in the sense of Volterra series interpolation. Similarly, gramians and balanced truncation[2] (BT) for linear dynamical systems [102, 103] have also been generalized to nonparametric bilinear systems [3, 21, 63, 75]. Structure-preserving interpolatory MOR has also been extended to (nonparametric) bilinear systems [30]. Moreover, [11] has applied the Loewner framework [96] to bilinear systems. Unlike the extensions of interpolation theory and IRKA to *non-parametric* bilinear systems, interpolatory methods had not yet been generalized to parametric bilinear systems until our work [42], which forms the foundation of this chapter.

In the remainder of this section, we introduce the ingredients of the model reduction prob-

[1]The $\mathcal{H}_2$ error norm is an input-independent least-squares error measure in time or frequency domain. For linear systems (4.5), the $\mathcal{H}_2$ norm is given by $\left(\frac{1}{2\pi}\int_{-\infty}^{\infty} \|\mathbf{H}(\imath\omega)\|_F^2 d\omega\right)^{1/2}$ where $\mathbf{H}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}$ is the transfer function and $\|\cdot\|_F$ denotes the Frobenius norm. It has been shown that optimality in the $\mathcal{H}_2$ error measure requires Hermite interpolation of the transfer function $\mathbf{H}(s)$. For details, we refer the reader to [66].

[2]For linear dynamical systems, the symmetric positive semidefinite matrices $\mathbf{G}_R$ and $\mathbf{G}_O$ that solve the generalized Lyapunov equations $\mathbf{A}\mathbf{G}_R\mathbf{E}^\top + \mathbf{E}\mathbf{G}_R\mathbf{A}^\top + \mathbf{B}\mathbf{B}^\top = 0$ and $\mathbf{A}^\top\mathbf{G}_O\mathbf{E} + \mathbf{E}^\top\mathbf{G}_O\mathbf{A} + \mathbf{C}^\top\mathbf{C} = 0$ are called the reachability and observability gramians, respectively. Balanced truncation corresponds to performing a truncation in a basis where $\mathbf{G}_R = \mathbf{G}_O$ are diagonal, with nonnegative diagonal entries called the Hankel singular values. For details, we refer the reader to [102, 103].

lem for parametric bilinear systems, such as projection, subsystem transfer functions, and tangential interpolation.

4.1.1 Projection-based model reduction of parametric bilinear systems via global basis

We construct the reduced parametric bilinear system (4.2) via projection. We follow the *global basis approach* (as opposed to using a local basis and performing extrapolation [76, 130] or interpolation [4, 46, 129]). Thus we construct two constant global model reduction bases, namely $\mathbf{V} \in \mathbb{C}^{n_{st} \times \widetilde{n}_{st}}$ and $\mathbf{W} \in \mathbb{C}^{n_{st} \times \widetilde{n}_{st}}$, that capture the parametric dependence of the underlying system using the information from various sampling points. We refer the reader to [26] for detailed explanations regarding global and local bases, and different sampling options. The matrices $\mathbf{V}$ and $\mathbf{W}$ are computed to enforce specific interpolation conditions as we will discuss in Section 4.2.

Once the model reduction matrices $\mathbf{V}$ and $\mathbf{W}$ are constructed, the reduced model quantities in (4.2) are obtained via Petrov-Galerkin projection:

$$\begin{aligned} &\widetilde{\mathbf{E}}(\mathbf{p}) = \mathbf{W}^\top \mathbf{E}(\mathbf{p})\mathbf{V}, \qquad \widetilde{\mathbf{N}}_j(\mathbf{p}) = \mathbf{W}^\top \mathbf{N}_j(\mathbf{p})\mathbf{V} \quad \text{for } j = 1, \ldots, n_{in}, \\ &\widetilde{\mathbf{A}}(\mathbf{p}) = \mathbf{W}^\top \mathbf{A}(\mathbf{p})\mathbf{V}, \qquad \widetilde{\mathbf{B}}(\mathbf{p}) = \mathbf{W}^\top \mathbf{B}(\mathbf{p}), \text{ and } \qquad \widetilde{\mathbf{C}}(\mathbf{p}) = \mathbf{C}(\mathbf{p})\mathbf{V}. \end{aligned} \tag{4.3}$$

Now consider reevaluating reduced model quantities in (4.3) for a new parameter value $\hat{\boldsymbol{\pi}} \in \mathcal{P}$[3]. Consider the case of $\widetilde{\mathbf{E}}(\hat{\boldsymbol{\pi}})$. This will require reevaluating $\widetilde{\mathbf{E}}(\hat{\boldsymbol{\pi}}) = \mathbf{W}^\top \mathbf{E}(\hat{\boldsymbol{\pi}})\mathbf{V}$ where the operations depend on the original system dimension n_{st}. In practice, many problems exhibit an affine parametric structure, which makes the projection step numerically efficient.

[3]In general we will consider the following notation: $\mathbf{p}$ (s) denotes the parameter (frequency) variable; $\boldsymbol{\pi}$ (σ) denotes the parameter (frequency) values where interpolation is enforced; and $\hat{\boldsymbol{\pi}}$ ($\hat{\sigma}$) denotes parameter (frequency) values at which the model is tested or evaluated.

For simplicity, continue to consider the matrix $\mathbf{E}(\mathbf{p})$ only. Assume that $\mathbf{E}(\mathbf{p})$ has the affine parametric form

$$\mathbf{E}(\mathbf{p}) = \mathbf{E}_0 + \sum_{k=1}^{K} f_k(\mathbf{p})\mathbf{E}_k, \tag{4.4}$$

where $f_k : \mathbb{R}^{n_{par}} \to \mathbb{R}$ are scalar (nonlinear) functions reflecting parametric dependency and $\mathbf{E}_k \in \mathbb{R}^{n_{st} \times n_{st}}$ for $k = 1, \ldots, K$ are constant matrices. Then, the reduced matrix $\widetilde{\mathbf{E}}(\mathbf{p})$ in (4.3) is given by

$$\widetilde{\mathbf{E}}(\mathbf{p}) = \mathbf{W}^\top \mathbf{E}(\mathbf{p})\mathbf{V} = \mathbf{W}^\top \mathbf{E}_0 \mathbf{V} + \sum_{k=1}^{K} f_k(\mathbf{p})\mathbf{W}^\top \mathbf{E}_k \mathbf{V},$$

where $\mathbf{W}^\top \mathbf{E}_i \mathbf{V}$, for $i = 0, \ldots, K$, have to be computed once in the offline phase, and then can be recombined for efficient computation of $\widetilde{\mathbf{E}}(\hat{\boldsymbol{\pi}})$ in the online phase. The same discussion applies to other matrices in (4.3) as well. When $\mathbf{E}(\mathbf{p})$ does not admit such an affine parametrization as in (4.4), one usually performs an affine approximation of $\mathbf{E}(\mathbf{p})$ first, usually via a matrix version of the discrete empirical interpolation method DEIM [43, 64]. We discuss details in Section 4.1.2 next and will revisit this issue in the second numerical example in Section 4.3.2.

4.1.2 Projection of non-affine matrices

We present here the DEIM method that we will implement in Section 4.3.2. In order to match the situation that will arise in Section 4.3.2, we consider the projection (4.3) of the input matrix of a SISO system, i.e., consider computing

$$\widetilde{\mathbf{b}}(\mathbf{p}) = \mathbf{W}^\top \mathbf{b}(\mathbf{p}),$$

where $\mathbf{b}(\mathbf{p}) \in \mathbb{C}^{n_{st}}$ is an arbitrary column vector that depends on a parameter $\mathbf{p} \in \mathbb{R}^{n_{par}}$ and $\mathbf{W} \in \mathbb{C}^{n_{st} \times \widetilde{n}_{st}}$ is a basis matrix with $\widetilde{n}_{st} \ll n_{st}$, hence making $\widetilde{\mathbf{b}}(\mathbf{p}) \in \mathbb{C}^{\widetilde{n}_{st}}$. This operation should allow for an efficient re-evaluation of $\widetilde{\mathbf{b}}(\mathbf{p})$ for any parameter values of interest, since the dimension of $\widetilde{\mathbf{b}}(\mathbf{p})$ is much smaller than the dimension of $\mathbf{b}(\mathbf{p})$. However, if there is no structure in $\mathbf{b}(\mathbf{p})$ to take advantage of (such as the affine structure discussed above) then computing $\widetilde{\mathbf{b}}(\mathbf{p})$ may be as costly, since $\mathbf{b}(\mathbf{p})$ would need to be computed first for every $\mathbf{p}$ at $O(n_{st})$ floating point operations. In order to mitigate this, we employ the Q-DEIM algorithm [47], which is a revised implementation of the discrete empirical interpolation method (DEIM) [43].

In this case, since $\mathbf{b}(\mathbf{p})$ is a vector, there is no need for a matrix-version and the original DEIM formulation suffices. To apply DEIM, we want to find a basis $\mathbf{U} \in \mathbb{R}^{n_{st} \times M}$ where $M \ll n_{st}$ and a row selector $\mathbb{S}$ so that

$$\mathbf{b}(\mathbf{p}) \approx \mathbf{U}(\mathbb{S}^\top \mathbf{U})^{-1} \mathbb{S}^\top \mathbf{b}(\mathbf{p})$$

is a good approximation, and hence so is

$$\widetilde{\mathbf{b}}(\mathbf{p}) \approx \mathbf{W}^\top \mathbf{U}(\mathbb{S}^\top \mathbf{U})^{-1} \mathbb{S}^\top \mathbf{b}(\mathbf{p}).$$

In this way $\mathbf{W}^\top \mathbf{U}(\mathbb{S}^\top \mathbf{U})^{-1}$ can be precomputed offline, while the online computation of $\mathbb{S}^\top \mathbf{b}(\mathbf{p})$ will now only require us to compute the M entries in $\mathbf{b}(\mathbf{p})$ indicated by $\mathbb{S}$. Clearly the accuracy of this approximation depends on $\mathbf{U}$ and $\mathbb{S}$.

We first find $\mathbf{U}$ via proper orthogonal decomposition (POD) [31, 93] [4]. That is, we generate

[4] POD is a widely used technique to build the basis $\mathbf{V}$ for projection MOR. In particular, it has become the preferred MOR technique for modeling turbulent flows [126]. In a nutshell, the POD basis $\mathbf{V} = \mathbf{W}$ captures as much of the system's energy, for a specific input, as possible for any basis with its same dimension $n_{st} \times \widetilde{n}_{st}$.

a matrix of snapshots of the vector $\mathbf{b}(\mathbf{p})$

$$\mathbb{B} = [\mathbf{b}(\mathbf{p}_1)\ \mathbf{b}(\mathbf{p}_2)\ \cdots\ \mathbf{b}(\mathbf{p}_N)]\,,$$

we compute its SVD

$$\mathbb{B} = \mathcal{U}\Sigma\mathcal{V}^*,$$

and select the leading M left singular vectors to be the columns of $\mathbf{U}$, which using MATLAB notation means

$$\mathbf{U} = \mathcal{U}(:, 1:M).$$

By taking *enough* snapshots and singular vectors, we expect the range of our basis $\mathbf{U}$ to represent the values of $\mathbf{b}(\mathbf{p})$ over the parameter domain. This is justified by the fact that $\mathbf{U}$ is the LS optimal M-dimensional subspace to approximate the snapshots.

Now, to choose the interpolation indices (row selector) in $\mathbb{S}$, we use the Q-DEIM algorithm [47], which determines $\mathbb{S}$ using a pivoted QR factorization of $\mathbf{U}^*$. The original DEIM method chooses the row selection matrix $\mathbb{S}$ so that $\|(\mathbb{S}^\top\mathbf{U})^{-1}\|_2$ is small via pivoted LU factorization of $\mathbf{U}$. This is motivated by the following theoretical upper bound on the approximation error:

$$\left\|\mathbf{b}(\mathbf{p}) - \mathbf{U}(\mathbb{S}^\top\mathbf{U})^{-1}\mathbb{S}^\top\mathbf{b}(\mathbf{p})\right\|_2 \leq \left\|(\mathbb{S}^\top\mathbf{U})^{-1}\right\|_2 \|(\mathbf{I} - \mathbf{U}\mathbf{U}^*)\mathbf{b}(\mathbf{p})\|_2\,.$$

For variations and extensions to DEIM, see, for example, DEIM-CUR [121], adaptive DEIM [108, 122], localized DEIM [110], and [111] studying DEIM in the noisy case. The one we choose to implement, the Q-DEIM implementation, consists of a QR factorization with column pivoting of $\mathbf{U}^*$.

4.1.3 Interpolatory projections for parametric linear systems

A powerful framework in the case of linear dynamical systems, i.e.,

$$\begin{cases} \mathbf{E}(\mathbf{p})\dot{\mathbf{x}}(t;\mathbf{p}) &= \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \mathbf{B}(\mathbf{p})\mathbf{u}(t), \\ \mathbf{y}(t;\mathbf{p}) &= \mathbf{C}(\mathbf{p})\mathbf{x}(t;\mathbf{p}), \end{cases} \tag{4.5}$$

is to transform the problem into the frequency domain via Laplace transform. To do so, let $\mathbf{Y}(s;\mathbf{p})$ and $\mathbf{U}(s)$ denote the Laplace transforms of $\mathbf{y}(t;\mathbf{p})$ and $\mathbf{u}(t)$, respectively. Then, applying the Laplace transform to (4.5) leads to

$$\mathbf{Y}(s;\mathbf{p}) = \mathbf{H}(s;\mathbf{p})\mathbf{U}(s), \quad \text{where} \quad \mathbf{H}(s;\mathbf{p}) = \mathbf{C}(\mathbf{p})\left(s\,\mathbf{E}(\mathbf{p}) - \mathbf{A}(\mathbf{p})\right)^{-1}\mathbf{B}(\mathbf{p})$$

is the transfer function of (4.5). Then, the goal is to construct a reduced parametric linear model

$$\begin{cases} \widetilde{\mathbf{E}}(\mathbf{p})\dot{\widetilde{\mathbf{x}}}(t;\mathbf{p}) &= \widetilde{\mathbf{A}}(\mathbf{p})\widetilde{\mathbf{x}}(t;\mathbf{p}) + \widetilde{\mathbf{B}}(\mathbf{p})\mathbf{u}(t), \\ \widetilde{\mathbf{y}}(t;\mathbf{p}) &= \widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{x}}(t;\mathbf{p}), \end{cases} \tag{4.6}$$

whose reduced parametric transfer function

$$\widetilde{\mathbf{H}}(s;\mathbf{p}) = \widetilde{\mathbf{C}}(\mathbf{p})\left(s\,\widetilde{\mathbf{E}}(\mathbf{p}) - \widetilde{\mathbf{A}}(\mathbf{p})\right)^{-1}\widetilde{\mathbf{B}}(\mathbf{p})$$

approximates $\mathbf{H}(s;\mathbf{p})$ well. This would in turn imply $\widetilde{\mathbf{y}}(t;\mathbf{p}) \approx \mathbf{y}(t;\mathbf{p})$, since

$$\mathbf{Y}(s;\mathbf{p}) - \widetilde{\mathbf{Y}}(s;\mathbf{p}) = (\mathbf{H}(s;\mathbf{p}) - \widetilde{\mathbf{H}}(s;\mathbf{p}))\mathbf{U}(s).$$

One way to enforce $\widetilde{\mathbf{H}}(s;\mathbf{p}) \approx \mathbf{H}(s;\mathbf{p})$ is via rational interpolation: Given the frequency interpolation points $\{\sigma_i\} \subset \mathbb{C}$, the right tangential directions $\{\mathbf{r}_i\} \subset \mathbb{C}^{n_{in}}$, the left tangential

directions $\{\boldsymbol{\ell}_i\} \subset \mathbb{C}^{n_{out}}$, and the parameter interpolation samples $\{\boldsymbol{\pi}_j\} \subset \mathcal{P}$, find a reduced model (4.6) such that $\widetilde{\mathbf{H}}(s;\mathbf{p})$ is a Hermite tangential interpolant to $\mathbf{H}(s;\mathbf{p})$ at the selected samples, i.e.,

$$\begin{aligned} \mathbf{H}(\sigma_i;\boldsymbol{\pi}_j)\mathbf{r}_i &= \widetilde{\mathbf{H}}(\sigma_i;\boldsymbol{\pi}_j)\mathbf{r}_i, & \frac{\partial}{\partial s}\left(\boldsymbol{\ell}_i^\top \mathbf{H}(\sigma_i;\boldsymbol{\pi}_j)\mathbf{r}_i\right) &= \frac{\partial}{\partial s}\left(\boldsymbol{\ell}_i^\top \widetilde{\mathbf{H}}(\sigma_i;\boldsymbol{\pi}_j)\mathbf{r}_i\right), \\ \boldsymbol{\ell}_i^\top \mathbf{H}(\sigma_i;\boldsymbol{\pi}_j) &= \boldsymbol{\ell}_i^\top \widetilde{\mathbf{H}}(\sigma_i;\boldsymbol{\pi}_j), \text{ and} & \nabla_{\mathbf{p}}\left(\boldsymbol{\ell}_i^\top \mathbf{H}(\sigma_i;\boldsymbol{\pi}_j)\mathbf{r}_i\right) &= \nabla_{\mathbf{p}}\left(\boldsymbol{\ell}_i^\top \widetilde{\mathbf{H}}(\sigma_i;\boldsymbol{\pi}_j)\mathbf{r}_i\right). \end{aligned}$$

In other words, the reduced model tangentially matches transfer function values, in addition to its frequency and parametric derivatives, at the sampled points. One can impose higher order interpolation conditions at the frequency and parameter samples as well, such as the parameter Hessian. We omit it for brevity here. The following result from [15] shows how to construct model reduction matrices $\mathbf{V}$ and $\mathbf{W}$ that satisfy the desired interpolation conditions for a particular frequency, directions, and parameter choice $(\sigma, \mathbf{r}, \boldsymbol{\ell}, \boldsymbol{\pi})$. To simplify notation here and in the bilinear case later, we define

$$\mathbf{K}(s;\mathbf{p}) = (s\mathbf{E}(\mathbf{p}) - \mathbf{A}(\mathbf{p}))^{-1} \quad \text{and} \quad \widetilde{\mathbf{K}}(s;\mathbf{p}) = \left(s\widetilde{\mathbf{E}}(\mathbf{p}) - \widetilde{\mathbf{A}}(\mathbf{p})\right)^{-1}. \tag{4.7}$$

Then the linear transfer functions can be rewritten as

$$\mathbf{H}(s;\mathbf{p}) = \mathbf{C}(\mathbf{p})\mathbf{K}(s;\mathbf{p})\mathbf{B}(\mathbf{p}) \quad \text{and} \quad \widetilde{\mathbf{H}}(s;\mathbf{p}) = \widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{K}}(s;\mathbf{p})\widetilde{\mathbf{B}}(\mathbf{p}). \tag{4.8}$$

Theorem 4.1. *Given* $\mathbf{H}(s;\mathbf{p}) = \mathbf{C}(\mathbf{p})\mathbf{K}(s;\mathbf{p})\mathbf{B}(\mathbf{p})$, *let* $\widetilde{\mathbf{H}}(s;\mathbf{p}) = \widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{K}}(s;\mathbf{p})\widetilde{\mathbf{B}}(\mathbf{p})$ *be obtained via Petrov-Galerkin projection using the matrices* $\mathbf{V}$ *and* $\mathbf{W}$. *Let* $\sigma \in \mathbb{C}$, $\boldsymbol{\pi} \in \mathcal{P}$, $\mathbf{r} \in \mathbb{C}^{n_{in}} \setminus \{\mathbf{0}\}$, *and* $\boldsymbol{\ell} \in \mathbb{C}^{n_{out}} \setminus \{\mathbf{0}\}$. *Assume* $\mathbf{K}(\sigma, \boldsymbol{\pi})$ *and* $\widetilde{\mathbf{K}}(\sigma;\boldsymbol{\pi})$ *exist.*

(a) If $\mathbf{K}(\sigma,\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r} \in \mathrm{Ran}(\mathbf{V})$*, then*

$$\mathbf{H}(\sigma;\boldsymbol{\pi})\mathbf{r} = \widetilde{\mathbf{H}}(\sigma;\boldsymbol{\pi})\mathbf{r};$$

(b) If $\mathbf{K}(\sigma,\boldsymbol{\pi})^\top\mathbf{C}(\boldsymbol{\pi})^\top\boldsymbol{\ell} \in \mathrm{Ran}(\mathbf{W})$*, then*

$$\boldsymbol{\ell}^\top\mathbf{H}(\sigma;\boldsymbol{\pi}) = \boldsymbol{\ell}^\top\widetilde{\mathbf{H}}(\sigma;\boldsymbol{\pi});$$

(c) If both (a) and (b) hold simultaneously, then

$$\frac{\partial}{\partial s}\left(\boldsymbol{\ell}^\top\mathbf{H}(\sigma;\boldsymbol{\pi})\mathbf{r}\right) = \frac{\partial}{\partial s}\left(\boldsymbol{\ell}^\top\widetilde{\mathbf{H}}(\sigma;\boldsymbol{\pi})\mathbf{r}\right) \quad \textit{and} \quad \nabla_{\mathbf{p}}\left(\boldsymbol{\ell}^\top\mathbf{H}(\sigma;\boldsymbol{\pi})\mathbf{r}\right) = \nabla_{\mathbf{p}}\left(\boldsymbol{\ell}^\top\widetilde{\mathbf{H}}(\sigma;\boldsymbol{\pi})\mathbf{r}\right).$$

Theorem 4.1 shows that in order to guarantee tangential interpolation of the transfer function and its first derivatives, it suffices to choose the columns of $\mathbf{V}$ and $\mathbf{W}$ as the vectors containing the right and left tangential information of the transfer function. See Figure 4.1 for a more

$$\begin{aligned}\boldsymbol{\ell}^\top\mathbf{H}(\sigma,\boldsymbol{\pi})\mathbf{r} &= \boldsymbol{\ell}^\top\mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma;\mathbf{p})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}\\ &= \underbrace{\boldsymbol{\ell}^\top\mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma;\boldsymbol{\pi})}_{\mathbf{W}^\top}\mathbf{B}(\boldsymbol{\pi})\mathbf{r}\\ &= \boldsymbol{\ell}^\top\mathbf{C}(\boldsymbol{\pi})\underbrace{\mathbf{K}(\sigma;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{V}}\end{aligned}$$

Figure 4.1: Visualization of the linear MOR bases in Theorem 4.1. We consider the scalar $\boldsymbol{\ell}^\top\mathbf{H}(\sigma,\pi)\mathbf{r}$ instead of each tangential direction separately for simplicity of presentation.

visual representation of this statement, where we highlight in blue the information needed in the range of $\mathbf{W}$ to guarantee left interpolation, and in red the information needed in the range of $\mathbf{V}$ to guarantee right interpolation. This color scheme will remain constant throughout the figures in this chapter meant to illustrate transfer function interpolation:

red for $\mathbf{V}$ and right interpolation; and blue for $\mathbf{W}$ and left interpolation. We do not provide a detailed proof of Theorem 4.1 here since one is already in [15], and we will prove a more general result in Theorems 4.2, 4.4, and 4.5. Instead, we provide a sketch of the part of the proof in Figure 4.2 in order to build the reader's intuition and familiarity with the kinds of manipulations that we will use in the proofs of our parametric bilinear results in Section 4.2. In Figure 4.2, we consider the error term between the original and the reduced transfer functions along both left and right directions. Then the proof consists of showing that this difference is zero. First, we use the definition (4.3) of the projected input and output matrices ($\widetilde{\mathbf{B}}(\mathbf{p})$ and $\widetilde{\mathbf{C}}(\mathbf{p})$) in order to expose common terms in both the full and reduced parts of the error expression. If one wants to use the assumptions on $\mathbf{V}$, then we rewrite the error term by factoring out to the right (in red) the vector included in the range of $\mathbf{V}$, by assumption. This is the same vector highlighted in red in Figure 4.1. We include the last line in Figure 4.2 to show the factorization needed in case we wanted to use the information in $\mathbf{W}$ instead. Then the details that we skip here would be to show that the null space of the expression in parenthesis in the last two lines contains the corresponding highlighted vectors, hence making the error term zero, as desired.

$$
\begin{aligned}
\boldsymbol{\ell}^\top \mathbf{H}(\sigma,\boldsymbol{\pi})\mathbf{r} - \boldsymbol{\ell}^\top \widetilde{\mathbf{H}}(\sigma,\boldsymbol{\pi})\mathbf{r} &= \boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r} - \boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\boldsymbol{\pi}) \quad \widetilde{\mathbf{K}}(\sigma;\boldsymbol{\pi}) \quad \widetilde{\mathbf{B}}(\boldsymbol{\pi})\mathbf{r} \\
&= \boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r} - \boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{V}\widetilde{\mathbf{K}}(\sigma;\boldsymbol{\pi})\mathbf{W}^\top \mathbf{B}(\boldsymbol{\pi})\mathbf{r} \\
&= \boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi}) \left(\mathbf{K}(\sigma;\boldsymbol{\pi}) - \mathbf{V}\widetilde{\mathbf{K}}(\sigma;\boldsymbol{\pi})\mathbf{W}^\top\right) \mathbf{B}(\boldsymbol{\pi})\mathbf{r} \\
&= \boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi}) \underbrace{\left(\mathbf{I} - \mathbf{V}\widetilde{\mathbf{K}}(\sigma;\boldsymbol{\pi})\mathbf{W}^\top \mathbf{K}^{-1}(\sigma;\boldsymbol{\pi})\right) \mathbf{K}(\sigma;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{0}} \\
&= \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma;\boldsymbol{\pi}) \left(\mathbf{I} - \mathbf{K}^{-1}(\sigma;\boldsymbol{\pi})\mathbf{V}\widetilde{\mathbf{K}}(\sigma;\boldsymbol{\pi})\mathbf{W}^\top\right)}_{\mathbf{0}} \mathbf{B}(\boldsymbol{\pi})\mathbf{r}
\end{aligned}
$$

Figure 4.2: Sketch of the proof for Theorem 4.1. We consider the scalar $\boldsymbol{\ell}^\top \mathbf{H}(\sigma,\pi)\mathbf{r}$ instead of each tangential direction separately for simplicity of presentation.

4.1.4 Interpolatory parametric bilinear model reduction problem

Reconsider the full-order parametric bilinear system in (4.1):

$$\begin{cases} \mathbf{E}(\mathbf{p})\dot{\mathbf{x}}(t;\mathbf{p}) &= \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \sum_{j=1}^{n_{in}} \mathbf{N}_j(\mathbf{p})\mathbf{x}(t;\mathbf{p})u_j(t) + \mathbf{B}(\mathbf{p})\mathbf{u}(t), \\ \mathbf{y}(t;\mathbf{p}) &= \mathbf{C}(\mathbf{p})\mathbf{x}(t;\mathbf{p}). \end{cases} \tag{4.1}$$

Even though this system is nonlinear due to the terms involving $\mathbf{N}_j(\mathbf{p})$, the concept of transfer function can still be applied via the Volterra series representation [116].

Under some mild assumptions, the output $\mathbf{y}(t;\mathbf{p})$ of (4.1) can be represented as the Volterra series

$$\mathbf{y}(t;\mathbf{p}) = \sum_{k=1}^{\infty} \int_0^{t_1} \int_0^{t_2} \cdots \int_0^{t_k} \mathbf{h}_k(t_1,\ldots,t_k;\mathbf{p}) \left(\mathbf{u}\left(t - \sum_{i=1}^{k} t_i\right) \otimes \cdots \otimes \mathbf{u}\left(t - t_k\right) \right) \mathrm{d}t_k \cdots \mathrm{d}t_1,$$

where $\mathbf{h}_k(t_1,t_2,\ldots,t_k;\mathbf{p})$'s are the regular Volterra kernels, also called subsystem kernels. Then, taking the multivariable Laplace transform of the degree k regular kernel $\mathbf{h}_k$ leads to the k^{th} subsystem transfer function:

$$\begin{aligned} \mathbf{H}_k(s_1,\ldots,s_k;\mathbf{p}) = \mathbf{C}(\mathbf{p})&\mathbf{K}(s_k;\mathbf{p})\mathbf{N}(\mathbf{p}) \\ &\times [\mathbf{I}_{n_{in}} \otimes \mathbf{K}(s_{k-1};\mathbf{p})\mathbf{N}(\mathbf{p})] \cdots [\mathbf{I}_{n_{in}}^{\otimes^{k-2}} \otimes \mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p})] \\ &\times [\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p})], \end{aligned} \tag{4.9}$$

where $\otimes$ is the Kronecker product, $\mathbf{K}(s,\mathbf{p})$ is as in (4.7),

$$\begin{aligned} \mathbf{N}(\mathbf{p}) &= [\mathbf{N}_1(\mathbf{p})\ \mathbf{N}_2(\mathbf{p})\ \cdots\ \mathbf{N}_{n_{in}}(\mathbf{p})], \qquad \text{and} \\ \mathbf{I}_{n_{in}}^{\otimes^k} &= \underbrace{\mathbf{I}_{n_{in}} \otimes \cdots \otimes \mathbf{I}_{n_{in}}}_{k \text{ times}}. \end{aligned} \tag{4.10}$$

Since the notation in (4.9) is dense, to give an idea of the structure of these transfer functions, see Figure 4.3 for a SISO system and Figure 4.4 for a MIMO system with 2 inputs. You will observe that the MIMO transfer function is a row with entries corresponding to a SISO transfer function for each bilinear matrix $\mathbf{N}_j(\mathbf{p})$. This observation justifies why we will show visualizations of our results only for the SISO case, while the results and corresponding proofs will apply to the general MIMO case.

$$\text{SISO} \implies \mathbf{N}(\mathbf{p}) = \mathbf{N}_1(\mathbf{p}), \quad \mathbf{B}(\mathbf{p}) = \mathbf{b}(\mathbf{p}), \quad \mathbf{C}(\mathbf{p}) = \mathbf{c}^\top(\mathbf{p})$$

$$\begin{aligned}
\mathbf{c}^\top(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{b}(\mathbf{p}) &= \mathbf{H}_1(s_1;\mathbf{p}) \qquad \text{(linear)} \\
\mathbf{c}^\top(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{b}(\mathbf{p}) &= \mathbf{H}_2(s_1, s_2;\mathbf{p}) \\
\mathbf{c}^\top(\mathbf{p})\mathbf{K}(s_3;\mathbf{p})\mathbf{N}(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{r}(\mathbf{p}) &= \mathbf{H}_3(s_1, s_2, s_3;\mathbf{p})
\end{aligned}$$

Figure 4.3: Transfer functions for SISO PBMOR

$$n_{in} = 2 \implies \mathbf{N}(\mathbf{p}) = [\mathbf{N}_1(\mathbf{p})\ \mathbf{N}_2(\mathbf{p})]$$

$$\begin{aligned}
\mathbf{C}(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) &= \mathbf{H}_1(s_1;\mathbf{p}) \\
\mathbf{C}(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p})\,[\mathbf{I}_2\otimes\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p})] &= \mathbf{H}_2(s_1, s_2;\mathbf{p}) \\
\mathbf{C}(\mathbf{p})\mathbf{K}(s_3;\mathbf{p})\mathbf{N}(\mathbf{p})\,[\mathbf{I}_2\otimes\mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p})]\,[\mathbf{I}_2\otimes\mathbf{I}_2\otimes\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p})] &= \mathbf{H}_3(s_1, s_2, s_3;\mathbf{p})
\end{aligned}$$

$$\begin{bmatrix} \mathbf{C}(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}_1(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) \\ \mathbf{C}(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}_2(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) \end{bmatrix}^\top = \mathbf{H}_2(s_1, s_2;\mathbf{p})$$

$$\begin{bmatrix} \mathbf{C}(\mathbf{p})\mathbf{K}(s_3;\mathbf{p})\mathbf{N}_1(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}_1(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) \\ \mathbf{C}(\mathbf{p})\mathbf{K}(s_3;\mathbf{p})\mathbf{N}_1(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}_2(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) \\ \mathbf{C}(\mathbf{p})\mathbf{K}(s_3;\mathbf{p})\mathbf{N}_2(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}_1(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) \\ \mathbf{C}(\mathbf{p})\mathbf{K}(s_3;\mathbf{p})\mathbf{N}_2(\mathbf{p})\mathbf{K}(s_2;\mathbf{p})\mathbf{N}_2(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}) \end{bmatrix}^\top = \mathbf{H}_3(s_1, s_2, s_3;\mathbf{p})$$

Figure 4.4: Transfer functions for MIMO PBMOR

For details of this analysis, we refer the reader to [116, 119]. The Volterra series representation of bilinear systems has been successfully used for interpolation-based input-independent,

optimal model reduction of non-parametric bilinear systems; see, e.g., [20, 57].

Similarly, for the reduced bilinear system (4.2), the k^{th} subsystem transfer function is given by

$$\begin{aligned}\widetilde{\mathbf{H}}_k(s_1,\ldots,s_k;\mathbf{p}) = \widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{K}}(s_k;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p}) \\ \times [\mathbf{I}_{n_{in}} \otimes \widetilde{\mathbf{K}}(s_{k-1};\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p})] \cdots [\mathbf{I}_{n_{in}}^{\otimes^{k-2}} \otimes \widetilde{\mathbf{K}}(s_2;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p})] \\ \times [\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \widetilde{\mathbf{K}}(s_1;\mathbf{p})\widetilde{\mathbf{B}}(\mathbf{p})],\end{aligned} \tag{4.11}$$

where

$$\widetilde{\mathbf{N}}(\mathbf{p}) = [\widetilde{\mathbf{N}}_1(\mathbf{p})\ \widetilde{\mathbf{N}}_2(\mathbf{p})\ \cdots\ \widetilde{\mathbf{N}}_{n_{in}}(\mathbf{p})]. \tag{4.12}$$

This allows us to formulate the parametric interpolatory model reduction problem in our setting: Given interpolation frequencies $\{\sigma_1,\ldots,\sigma_q\} \subset \mathbb{C}$, nontrivial right direction $\mathbf{r} \in \mathbb{C}^{n_{in}}$, nontrivial left direction $\boldsymbol{\ell} \in \mathbb{C}^{n_{out}}$, and interpolation parameter sample $\boldsymbol{\pi} \in \mathcal{P}$, find $\mathbf{V}, \mathbf{W}$ such that the reduced model (4.2) constructed via projection as in (4.3) satisfies the following interpolation conditions for any $k \in \{1,\ldots,q\}$:

- tangential interpolation of the subsystems:

$$\mathbf{H}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right) = \widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right), \tag{4.13}$$

$$\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}) = \boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}); \tag{4.14}$$

- tangential interpolation of the subsystems' partial derivatives with respect to each

frequency variable: for each $i \in \{1, \ldots, k\}$

$$\frac{\partial}{\partial s_i}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right)\right) = \frac{\partial}{\partial s_i}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right)\right);; \tag{4.15}$$

- tangential interpolation of the subsystems' partial derivatives with respect to each parameter variable entry:

$$\boldsymbol{\mathcal{J}}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right)\right) = \boldsymbol{\mathcal{J}}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right)\right), \tag{4.16}$$

  where $\boldsymbol{\mathcal{J}}_{\mathbf{p}}(\cdot)$ denotes the matrix of sensitivities (Jacobian) with respect to $\mathbf{p}$;

- tangential interpolation of the subsystems' second partial derivatives with respect to each parameter variable entry:

$$\boldsymbol{\mathcal{H}}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right)\right) = \boldsymbol{\mathcal{H}}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})\left(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}\right)\right), \tag{4.17}$$

  where $\boldsymbol{\mathcal{H}}_{\mathbf{p}}(\cdot)$ denotes the Hessian (tensor) with respect to $\mathbf{p}$.

In other words, we would like to construct a reduced parametric bilinear system whose leading subsystems interpolate (both in frequency and parameter space) the corresponding leading subsystems of the full order parametric model. Note that we are not only enforcing Lagrange interpolation. We require the reduced model to match the parameter sensitivities and Hessians as well, which is important, especially, in the setting of reduced models in optimization. Moreover, these conditions can then be generalized for different reordering of the frequencies, multiple tangential directions, and several parameter values.

For a MIMO system, the output dimension of the transfer functions remains constant for all subsystems, while the input dimension increases with the subsystem's order:

$$\mathbf{H}_k \in \mathbb{C}^{n_{out} \times (n_{in})^k}.$$

This allows several choices for the right tangential interpolation. Here we chose (4.13), which has dimension $(n_{in})^k \times (n_{in})^{k-1}$, in order to tangentially interpolate each entry of the transfer functions as in the linear case. Other choices for right tangential interpolation are possible. For example, one might choose to multiply by $(\mathbf{1}_{n_{in}} \otimes \mathbf{r})$, where $\mathbf{1}_{n_{in}}$ denotes the column vector of dimension n_{in} with all its entries 1. Then one would be looking to interpolate the sum of the entries of the transfer function instead of each entry separately. This new choice would decrease the dimension of the objects to interpolate to scalars instead of vectors (with dimension increasing with subsystem order). In practice, we do not suffer from this, since we only choose to interpolate the first two transfer functions. If one wishes to interpolate a large number of subsystems, it may be worth considering an option like $(\mathbf{1}_{n_{in}} \otimes \mathbf{r})$, so that the reduced order model state dimension $\widetilde{n}_{st}$ is not too large. Results would be analogous to the ones presented below.

4.2 Subspace conditions for parametric bilinear interpolation

In this section, we establish the subspace conditions to enforce the desired interpolation conditions (4.13)–(4.17) for parametric bilinear systems. Note that even for the parametric bilinear system (4.1) we consider here, some of these interpolation conditions, e.g, (4.13), do not involve parameter gradient and/or parameter Hessian interpolation, and thus can be in-

terpreted as regular tangential bilinear subsystem interpolation for a fixed parameter $\mathbf{p} = \boldsymbol{\pi}$, as considered in [25]. However, even though our subspace conditions for (4.13)-(4.14) will look similar to those in [25], we include the corresponding theorem (Theorem 4.2 below) and its complete proof for the following reasons. Although [25] considers tangential interpolation for non-parametric bilinear systems, the tangential interpolation conditions appear differently. In our formulation, tangential directions appear in Kronecker product form due to the structure of $\mathbf{H}_k(s_1, s_2, \ldots, s_k; \mathbf{p})$ as defined in (4.9), illustrating that $\mathbf{H}_k(s_1, s_2, \ldots, s_k; \mathbf{p})$ can be considered to have ${n_{in}}^k$ inputs. Our conditions result in regular tangential interpolation in the subblocks of $\mathbf{H}_k(s_1, s_2, \ldots, s_k; \mathbf{p})$; details will be given below. Moreover, we provide different proofs for (4.13) and (4.14), which we later use in the proof of Theorem 4.4. Finally, we include the $\mathbf{E}(\mathbf{p})$ term in the full model. Clearly, the subspace conditions (4.16) for matching the parameter gradient and (4.17) for matching the parameter Hessian are new and will be fully discussed.

Theorem 4.2 will establish the subspace conditions for enforcing Lagrange interpolation for the subsystem transfer functions, namely (4.13) and (4.14). This will be achieved using one-sided projections. Then, using two-sided projections, Theorem 4.4 will extend these results to matching sensitivities (derivatives) with respect to frequencies and parameters, thus satisfying (4.15) and (4.16). Finally, by adding further subspace conditions, Theorem 4.5 will show how to match the parameter Hessian, i.e., satisfying (4.17).

Theorem 4.2. *Let q be the number of subsystems we wish to interpolate, and $\{\sigma_1, \ldots, \sigma_q\} \subset \mathbb{C}$ and $\boldsymbol{\pi} \in \mathcal{P}$ be such that $\mathbf{K}(\sigma_i; \boldsymbol{\pi})$ exists for all $i \in \{1, \ldots, q\}$. Consider also the nontrivial*

vectors $\mathbf{r} \in \mathbb{C}^{n_{in}}$ *and* $\boldsymbol{\ell} \in \mathbb{C}^{n_{out}}$. *Define*

$$\begin{aligned} \mathbf{V}_1 &= \mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}, \\ \mathbf{V}_k &= \mathbf{K}(\sigma_k;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})(\mathbf{I}_{n_{in}} \otimes \mathbf{V}_{k-1}), \quad \textit{for} \;\; k = 2,\ldots,q, \end{aligned} \tag{4.18}$$

$$\begin{aligned} \mathbf{W}_1 &= \mathbf{K}(\sigma_q;\boldsymbol{\pi})^\top \mathbf{C}(\boldsymbol{\pi})^\top \boldsymbol{\ell}, \;\; \textit{and} \\ \mathbf{W}_k &= \mathbf{K}(\sigma_{q+1-k};\boldsymbol{\pi})^\top \overline{\mathbf{N}}(\boldsymbol{\pi})^\top (\mathbf{I}_{n_{in}} \otimes \mathbf{W}_{k-1}), \quad \textit{for} \;\; k = 2,\ldots,q, \end{aligned} \tag{4.19}$$

where

$$\overline{\mathbf{N}}(\mathbf{p}) = \begin{bmatrix} \mathbf{N}_1(\mathbf{p}) \\ \mathbf{N}_2(\mathbf{p}) \\ \vdots \\ \mathbf{N}_{n_{in}}(\mathbf{p}) \end{bmatrix}.$$

If

$$\bigcup_{k=1}^{q} \mathrm{Ran}(\mathbf{V}_k) \subseteq \mathrm{Ran}(\mathbf{V}), \tag{4.20}$$

then, for $k = 1,\ldots,q$,

$$\mathbf{H}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}) = \widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r}). \tag{4.21}$$

If

$$\bigcup_{k=1}^{q} \mathrm{Ran}(\mathbf{W}_k) \subseteq \mathrm{Ran}(\mathbf{W}), \tag{4.22}$$

then, for $k = 1,\ldots,q$,

$$\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi}) = \boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi}). \tag{4.23}$$

The proof of this result follows by induction. The case $k = 1$ corresponds to the linear case, which we provided insight for in Figures 4.1 and 4.2. Before jumping into the complete

general proof below, we want to again provide a warm-up for the reader to familiarize themselves with the tedious manipulations ahead. Hence we visualize the assumptions and sketch the proof for the second subsystem ($k = 2$) in Figures 4.5 and 4.6. For simplicity, we will assume $\mathbf{N}(\mathbf{p}) = \mathbf{N}_1(\mathbf{p})$ in these figures, i.e., we provide intuition for only one entry in $\mathbf{H}_2$, since all other entries in $\mathbf{H}_2$ have analogous structure in terms of $\mathbf{N}_i(\mathbf{p})$ (as seen in Figure 4.4). Note that $\mathbf{V}_1$ and $\mathbf{W}_1$ in Figure 4.5 are the same as $\mathbf{V}$ and $\mathbf{W}$ in Figure 4.1.

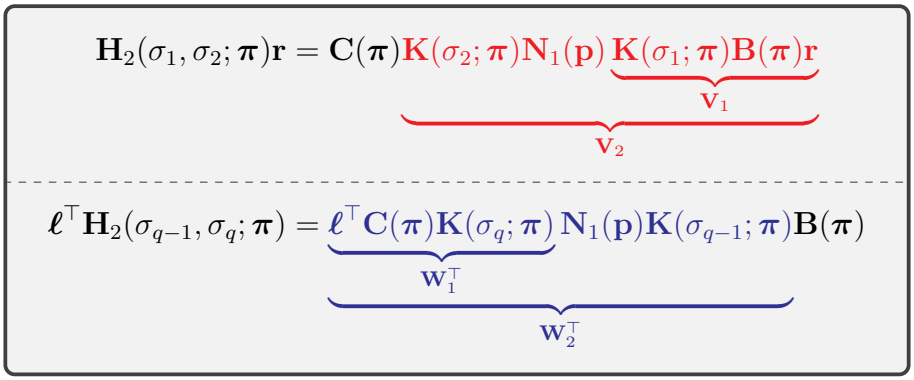


Figure 4.5: Visualization of the MOR basis in Theorem 4.2 for $k = 2$

Hence interpolation of $\mathbf{H}_1$ is guaranteed. Since we want to interpolate $\mathbf{H}_2$, we include $\mathbf{V}_2$ and $\mathbf{W}_2$, which contain the right and left information $\mathbf{H}_2(\sigma_1, \sigma_2; \mathbf{p})\mathbf{r}$ and $\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_{q-1}, \sigma_q; \boldsymbol{\pi})$, respectively. In Figure 4.6 we show the difference $\mathbf{H}_2(\sigma_1, \sigma_2; \mathbf{p})\mathbf{r} - \widetilde{\mathbf{H}}(\sigma_1, \sigma_2; \mathbf{p})\mathbf{r}$. Even though Figure 4.6 may seem intimidating, it shows very similar steps to those in Figure 4.2: we highlight in red the vector added to the range of $\mathbf{V}$ specifically to interpolate $\mathbf{H}_2$ from the right; we use the definition of the reduced matrices to create common factors with the term corresponding to the full order system; we apply the fact that we know $\mathbf{H}_1$ is interpolated; and we factor out the highlighted vector from both terms. Then all is left to show, as before, is that this expression is indeed zero. The analysis follows analogously for left interpolation with $\mathbf{W}$ (in blue) at the bottom of Figure 4.6.

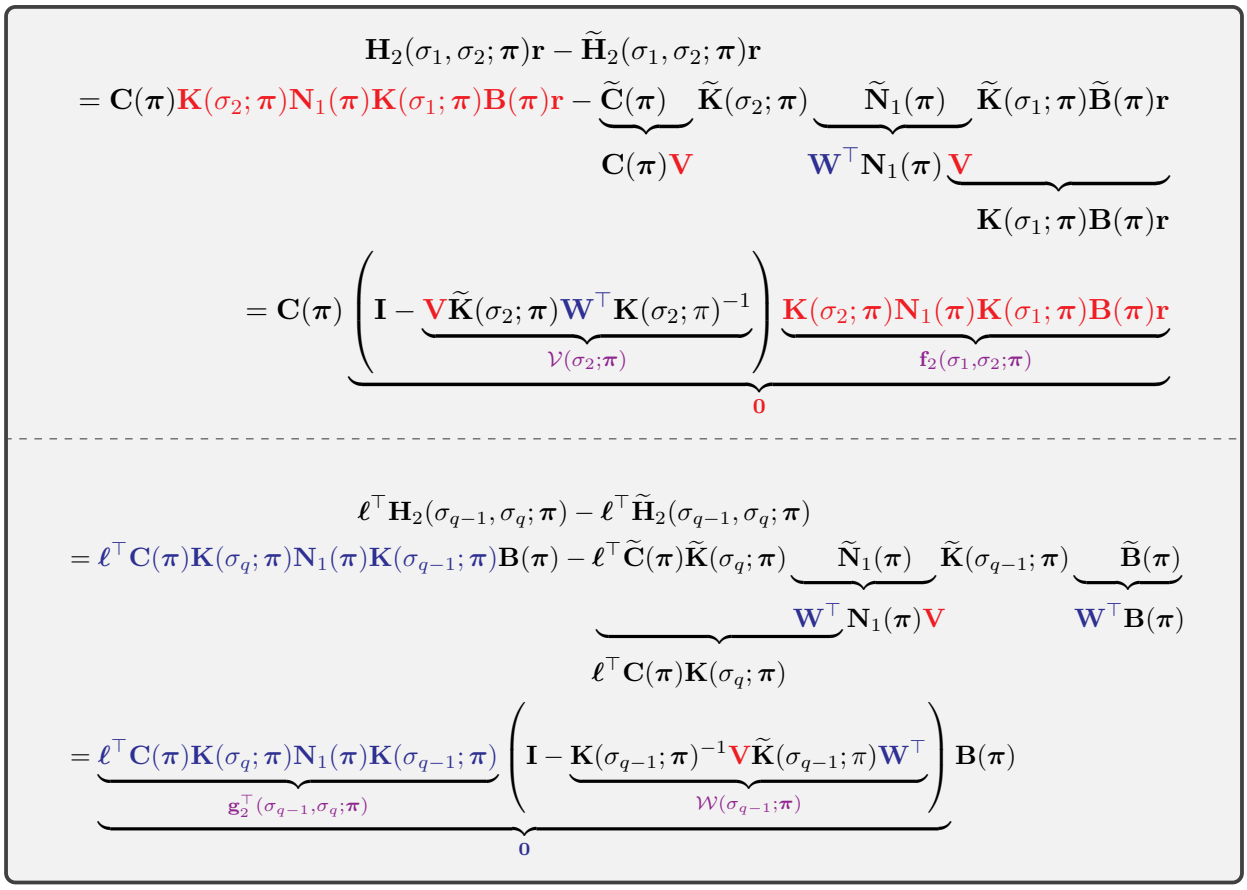


Figure 4.6: Sketch of the proof for Theorem 4.2 for $k = 2$ (notation from the proof in purple)

Proof. Define

$$\begin{aligned}
\mathcal{V}(s;\mathbf{p}) &= \mathbf{V}\widetilde{\mathbf{K}}(s;\mathbf{p})\mathbf{W}^\top\mathbf{K}(s;\mathbf{p})^{-1},\\
\mathcal{W}(s;\mathbf{p}) &= \mathbf{K}(s;\mathbf{p})^{-1}\mathbf{V}\widetilde{\mathbf{K}}(s;\mathbf{p})\mathbf{W}^\top,\\
\mathbf{f}_k(s_1,\ldots,s_k;\mathbf{p}) &= \mathbf{K}(s_k;\mathbf{p})\mathbf{N}(\mathbf{p})\\
&\quad\times(\mathbf{I}_{n_{in}}\otimes\mathbf{K}(s_{k-1};\mathbf{p})\mathbf{N}(\mathbf{p}))\cdots(\mathbf{I}_{n_{in}}^{\otimes^{k-2}}\otimes\mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p}))\\
&\quad\times(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p})\mathbf{r}),\ \text{and}\\
\mathbf{g}_k^\top(s_1,\ldots,s_k;\mathbf{p}) &= \boldsymbol{\ell}^\top\mathbf{C}(\mathbf{p})\mathbf{K}(s_k;\mathbf{p})\mathbf{N}(\mathbf{p})\\
&\quad\times(\mathbf{I}_{n_{in}}\otimes\mathbf{K}(s_{k-1};\mathbf{p})\mathbf{N}(\mathbf{p}))\cdots(\mathbf{I}_{n_{in}}^{\otimes^{k-2}}\otimes\mathbf{K}(s_2;\mathbf{p})\mathbf{N}(\mathbf{p}))\\
&\quad\times(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{K}(s_1;\mathbf{p})).
\end{aligned}\tag{4.24}$$

Note that $\widetilde{\mathbf{K}}(s;\mathbf{p})^{-1} = \mathbf{W}^\top\mathbf{K}(s;\mathbf{p})^{-1}\mathbf{V}$ and

$$\begin{aligned}
\mathcal{V}^2(s;\mathbf{p}) &= \mathbf{V}\widetilde{\mathbf{K}}(s;\mathbf{p})\mathbf{W}^\top\mathbf{K}(s;\mathbf{p})^{-1}\mathbf{V}\widetilde{\mathbf{K}}(s;\mathbf{p})\mathbf{W}^\top\mathbf{K}(s;\mathbf{p})^{-1}\\
&= \mathbf{V}\widetilde{\mathbf{K}}(s;\mathbf{p})\mathbf{W}^\top\mathbf{K}(s;\mathbf{p})^{-1}\\
&= \mathcal{V}(s;\mathbf{p}).
\end{aligned}$$

Thus, $\mathcal{V}(s;\mathbf{p})$ is a skew projector onto $\mathrm{Ran}(\mathbf{V})$. Similarly $\mathcal{W}(s;\mathbf{p})$ is a skew projector along $\mathrm{Ker}(\mathbf{W}^\top)$. Also note that $\mathbf{H}_k(s_1,\ldots,s_k;\mathbf{p})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{r}) = \mathbf{C}(\mathbf{p})\mathbf{f}_k(s_1,\ldots,s_k;\mathbf{p})$ and $\boldsymbol{\ell}^\top\mathbf{H}_k(s_1,\ldots,s_k;\mathbf{p}) = \mathbf{g}_k^\top(s_1,\ldots,s_k;\mathbf{p})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{B}(\mathbf{p}))$.

First we prove (4.21). Suppose (4.20) holds. We know that (4.21) is true for $k = 1$ by Theorem 4.1. Assume that the result is true for $k-1$; recall $k < q$. Then using the

definitions of $\mathcal{V}(s;\mathbf{p})$ and $\mathbf{f}_k(s_1,\ldots,s_k;\mathbf{p})$ from (4.24), we obtain

$$\begin{aligned}\mathbf{H}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{r})-\widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{r})\\ =\mathbf{C}(\boldsymbol{\pi})(\mathbf{I}_{n_{st}}-\mathcal{V}(\sigma_k;\boldsymbol{\pi}))\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}),\end{aligned}\tag{4.25}$$

where we factor out the term $\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})$ using the right interpolation of

$$\mathbf{H}_{k-1}(\sigma_1,\ldots,\sigma_{k-1};\boldsymbol{\pi})$$

due to the induction assumption (see top of Figure 4.6). Then, what is left to show is that (4.25) is zero: By the construction of $\mathbf{V}$ in (4.20), we obtain $\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})\in\mathrm{Ran}(\mathbf{V})$. Hence $\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})\in\mathrm{Ran}(\mathcal{V}(\sigma_k;\boldsymbol{\pi}))$, which implies

$$(\mathbf{I}_{n_{st}}-\mathcal{V}(\sigma_k;\boldsymbol{\pi}))\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})=0,\tag{4.26}$$

since $\mathcal{V}(s;\mathbf{p})$ is a skew projector onto $\mathrm{Ran}(\mathbf{V})$. The proof of (4.23) follows similarly. Note that the order of the interpolation frequencies in (4.23) is different than the order in (4.21). Suppose (4.22) holds. As before, the result is true for $k=1$. Assume that it holds for $k-1$. Similar to (4.25), we obtain (see bottom of Figure 4.6)

$$\begin{aligned}\boldsymbol{\ell}^\top\mathbf{H}_k(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi})-\boldsymbol{\ell}^\top\widetilde{\mathbf{H}}_k(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi})\\ =\mathbf{g}_k^\top(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi})[\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes(\mathbf{I}_n-\mathcal{W}(\sigma_{q+1-k};\boldsymbol{\pi}))\mathbf{B}(\boldsymbol{\pi})].\end{aligned}\tag{4.27}$$

Again we show that this expression is zero. Note that by the definition of $\mathbf{g}_k$ and the construction of $\mathbf{W}$ in (4.22), we have

$$\mathbf{g}_k(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi})\perp\mathrm{Ker}(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathcal{W}(\sigma_{q+1-k};\boldsymbol{\pi}));$$

thus

$$\mathbf{g}_k^\top(\sigma_{q+1-k},\ldots,\sigma_q;\boldsymbol{\pi})[\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes(\mathbf{I}_{n_{st}}-\mathcal{W}(\sigma_{q+1-k};\boldsymbol{\pi}))]=0. \tag{4.28}$$

□

Theorem 4.2 provides tangential interpolation of $\mathbf{H}_k(s_1,\ldots,s_k;\mathbf{p})$ in a specific order of the frequencies, namely in the order $\{\sigma_1,\ldots,\sigma_k\}$. However one might also consider enforcing interpolation at the frequency samples $\{\sigma_1,\ldots,\sigma_k\}$ in any order, including repetitions, as is considered in [25]. Indeed, as we will show in Theorem 4.4, interpolation of the transfer function sensitivities will require this. The result as shown in Corollary 4.3 below is a direct extension of Theorem 4.2; thus we skip the details. It simply requires the subspaces to contain all possible combinations.

Corollary 4.3. *Let q be the number of subsystems we wish to interpolate. Consider $\{\sigma_1,\ldots,\sigma_q\}\subset\mathbb{C}$ and $\boldsymbol{\pi}\in\mathcal{P}$ such that $\mathbf{K}(\sigma_i;\boldsymbol{\pi})$ exists for all $i\in\{1,\ldots,q\}$. Consider also the nontrivial vectors $\mathbf{r}\in\mathbb{C}^{n_{in}}$ and $\boldsymbol{\ell}\in\mathbb{C}^{n_{out}}$. Define, for $k=2,\ldots,q$:*

$$\begin{aligned}
\mathbf{V}_1&=[\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r},\ \ \cdots,\ \ \mathbf{K}(\sigma_q;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}],\\
\mathbf{V}_k&=[\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes\mathbf{V}_{k-1}),\ \ \cdots,\ \ \mathbf{K}(\sigma_q;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes\mathbf{V}_{k-1})],\\
\mathbf{W}_1&=[\mathbf{K}(\sigma_1;\boldsymbol{\pi})^\top\mathbf{C}(\boldsymbol{\pi})^\top\boldsymbol{\ell},\ \ \cdots,\ \ \mathbf{K}(\sigma_q;\boldsymbol{\pi})^\top\mathbf{C}(\boldsymbol{\pi})^\top\boldsymbol{\ell}],\ \textit{and}\\
\mathbf{W}_k&=[\mathbf{K}(\sigma_1;\boldsymbol{\pi})^\top\overline{\mathbf{N}}(\boldsymbol{\pi})^\top(\mathbf{I}_{n_{in}}\otimes\mathbf{W}_{k-1}),\ \ \cdots,\ \ \mathbf{K}(\sigma_q;\boldsymbol{\pi})^\top\overline{\mathbf{N}}(\boldsymbol{\pi})^\top(\mathbf{I}_{n_{in}}\otimes\mathbf{W}_{k-1})].
\end{aligned} \tag{4.29}$$

If

$$\bigcup_{k=1}^{q}\mathrm{Ran}(\mathbf{V}_k)\subseteq\mathrm{Ran}(\mathbf{V}), \tag{4.30}$$

then, for $k=1,\ldots,q$, and for any $i_1,\ldots,i_k\in\{1,\ldots,q\}$,

$$\mathbf{H}_k(\sigma_{i_1},\ldots,\sigma_{i_k};\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{r})=\widetilde{\mathbf{H}}_k(\sigma_{i_1},\ldots,\sigma_{i_k};\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\mathbf{r}).$$

If

$$\bigcup_{k=1}^{q} \mathrm{Ran}(\mathbf{W}_k) \subseteq \mathrm{Ran}(\mathbf{W}), \tag{4.31}$$

then, for $k = 1, \ldots, q$, *and for any* $i_1, \ldots, i_k \in \{1, \ldots, q\}$,

$$\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_{i_1}, \ldots, \sigma_{i_k}; \boldsymbol{\pi}) = \boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_{i_1}, \ldots, \sigma_{i_k}; \boldsymbol{\pi}).$$

So far, we proved the interpolation conditions using either only $\mathbf{V}$ or only $\mathbf{W}$; i.e., we assumed interpolation information only in one of the subspaces, considering one-sided projection. The next theorem shows that when both subspaces are considered, one automatically matches the sensitivities (derivatives) with respect to the frequencies and parameter; indeed without computing the sensitivities to be matched.

Theorem 4.4. *Assume the hypotheses of Corollary 4.3. Let* $\mathbf{V}_k$ *and* $\mathbf{W}_k$ *be constructed as in (4.29) for* $k = 1, 2, \ldots, q$. *If both*

$$\bigcup_{k=1}^{q} \mathrm{Ran}(\mathbf{V}_k) \subseteq \mathrm{Ran}(\mathbf{V}) \qquad \textit{and} \qquad \bigcup_{k=1}^{q} \mathrm{Ran}(\mathbf{W}_k) \subseteq \mathrm{Ran}(\mathbf{W}),$$

then for $k = 1, \ldots, q$ *and for* $i = 1, \ldots, k$*:*

$$\frac{\partial}{\partial s_i}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r})\right) = \frac{\partial}{\partial s_i}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r})\right),$$

and

$$\mathcal{J}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r})\right) = \mathcal{J}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1, \ldots, \sigma_k; \boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r})\right),$$

where $\mathcal{J}_{\mathbf{p}}(\cdot)$ *denotes the matrix of sensitivities (Jacobian) with respect to* $\mathbf{p}$.

Proof. The proof will use similar ideas to the proof of Theorem 4.2. We will use appropriately defined projectors to eliminate the terms from the error expressions yielding the necessary interpolation conditions. Recall the definitions $\mathcal{V}(s;\mathbf{p})$, $\mathcal{W}(s;\mathbf{p})$, $\mathbf{f}_k(s_1,\dots,s_k;\mathbf{p})$, and $\mathbf{g}_k^\top(s_1,\dots,s_k;\mathbf{p})$ from (4.24). Similarly, define

$$\begin{aligned}\widetilde{\mathbf{f}}_k(s_1,\dots,s_k;\mathbf{p}) &= \widetilde{\mathbf{K}}(s_k;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p})(\mathbf{I}_{n_{in}}\otimes\widetilde{\mathbf{K}}(s_{k-1};\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p}))\cdots(\mathbf{I}_{n_{in}}^{\otimes^{k-2}}\otimes\widetilde{\mathbf{K}}(s_2;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p}))\\ &\quad\times(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\widetilde{\mathbf{K}}(s_1;\mathbf{p})\widetilde{\mathbf{B}}(\mathbf{p})\mathbf{r}),\ \text{and}\\ \widetilde{\mathbf{g}}_k^\top(s_1,\dots,s_k;\mathbf{p}) &= \boldsymbol{\ell}^\top\widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{K}}(s_k;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p})(\mathbf{I}_{n_{in}}\otimes\widetilde{\mathbf{K}}(s_{k-1};\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p}))\cdots(\mathbf{I}_{n_{in}}^{\otimes^{k-2}}\otimes\widetilde{\mathbf{K}}(s_2;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p}))\\ &\quad\times(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes\widetilde{\mathbf{K}}(s_1;\mathbf{p})).\end{aligned}$$

Let $k\in\{1,\dots,q\}$ and $i\in\{1,\dots,k\}$. Under the assumptions of the theorem, we know (4.26) and (4.28) are satisfied for any choice of frequencies due to Corollary 4.3; in particular,

$$\begin{aligned}(\mathbf{I}_{n_{st}}-\mathcal{V}(\sigma_\kappa;\boldsymbol{\pi}))\mathbf{f}_\kappa(\sigma_1,\dots,\sigma_\kappa;\boldsymbol{\pi})=0,\quad&\text{and}\\ \mathbf{g}_{\kappa-\iota+1}^\top(\sigma_\iota,\dots,\sigma_\kappa;\boldsymbol{\pi})[\mathbf{I}_{n_{in}}^{\otimes^{\kappa-\iota}}\otimes(\mathbf{I}_n-\mathcal{W}(\sigma_\kappa;\boldsymbol{\pi}))]=0,&\end{aligned}$$

for any $\kappa\in\{1,\dots,q\}$ and $\iota<\kappa$. Recalling the definitions of the full and reduced transfer functions (4.9) and (4.11), together with (4.25) and (4.27), and our definitions in this proof, we can rewrite these two terms as

$$\begin{aligned}\mathbf{f}_\kappa(\sigma_1,\dots,\sigma_\kappa;\boldsymbol{\pi})-\mathbf{V}\widetilde{\mathbf{f}}_\kappa(\sigma_1,\dots,\sigma_\kappa;\boldsymbol{\pi})=0,\quad&\text{and}\\ \mathbf{g}_{\kappa-\iota+1}^\top(\sigma_\iota,\dots,\sigma_\kappa;\boldsymbol{\pi})-\widetilde{\mathbf{g}}_{\kappa-\iota+1}^\top(\sigma_\iota,\dots,\sigma_\kappa;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{\kappa-\iota}}\otimes\mathbf{W}^\top)=0,&\end{aligned}$$

or equivalently

$$\begin{aligned} \mathbf{f}_\kappa(\sigma_1,\dots,\sigma_\kappa;\boldsymbol{\pi}) &= \mathbf{V}\widetilde{\mathbf{f}}_\kappa(\sigma_1,\dots,\sigma_\kappa;\boldsymbol{\pi}) \quad \text{and} \\ \mathbf{g}_{\kappa-\iota+1}^\top(\sigma_\iota,\dots,\sigma_\kappa;\boldsymbol{\pi}) &= \widetilde{\mathbf{g}}_{\kappa-\iota+1}^\top(\sigma_\iota,\dots,\sigma_\kappa;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{\kappa-\iota}} \otimes \mathbf{W}^\top). \end{aligned} \tag{4.32}$$

Now fix $k \in \{1,\dots,q\}$ and $i \in \{1,\dots,k\}$, and recall that $\widetilde{\mathbf{E}}(\mathbf{p}) = \mathbf{W}^\top\mathbf{E}(\mathbf{p})\mathbf{V}$. Then

$$\begin{aligned} &\frac{\partial}{\partial s_i}\left(\boldsymbol{\ell}^\top\widetilde{\mathbf{H}}_k(\sigma_1,\dots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r})\right) \\ &= \widetilde{\mathbf{g}}_{k-i+1}^\top(\sigma_i,\dots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-i}} \otimes \mathbf{W}^\top)\mathbf{E}(\boldsymbol{\pi})\mathbf{V}\widetilde{\mathbf{f}}_i(\sigma_1,\dots,\sigma_i;\boldsymbol{\pi}) \\ &= \mathbf{g}_{k-i+1}^\top(\sigma_i,\dots,\sigma_k;\boldsymbol{\pi})\mathbf{E}(\boldsymbol{\pi})\mathbf{f}_i(\sigma_1,\dots,\sigma_i;\boldsymbol{\pi}) \qquad \text{(by (4.32))} \\ &= \frac{\partial}{\partial s_i}\left(\boldsymbol{\ell}^\top\mathbf{H}_k(\sigma_1,\dots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}} \otimes \mathbf{r})\right). \end{aligned} \tag{4.33}$$

To summarize this proof so far and for a break from notation, see Figure 4.7. The top of the figure highlights that the basis again contains left and right information about the transfer function, while the bottom provides a visualization of (4.33) for $k = 2$. To prove interpolation, we only need to be careful where to split the expression of the transfer function, so that both the right and the left side of the split are included in $\mathbf{V}$ and $\mathbf{W}$, respectively. Note that the split happens in the term resulting from taking the partial derivative. Once the split is made, we simply apply the definition of the reduced matrix in the middle of the split ($\widetilde{\mathbf{E}}(\mathbf{p})$ in this case) and apply (4.21) to recover the full order transfer function terms.

Similarly, we can prove interpolation of the parameter gradients. Let p_j refer to any entry of the parameter vector $\mathbf{p} \in \mathcal{P}$ and let $\mathbf{M}_{p_j}(\mathbf{p})$ denote the partial derivative of $\mathbf{M}(\mathbf{p})$ with respect to p_j. Since the expression of the parameter gradient for a general subsystem transfer function $\mathbf{H}_k(s_1,\dots,s_k;\mathbf{p})$ becomes too involved to properly present in a single page, we provide the proof for the second subsystem (the result for the first subsystem follows from Theorem 4.1) in Figure 4.8 and sketch the proof for a general subsystem. Figure 4.8 is even

$$\frac{\partial}{\partial s_1}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right) = \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma_2;\boldsymbol{\pi})\mathbf{N}_1(\boldsymbol{\pi})\mathbf{K}(\sigma_1;\boldsymbol{\pi})}_{\mathbf{g}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})=\mathbf{W}_2(:,2)^\top}\mathbf{E}(\boldsymbol{\pi})\underbrace{\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{f}_1(\sigma_1;\boldsymbol{\pi})=\mathbf{V}_1(:,1)}$$

$$\frac{\partial}{\partial s_2}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right) = \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma_2;\boldsymbol{\pi})}_{\mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})=\mathbf{W}_1(:,2)^\top}\mathbf{E}(\boldsymbol{\pi})\underbrace{\mathbf{K}(\sigma_2;\boldsymbol{\pi})\mathbf{N}_1(\boldsymbol{\pi})\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{f}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})=\mathbf{V}_2(:,q+1)}$$

Note: We use MATLAB notation to refer to the columns of the basis in (4.29).

$$\begin{aligned}
&\frac{\partial}{\partial s_2}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right)\\
&= \underbrace{\boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})}_{\widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi})}\ \underbrace{\widetilde{\mathbf{E}}(\boldsymbol{\pi})}_{\mathbf{W}^\top\mathbf{E}(\boldsymbol{\pi})\mathbf{V}}\ \underbrace{\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})\widetilde{\mathbf{N}}_1(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})\widetilde{\mathbf{B}}(\boldsymbol{\pi})\mathbf{r}}_{\widetilde{\mathbf{f}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})}\\
&= \quad \mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi}) \qquad \mathbf{E}(\boldsymbol{\pi}) \qquad \mathbf{f}_2(\sigma_1,\sigma_2;\boldsymbol{\pi}) \qquad \text{(by (4.32))}\\
&= \frac{\partial}{\partial s_2}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right)
\end{aligned}$$

Figure 4.7: Visualization of the interpolation of the partial derivatives with respect to frequencies for $k = 2$ and with only one bilinear term $\mathbf{N}(\mathbf{p}) = \mathbf{N}_1(\mathbf{p})$

$$
\begin{aligned}
&\frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right)\\
&= \frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{K}}(s_2;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p})\widetilde{\mathbf{K}}(s_1;\mathbf{p})\widetilde{\mathbf{B}}(\mathbf{p})\mathbf{r}\right)\Big|_{(\sigma_1,\sigma_2;\boldsymbol{\pi})}\\
&= \boldsymbol{\ell}^\top \underbrace{\widetilde{\mathbf{C}}_{p_j}(\boldsymbol{\pi})}_{\mathbf{C}_{p_j}(\boldsymbol{\pi})\mathbf{V}} \underbrace{\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})\widetilde{\mathbf{N}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})\widetilde{\mathbf{B}}(\boldsymbol{\pi})\mathbf{r}}_{\widetilde{\mathbf{f}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})}\\
&\quad - \underbrace{\boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})}_{\widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi})} \underbrace{\widetilde{\mathbf{K}}_{p_j}(\sigma_2;\boldsymbol{\pi})^{-1}}_{\mathbf{W}^\top\mathbf{K}_{p_j}(\sigma_2;\boldsymbol{\pi})^{-1}\mathbf{V}} \underbrace{\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})\widetilde{\mathbf{N}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})\widetilde{\mathbf{B}}(\boldsymbol{\pi})\mathbf{r}}_{\widetilde{\mathbf{f}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})}\\
&\quad + \underbrace{\boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})}_{\widetilde{\mathbf{g}}_1(\sigma_2;\boldsymbol{\pi})} \underbrace{\widetilde{\mathbf{N}}_{p_j}(\boldsymbol{\pi})}_{\mathbf{W}^\top\mathbf{N}_{p_j}(\boldsymbol{\pi})\mathbf{V}} \underbrace{\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})\widetilde{\mathbf{B}}(\boldsymbol{\pi})\mathbf{r}}_{\widetilde{\mathbf{f}}_1(\sigma_1;\boldsymbol{\pi})}\\
&\quad - \underbrace{\boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})\widetilde{\mathbf{N}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})}_{\widetilde{\mathbf{g}}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})} \underbrace{\widetilde{\mathbf{K}}_{p_j}(\sigma_1;\boldsymbol{\pi})^{-1}}_{\mathbf{W}^\top\mathbf{K}_{p_j}(\sigma_1;\boldsymbol{\pi})^{-1}\mathbf{V}} \underbrace{\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})\widetilde{\mathbf{B}}(\boldsymbol{\pi})\mathbf{r}}_{\widetilde{\mathbf{f}}_1(\sigma_1;\boldsymbol{\pi})}\\
&\quad + \underbrace{\boldsymbol{\ell}^\top \widetilde{\mathbf{C}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_2;\boldsymbol{\pi})\widetilde{\mathbf{N}}(\boldsymbol{\pi})\widetilde{\mathbf{K}}(\sigma_1;\boldsymbol{\pi})}_{\widetilde{\mathbf{g}}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})} \underbrace{\widetilde{\mathbf{B}}_{p_j}(\boldsymbol{\pi})}_{\mathbf{W}^\top\mathbf{B}_p(\boldsymbol{\pi})}\mathbf{r}\\
&= \frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right) \qquad \text{(by (4.32))}
\end{aligned}
$$

Figure 4.8: Visualization of the interpolation of the partial derivative with respect to a parameter entry for $k = 2$ and with only one bilinear term, $\mathbf{N}(\mathbf{p}) = \mathbf{N}_1(\mathbf{p})$

more involved than the last figure, but the steps will again be the same (repeated several times). Since now the partial derivative affects all system matrices, the product rule results in multiple additive terms for $\frac{\partial}{\partial p_j}\mathbf{H}_2$. Then we repeat the steps (split, rewrite a reduced matrix, interpolation of previous subsystem already guaranteed) for each term. Figure 4.8 shows

$$\frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes \mathbf{r})\right) = \frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes \mathbf{r})\right).$$

Since p_j was an arbitrary entry of $\mathbf{p}$, this yields

$$\mathcal{J}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes \mathbf{r})\right) = \mathcal{J}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes \mathbf{r})\right),$$

as desired. Now, for the general case, consider

$$\begin{aligned}
&\frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes \mathbf{r})\right)\\
&= \boldsymbol{\ell}^\top \mathbf{C}_{p_j}(\boldsymbol{\pi})\mathbf{V}\widetilde{\mathbf{f}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}) && (4.34)\\
&\quad + \widetilde{\mathbf{g}}_1^\top(\sigma_k;\boldsymbol{\pi})\mathbf{W}^\top \mathbf{K}_{p_j}(\sigma_k;\boldsymbol{\pi})^{-1}\mathbf{V}\widetilde{\mathbf{f}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}) && (4.35)\\
&\quad + \widetilde{\mathbf{g}}_1^\top(\sigma_k;\boldsymbol{\pi})\mathbf{W}^\top \mathbf{N}_{p_j}(\boldsymbol{\pi})(\mathbf{I}_{n_{in}}\otimes \mathbf{V})\widetilde{\mathbf{f}}_{k-1}(\sigma_1,\ldots,\sigma_{k-1};\boldsymbol{\pi}) && (4.36)\\
&\quad + \cdots + \widetilde{\mathbf{g}}_k^\top(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes \mathbf{W}^\top \mathbf{B}_{p_j}(\boldsymbol{\pi})\mathbf{r}). && (4.37)
\end{aligned}$$

Consider the right-hand side of (4.34). Using (4.32), one can replace $\mathbf{V}\widetilde{\mathbf{f}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})$ with $\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})$. Similarly, in (4.35), using (4.32), one replaces $\widetilde{\mathbf{g}}_1^\top(\sigma_k;\boldsymbol{\pi})\mathbf{W}^\top$ with $\mathbf{g}_1^\top(\sigma_k;\boldsymbol{\pi})$.

Continuing in this fashion, we obtain

$$\begin{aligned}\frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes \mathbf{r})\right) &= \boldsymbol{\ell}^\top \mathbf{C}_{p_j}(\boldsymbol{\pi})\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}) \\ &\quad + \mathbf{g}_1^\top(\sigma_k;\boldsymbol{\pi})\mathbf{K}_{p_j}(\sigma_k;\boldsymbol{\pi})^{-1}\mathbf{f}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi}) \\ &\quad + \mathbf{g}_1^\top(\sigma_k;\boldsymbol{\pi})\mathbf{N}_{p_j}(\boldsymbol{\pi})\mathbf{f}_{k-1}(\sigma_1,\ldots,\sigma_{k-1};\boldsymbol{\pi}) \\ &\quad + \cdots + \mathbf{g}_k^\top(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes \mathbf{B}_{p_j}(\boldsymbol{\pi})\mathbf{r}) \\ &= \frac{\partial}{\partial p_j}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_{n_{in}}^{\otimes^{k-1}}\otimes \mathbf{r})\right),\end{aligned}$$

as desired. □

We now present the final theoretical result, showing the interpolation of the parameter Hessian. As the expressions become too involved for a general subsystem transfer function, we write and prove the conditions for the first and second subsystems only, but the results can be generalized similarly.

Theorem 4.5. *Assume the hypotheses of Corollary 4.3 for $q = 2$. Define*

$$\begin{aligned}\mathbf{V}_1(\mathbf{p}) &= [\mathbf{K}(\sigma_1;\mathbf{p})\mathbf{B}(\mathbf{p})\mathbf{r}, \quad \mathbf{K}(\sigma_2;\mathbf{p})\mathbf{B}(\mathbf{p})\mathbf{r}], \\ \mathbf{V}_2(\mathbf{p}) &= [\mathbf{K}(\sigma_1;\mathbf{p})\mathbf{N}(\mathbf{p})(\mathbf{I}_m\otimes\mathbf{V}_1(\mathbf{p})), \quad \mathbf{K}(\sigma_2;\mathbf{p})\mathbf{N}(\mathbf{p})(\mathbf{I}_m\otimes\mathbf{V}_1(\mathbf{p}))], \\ \mathbf{W}_1(\mathbf{p}) &= \left[(\mathbf{K}(\sigma_1;\mathbf{p}))^\top \mathbf{C}(\mathbf{p})^\top\boldsymbol{\ell}, \quad (\mathbf{K}(\sigma_2;\mathbf{p}))^\top \mathbf{C}(\mathbf{p})^\top\boldsymbol{\ell}\right], \text{ and} \\ \mathbf{W}_2(\mathbf{p}) &= \left[(\mathbf{K}(\sigma_1;\mathbf{p}))^\top \overline{\mathbf{N}}(\mathbf{p})^\top(\mathbf{I}_m\otimes\mathbf{W}_1(\mathbf{p})), \quad (\mathbf{K}(\sigma_2;\mathbf{p}))^\top \overline{\mathbf{N}}(\mathbf{p})^\top(\mathbf{I}_m\otimes\mathbf{W}_1(\mathbf{p}))\right].\end{aligned}$$

Assume

$$\bigcup_{k=1}^{2}\operatorname{Ran}(\mathbf{V}_k(\boldsymbol{\pi}))\subseteq\operatorname{Ran}(\mathbf{V}) \quad \text{and} \quad \bigcup_{k=1}^{2}\operatorname{Ran}(\mathbf{W}_k(\boldsymbol{\pi}))\subseteq\operatorname{Ran}(\mathbf{W}). \tag{4.38}$$

If either

$$\bigcup_{k=1}^{2}\bigcup_{j=1}^{\nu}\mathrm{Ran}\left(\frac{\partial}{\partial p_j}\mathbf{V}_k(\boldsymbol{\pi})\right)\subseteq\mathrm{Ran}(\mathbf{V})\quad or\quad\bigcup_{k=1}^{2}\bigcup_{j=1}^{\nu}\mathrm{Ran}\left(\frac{\partial}{\partial p_j}\mathbf{W}_k(\boldsymbol{\pi})\right)\subseteq\mathrm{Ran}(\mathbf{W}),\quad(4.39)$$

then

$$\mathcal{H}_{\mathbf{p}}\left(\boldsymbol{\ell}^{\top}\mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_m\otimes\mathbf{r})\right)=\mathcal{H}_{\mathbf{p}}\left(\boldsymbol{\ell}^{\top}\widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_m\otimes\mathbf{r})\right),$$

where $\mathcal{H}_{\mathbf{p}}(\cdot)$ *denotes the Hessian (tensor) with respect to* $\mathbf{p}$.

Proof. The proof will be building on the proofs of Theorems 4.2 and 4.4. We will compute the parameter Hessians of the subsystems, leading to various additive terms, and then show that the subspace conditions will correspond to matching them.

Assume we have the extra conditions on $\mathbf{V}$. First note that we have interpolation of $\mathcal{H}_{\mathbf{p}}\left(\boldsymbol{\ell}^{\top}\mathbf{H}_1(\sigma_i;\boldsymbol{\pi})\mathbf{r}\right)$ for $i\in\{1,2\}$ since this is the linear case (see [15] for details). Let p_i and p_j refer to any entries in the parameter vector $\mathbf{p}$. Recall the definition of the second transfer function of the full and reduced model, respectively:

$$\mathbf{H}_2(s_1,s_2;\mathbf{p})=\mathbf{C}(\mathbf{p})\mathbf{K}(s_1;\mathbf{p})\mathbf{N}(\mathbf{p})(\mathbf{I}_m\otimes\mathbf{K}(s_1;\mathbf{p})\mathbf{B}(\mathbf{p}))\text{ and}$$
$$\widetilde{\mathbf{H}}_2(s_1,s_2;\mathbf{p})=\widetilde{\mathbf{C}}(\mathbf{p})\widetilde{\mathbf{K}}(s_1;\mathbf{p})\widetilde{\mathbf{N}}(\mathbf{p})(\mathbf{I}_m\otimes\widetilde{\mathbf{K}}(s_1;\mathbf{p})\widetilde{\mathbf{B}}(\mathbf{p})).$$

We take the second partial derivative of $\widetilde{\mathbf{H}}_2(s_1,s_2;\mathbf{p})$ with respect to p_j and p_i, apply the definition of the reduced order matrices, rearrange the terms and use the notation in previous

proofs to obtain

$$
\begin{aligned}
&\frac{\partial^2}{\partial p_j \partial p_i}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_m \otimes \mathbf{r})\right) \\
&= \boldsymbol{\ell}^\top \mathbf{C}_{p_j p_i}(\boldsymbol{\pi}) \mathbf{V} \widetilde{\mathbf{f}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi}) && (4.40)\\
&\quad + \widetilde{\mathbf{g}}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{B}_{p_j p_i}(\boldsymbol{\pi}) \mathbf{r} && (4.41)\\
&\quad + \widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{N}_{p_j p_i}(\boldsymbol{\pi})(\mathbf{I}_m \otimes \mathbf{V} \widetilde{\mathbf{f}}_1(\sigma_1;\boldsymbol{\pi})) && (4.42)\\
&\quad + \widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{K}_{p_j p_i}(\sigma_2;\boldsymbol{\pi})^{-1} \mathbf{V} \widetilde{\mathbf{f}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi}) && (4.43)\\
&\quad + \widetilde{\mathbf{g}}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_m \otimes \mathbf{W}^\top \mathbf{K}_{p_j p_i}(\sigma_1;\boldsymbol{\pi})^{-1} \mathbf{V} \widetilde{\mathbf{f}}_1(\sigma_1;\boldsymbol{\pi})) && (4.44)\\
&\quad + \{\widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{N}_{p_j}(\boldsymbol{\pi}) + \widetilde{\mathbf{g}}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{K}_{p_j}(\sigma_1;\boldsymbol{\pi})^{-1}\}(\mathbf{I}_m \otimes \mathbf{V}[\widetilde{\mathbf{f}}_1]_{p_i}(\sigma_1;\boldsymbol{\pi})) && (4.45)\\
&\quad + \{\widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{N}_{p_i}(\boldsymbol{\pi}) + \widetilde{\mathbf{g}}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{K}_{p_i}(\sigma_1;\boldsymbol{\pi})^{-1}\}(\mathbf{I}_m \otimes \mathbf{V}[\widetilde{\mathbf{f}}_1]_{p_j}(\sigma_1;\boldsymbol{\pi})) && (4.46)\\
&\quad + \{\boldsymbol{\ell}^\top \mathbf{C}_{p_j}(\boldsymbol{\pi}) + \widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{K}_{p_j}(\sigma_2;\boldsymbol{\pi})^{-1}\} \mathbf{V}[\widetilde{\mathbf{f}}_2]_{p_i}(\sigma_1,\sigma_2;\boldsymbol{\pi}) && (4.47)\\
&\quad + \{\boldsymbol{\ell}^\top \mathbf{C}_{p_i}(\boldsymbol{\pi}) + \widetilde{\mathbf{g}}_1^\top(\sigma_2;\boldsymbol{\pi}) \mathbf{W}^\top \mathbf{K}_{p_i}(\sigma_2;\boldsymbol{\pi})^{-1}\} \mathbf{V}[\widetilde{\mathbf{f}}_2]_{p_j}(\sigma_1,\sigma_2;\boldsymbol{\pi}), && (4.48)
\end{aligned}
$$

where $\mathbf{M}_{p_j p_i}(\mathbf{p})$ denotes the second partial derivative of $\mathbf{M}(\mathbf{p})$ with respect to p_j and p_i, and $[\mathbf{f}_k]_{p_i}$ denotes the partial derivative of $\mathbf{f}_k$ with respect p_i. Then, we follow the similar manipulations used in the proof of Theorem 4.4 for (4.34)-(4.37): Equations (4.40)-(4.44) contain the same terms, and thus we follow the same reasoning that we used for (4.34)-(4.37). Then, even though (4.45)-(4.48) contain the new terms $[\mathbf{f}_1]_{p_j}(\sigma_1;\boldsymbol{\pi})$ and $[\mathbf{f}_2]_{p_j}(\sigma_1,\sigma_2;\boldsymbol{\pi})$, the same manipulations still apply here due to the construction of $\mathbf{V}$ in (4.38) and (4.39), $[\mathbf{f}_1]_{p_j}(\sigma_1;\boldsymbol{\pi})$ and $[\mathbf{f}_2]_{p_j}(\sigma_1,\sigma_2;\boldsymbol{\pi})$ are now also spanned by $\mathrm{Ran}(\mathbf{V})$ for any p_j. Therefore, we obtain

$$
\frac{\partial^2}{\partial p_j \partial p_i}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_m \otimes \mathbf{r})\right) = \frac{\partial^2}{\partial p_j \partial p_i}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})(\mathbf{I}_m \otimes \mathbf{r})\right).
$$

See Figures 4.9 and 4.10 to follow some of the steps above led by colours. The manipulations

$$
\begin{aligned}
&\frac{\partial^2}{\partial p_i \partial p_j}\left(\boldsymbol{\ell}^\top \mathbf{H}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{r}\right)\\
&= \frac{\partial}{\partial p_i}\Big(\boldsymbol{\ell}^\top \mathbf{C}_{p_j}(\boldsymbol{\pi})\underbrace{\mathbf{K}(\sigma_2;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{f}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})}\\
&\quad - \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma_2;\boldsymbol{\pi})}_{\mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})}\mathbf{K}_{p_j}(\sigma_2;\boldsymbol{\pi})^{-1}\underbrace{\mathbf{K}(\sigma_2;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{f}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})}\\
&\quad + \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma_2;\boldsymbol{\pi})}_{\mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})}\mathbf{N}_{p_j}(\boldsymbol{\pi})\underbrace{\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{f}_1(\sigma_1;\boldsymbol{\pi})}\\
&\quad - \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma_2;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})\mathbf{K}(\sigma_1;\boldsymbol{\pi})}_{\mathbf{g}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})}\mathbf{K}_{p_j}(\sigma_1;\boldsymbol{\pi})^{-1}\underbrace{\mathbf{K}(\sigma_1;\boldsymbol{\pi})\mathbf{B}(\boldsymbol{\pi})\mathbf{r}}_{\mathbf{f}_1(\sigma_1;\boldsymbol{\pi})}\\
&\quad + \underbrace{\boldsymbol{\ell}^\top \mathbf{C}(\boldsymbol{\pi})\mathbf{K}(\sigma_2;\boldsymbol{\pi})\mathbf{N}(\boldsymbol{\pi})\mathbf{K}(\sigma_1;\boldsymbol{\pi})}_{\mathbf{g}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})}\mathbf{B}_{p_j}(\boldsymbol{\pi})\mathbf{r}\Big)\\
&= \left(\boldsymbol{\ell}^\top \mathbf{C}_{p_ip_j}(\boldsymbol{\pi}) - \mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})\mathbf{K}_{p_ip_j}(\sigma_2;\boldsymbol{\pi})^{-1}\right)\mathbf{f}_2(\sigma_1,\sigma_2;\boldsymbol{\pi})\\
&\quad + \mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})\mathbf{N}_{p_ip_j}(\boldsymbol{\pi})\mathbf{f}_1(\sigma_1;\boldsymbol{\pi})\\
&\quad - \mathbf{g}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})\left(\mathbf{K}_{p_ip_j}(\sigma_1;\boldsymbol{\pi})^{-1}\mathbf{f}_1(\sigma_1;\boldsymbol{\pi}) - \mathbf{B}_{p_ip_j}(\boldsymbol{\pi})\mathbf{r}\right)\\
&\quad + \left(\boldsymbol{\ell}^\top \mathbf{C}_{p_j}(\boldsymbol{\pi}) - \mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})\mathbf{K}_{p_j}(\sigma_2;\boldsymbol{\pi})^{-1}\right)[\mathbf{f}_2]_{p_i}(\sigma_1,\sigma_2;\boldsymbol{\pi}) \qquad (4.49)\\
&\quad + \left(\mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})\mathbf{N}_{p_j}(\boldsymbol{\pi}) - \mathbf{g}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{K}_{p_j}(\sigma_1;\boldsymbol{\pi})^{-1}\right)[\mathbf{f}_1]_{p_i}(\sigma_1;\boldsymbol{\pi}) \qquad (4.50)\\
&\quad - [\mathbf{g}_1^\top]_{p_i}(\sigma_2;\boldsymbol{\pi})\left(\mathbf{K}_{p_j}(\sigma_2;\boldsymbol{\pi})^{-1}\mathbf{f}_2(\sigma_1,\sigma_2;\boldsymbol{\pi}) - \mathbf{N}_{p_j}(\boldsymbol{\pi})\mathbf{f}_1(\sigma_1;\boldsymbol{\pi})\right) \qquad (4.51)\\
&\quad - [\mathbf{g}_2^\top]_{p_i}(\sigma_1,\sigma_2;\boldsymbol{\pi})\left(\mathbf{K}_{p_j}(\sigma_1;\boldsymbol{\pi})^{-1}\mathbf{f}_1(\sigma_1;\boldsymbol{\pi}) - \mathbf{B}_{p_j}(\boldsymbol{\pi})\mathbf{r}\right) \qquad (4.52)
\end{aligned}
$$

Figure 4.9: Hessian entry for H_2 with $\mathbf{N}(\mathbf{p}) = \mathbf{N}_1(\mathbf{p})$

To use $\mathbf{V}$ in (4.39), rearrange (4.49) and (4.50) as:

$$
\begin{aligned}
&\left(\boldsymbol{\ell}^\top \mathbf{C}_{p_i}(\boldsymbol{\pi}) - \mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})\mathbf{K}_{p_i}(\sigma_2;\boldsymbol{\pi})^{-1}\right)[\mathbf{f}_2]_{p_j}(\sigma_1,\sigma_2;\boldsymbol{\pi})\\
&\left(\mathbf{g}_1^\top(\sigma_2;\boldsymbol{\pi})\mathbf{N}_{p_i}(\boldsymbol{\pi}) - \mathbf{g}_2^\top(\sigma_1,\sigma_2;\boldsymbol{\pi})\mathbf{K}_{p_i}(\sigma_1;\boldsymbol{\pi})^{-1}\right)[\mathbf{f}_1]_{p_j}(\sigma_1;\boldsymbol{\pi}).
\end{aligned}
$$

To use $\mathbf{W}$ in (4.39), rearrange (4.51) and (4.52) analogously.

Figure 4.10: Visualization of Hessian manipulations in the proof of Theorem 4.5

in Figures 4.9 and 4.10 are analogous to those in Figure 4.8 with more terms to track, since the double partial derivative creates more additive terms. A more careful rearrangement is needed in Figure 4.10, since the derivative information is only assumed in either $\mathbf{V}$ or $\mathbf{W}$. Since p_j and p_i were arbitrary, we obtain the Hessian matching as desired. The proof would be analogous if we assumed the extra conditions on $\mathbf{W}$ instead, only the rearrangement of the terms would change so that the expression depends on $[\mathbf{g}_1]_{p_j}(\sigma_2;\boldsymbol{\pi})$ and $[\mathbf{g}_2]_{p_j}(\sigma_1,\sigma_2;\boldsymbol{\pi})$ instead. □

Remark 4.6. As we stated above, one can write the conditions for matching the parameter Hessian of the higher index subsystems. Let q be the number of subsystems we wish to interpolate. To obtain the parameter Hessian matching for the general k^{th} order subsystem, i.e., to satisfy

$$\mathcal{H}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \mathbf{H}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_m^{\otimes^{k-1}}\otimes\mathbf{r})\right) = \mathcal{H}_{\mathbf{p}}\left(\boldsymbol{\ell}^\top \widetilde{\mathbf{H}}_k(\sigma_1,\ldots,\sigma_k;\boldsymbol{\pi})(\mathbf{I}_m^{\otimes^{k-1}}\otimes\mathbf{r})\right),$$

for $k=1,\ldots,q$, we would need

$$\begin{aligned}
\mathbf{V}_1(\mathbf{p}) &= [\mathbf{K}(\sigma_1;\mathbf{p})\mathbf{B}(\mathbf{p})\mathbf{r}, \quad \cdots, \quad \mathbf{K}(\sigma_q;\mathbf{p})\mathbf{B}(\mathbf{p})\mathbf{r}],\\
\mathbf{V}_k(\mathbf{p}) &= [\mathbf{K}(\sigma_1;\mathbf{p})\mathbf{N}(\mathbf{p})(\mathbf{I}_m\otimes\mathbf{V}_{k-1}(\mathbf{p})), \quad \cdots, \quad \mathbf{K}(\sigma_q;\mathbf{p})\mathbf{N}(\mathbf{p})(\mathbf{I}_m\otimes\mathbf{V}_{k-1}(\mathbf{p}))], \qquad k=1,\ldots,q,\\
\mathbf{W}_1(\mathbf{p}) &= \left[\mathbf{K}(\sigma_1;\mathbf{p})^\top\mathbf{C}(\mathbf{p})^\top\boldsymbol{\ell}, \quad \cdots, \quad \mathbf{K}(\sigma_q;\mathbf{p})^\top\mathbf{C}(\mathbf{p})^\top\boldsymbol{\ell}\right], \text{ and}\\
\mathbf{W}_k(\mathbf{p}) &= \left[\mathbf{K}(\sigma_1;\mathbf{p})^\top\overline{\mathbf{N}}(\mathbf{p})^\top(\mathbf{I}_m\otimes\mathbf{W}_1(\mathbf{p})), \quad \cdots, \quad \mathbf{K}(\sigma_q;\mathbf{p})^\top\overline{\mathbf{N}}(\mathbf{p})^\top(\mathbf{I}_m\otimes\mathbf{W}_1(\mathbf{p}))\right], \quad k=1,\ldots,q,
\end{aligned}$$

(evaluated at $\boldsymbol{\pi}$) to be contained in the ranges of the basis $\mathbf{V}$ and $\mathbf{W}$, respectively, *together with* either the partial derivatives of the $\mathbf{V}_k$'s or the partial derivatives of the $\mathbf{W}_k$'s with respect to the parameter entries (evaluated at $\boldsymbol{\pi}$).

Remark 4.7. In Chapter 2, we noted that our p-AAA algorithm may not be invariant to parameter scaling. However, re-scaling the parameter does not affect our PBMOR method here. Different applications may use the same dynamical system (4.1), but different units

to measure the parameter $\mathbf{p}$. Applying our method to both scenarios will yield the same approximation, provided both systems are sampled at the same frequency values and at the scale-equivalent parameter values. This will provide the same column vectors that define the projection basis $\mathbf{V}$ and $\mathbf{W}$, since these vectors contain the right and left information of the transfer functions and the transfer function values remain invariant under parameter re-scaling (the definition of the system matrices will change, but their value when evaluated at equivalent parameter values will not). That is, if f is the parameter scaling and we use the superscript *new* to denote the system's description corresponding to this scaling, then

$$\mathbf{A}^{new}(\mathbf{p}^{new}) = \mathbf{A}^{new}(f(\mathbf{p})) = \mathbf{A}(\mathbf{p}),$$

and similarly for any other system matrix $(\mathbf{E}(\mathbf{p}), \mathbf{N}_j(\mathbf{p}), \mathbf{B}(\mathbf{p}), \mathbf{C}(\mathbf{p}))$ in (4.1). Therefore $\mathbf{V}^{new} = \mathbf{V}$.

4.3 Numerical Examples

In this section, we illustrate the theoretical discussion from Section 4.1 using three examples: a nonlinear RC circuit, an advection-diffusion equation, and a Burgers equation. Throughout this section, $\boldsymbol{\pi}^{(i)}$ (or $\pi^{(i)}$ when the parameter is a scalar) denotes the parameter sampling points we used in constructing the model reduction bases $\mathbf{V}$ and $\mathbf{W}$, and $\hat{\boldsymbol{\pi}}^{(i)}$ (or $\hat{\pi}^{(i)}$) denotes the parameter points (which are not sampled) at which we evaluate both reduced and full models to investigate the accuracy of the reduced model.

4.3.1 A nonlinear RC circuit

We begin with a modified version of a standard benchmark problem for bilinear systems, namely a nonlinear RC circuit [14, 113]. The original benchmark problem leads to a non-parametric bilinear system. We have revised the problem to add parametric dependence. To clearly motivate this parametric dependence, we include the details of the model derivation.

Consider the SISO parametric nonlinear system

$$\begin{cases} \dot{\mathbf{v}}(t;p) = \mathbf{f}(\mathbf{v}(t);p) + \mathbf{b}u(t), \\ y(t;p) = \mathbf{c}^\top \mathbf{v}(t;p), \end{cases} \tag{4.53}$$

where $\mathbf{v}(t;p) \in \mathbb{R}^N$, $\mathbf{b} = \mathbf{c} = [1\ 0\ \cdots\ 0]^\top \in \mathbb{R}^N$,

$$\mathbf{f}(\mathbf{v};p) = \begin{bmatrix} -g(v_1;p) - g(v_1 - v_2;p) \\ g(v_1 - v_2;p) - g(v_2 - v_3;p) \\ \vdots \\ g(v_{k-1} - v_k;p) - g(v_k - v_{k+1};p) \\ \vdots \\ g(v_{N-1} - v_N;p) \end{bmatrix}, \tag{4.54}$$

and

$$g(v;p) = e^{pv} + v - 1. \tag{4.55}$$

System (4.53) models a nonlinear RC circuit with N resistors, where the state variable $\mathbf{v}(t;p)$ is the voltage at each node, $u(t)$ is the input signal to the current source, the output $y(t;p)$ is the voltage between node 1 and ground, and $g(\nu;p)$ gives the current-voltage dependency

at each resistor. We have introduced a parameter dependency $p \in \mathbb{R}$ in the exponential term of this current-voltage dependency, which models the influence of the operating temperature on the current.

Following [14, 113], since the nonlinearity has equilibrium $\mathbf{f}(\mathbf{0}; p) = \mathbf{0}$, we apply Carleman bilinearization to $\mathbf{f}(\mathbf{v}; p) \approx \mathbf{A}_1(p)\mathbf{v} + \mathbf{A}_2(p)(\mathbf{v} \otimes \mathbf{v})$ and a second-order approximation to $g(v; p) \approx (p+1)v + \frac{1}{2}p^2v^2$, leading to an approximation of the nonlinear dynamics (4.53) by the following parametric bilinear system:

$$\begin{cases} \mathbf{E}\dot{\mathbf{x}}(t; p) = \mathbf{A}(p)\mathbf{x}(t; p) + \mathbf{N}\mathbf{x}(t; p)u(t) + \mathbf{b}u(t), \\ y_b(t; p) = \mathbf{c}^\top \mathbf{x}(t; p), \end{cases} \tag{4.56}$$

where

$$\mathbf{x}(t; p) = \begin{bmatrix} \mathbf{v}(t; p) \\ \mathbf{v}(t; p) \otimes \mathbf{v}(t; p) \end{bmatrix}, \qquad \mathbf{E} = \mathbf{I}_{n_{st}}, \quad \mathbf{c} = \mathbf{b} = \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix}, \tag{4.57}$$

$$\mathbf{A}(p) = \begin{bmatrix} \mathbf{A}_1(p) & \mathbf{A}_2(p) \\ \mathbf{0} & \mathbf{A}_1(p) \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{A}_1(p) \end{bmatrix}, \qquad \mathbf{N} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{b} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{b} & \mathbf{0} \end{bmatrix}, \tag{4.58}$$

and $y_b(t; p)$ denotes the output of the full-order parametric bilinear system. Note that the dimension of the bilinear system is $n_{st} = N + N^2$ and the matrices $\mathbf{A}_1(p) \in \mathbb{R}^{N \times N}$ and $\mathbf{A}_2(p) \in \mathbb{R}^{N \times N^2}$ are given by

$$\mathbf{A}_1(p) = (1+p) \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix},$$

and, for $k = 2, \ldots, N-1$,

$$\begin{aligned}
[\mathbf{A}_2(p)]_{(1,1)} &= -p^2, \\
[\mathbf{A}_2(p)]_{(1,2)} &= [\mathbf{A}_2(p)]_{(1,N+1)} = [\mathbf{A}_2(p)]_{(k,(k-2)N+k-1)} = [\mathbf{A}_2(p)]_{(k,(k-1)N+k+1)} \\
&= [\mathbf{A}_2(p)]_{(k,kN+k)} = [\mathbf{A}_2(p)]_{(N,(N-2)N+N-1)} = [\mathbf{A}_2(p)]_{(N,(N-1)N+N)} = \frac{p^2}{2}, \\
[\mathbf{A}_2(p)]_{(1,N+2)} &= [\mathbf{A}_2(p)]_{(k,(k-2)N+k)} = [\mathbf{A}_2(p)]_{(k,(k-1)N+k-1)} = [\mathbf{A}_2(p)]_{(k,kN+k+1)} \\
&= [\mathbf{A}_2(p)]_{(N,(N-2)N+N)} = [\mathbf{A}_2(p)]_{(N,(N-1)N+N-1)} = -\frac{p^2}{2},
\end{aligned}$$

where $[\mathbf{A}_2(p)]_{(i,j)}$ denotes the $(i,j)^{\text{th}}$ entry of $\mathbf{A}_2(p)$. Note that both $\mathbf{A}_1(p)$ and $\mathbf{A}_2(p)$ have the desired affine structure (4.4) with the scalar parametric functions

$$f_1(p) = 1 + p \quad \text{and} \quad f_2(p) = p^2.$$

As in the original benchmark problem, we choose $N = 200$, and thus obtain a parametric bilinear system of dimension $n_{st} = 40200$. We are interested in the parameter range $p \in [0, 70]$, and choose two parameter sampling points, $\pi^{(1)} = 1$ and $\pi^{(2)} = 50$. For each sampling point, we focus on the leading $q = 2$ subsystems. We choose $\{\sigma_1, \sigma_2\}$ by running IRKA on the linearized model (by setting $\mathbf{N} = 0$); i.e, $\{\sigma_1, \sigma_2\}$ correspond to optimal sampling points for the linear model. With these frequencies, we construct the basis $\mathbf{V}_1$ and $\mathbf{W}_1$ (using Theorem 4.5) that guarantees interpolation of $\mathbf{H}_1(s; p)$, $\mathbf{H}_2(s_1, s_2; p)$, and their sensitives for $p = \pi^{(1)} = 1$ and $s = \sigma_1, \sigma_2$. Similarly, we construct $\mathbf{V}_2$ and $\mathbf{W}_2$ for $\pi^{(2)} = 50$. We then construct the global bases $\mathbf{V} = [\mathbf{V}_1 \ \mathbf{V}_2]$ and $\mathbf{W} = [\mathbf{W}_1 \ \mathbf{W}_2]$ and obtain a reduced parametric bilinear model of dimension $\widetilde{n}_{st} = 12$ using the projection described in (4.3); thus we are approximating a parametric bilinear system of dimension $n_{st} = 40200$ by a reduced parametric bilinear model of dimension $\widetilde{n}_{st} = 12$. To test the accuracy of the parametric reduced model, we simulate and compare the outputs of the original nonlinear

Table 4.1: Example 4.3.1 (RC circuit): Relative L_2 output error for various inputs $u(t)$ and non-sampled parameter values $p = \hat{\pi}^{(i)}$.

	Relative Error		
$u(t)$	$\hat{\pi}^{(1)}$	$\hat{\pi}^{(2)}$	$\hat{\pi}^{(3)}$
e^{-t}	2.54×10^{-3}	2.91×10^{-3}	1.53×10^{-3}
$\frac{1}{2}(\cos(5\pi t) + 1)$	2.54×10^{-3}	4.33×10^{-3}	4.57×10^{-3}

model (4.53), the full bilinear model (4.56), and the reduced bilinear model for two different inputs, $u(t) = e^{-t}$ and $u(t) = \frac{1}{2}(\cos(5\pi t)+1)$, and three different parameter values, $\hat{\pi}^{(1)} = 18$, $\hat{\pi}^{(2)} = 40$, and $\hat{\pi}^{(3)} = 62$. Note that these parameter values are not the sampled values; indeed $\hat{\pi}_3 = 62$ is even outside the sampling range $[1, 50]$. Moreover, note that the inputs $u(t) = e^{-t}$ and $u(t) = \frac{1}{2}(\cos(5\pi t) + 1)$ were not used in the model reduction step, i.e., the reduced model is not informed by these choices of excitation. As Figure 4.11 shows, the parametric reduced bilinear system provides a very accurate approximation to the full bilinear model; their responses are almost indistinguishable. Relative L_2 errors[5] in the outputs for our three parameter values $\hat{\pi}^{(1)}$, $\hat{\pi}^{(2)}$, $\hat{\pi}^{(3)}$ are listed in Table 4.1, showing a relative error on the order of 10^{-3}. We also emphasize that the only deviations visible in Figure 4.11 are deviations from the original nonlinear system, due to Carleman bilinearization, and not due to the model reduction step. We also note that even though the responses might look similar for different parameter values, the scales of the outputs are different.

To make the numerical investigations more detailed, we performed a parameter sweep using 10^3 linearly sampled points in the interval $[0, 70]$, then identified the performance of the reduced model measured in terms of the relative L_2 error in the output $y_b(t; p)$ for each of the inputs. We found that for the first input $u(t) = e^{-t}$, in the *worst*-case scenario, the reduced model led to a relative L_2 output error of 6.31×10^{-3}. For the second input $u(t) = \frac{1}{2}(\cos(5\pi t)+1)$, the *worst* performance yielded a relative L_2 output error of 5.96×10^{-3}.

[5] All relative L_2 errors referred to in this example are measured in the time interval $[0, 2]$.

These numbers further illustrate the ability of the reduced model to accurately approximate the full order parametric bilinear system.

4.3.2 Advection-diffusion equation

For our second example consider a model of the transport and diffusion of a passive scalar field T (representing a chemical concentration, temperature, etc.) on the domain

$$\Omega = [-1, 1] \times [-1, 1].$$

The transport of T is controlled using a background velocity field described with two input parameters u_1 and u_2, and two velocity fields $\mathbf{v}_1$ and $\mathbf{v}_2$. Thus the background velocity field is $\mathbf{v}(x, y) = u_1(t)\mathbf{v}_1(x, y) + u_2(t)\mathbf{v}_2(x, y)$. The value of the passive scalar on the boundary of Ω ($\partial\Omega$) is controlled by an input u_3. We model the diffusion using the viscosity parameter p_1, and include a source term centered at $(p_2, p_3) \in \Omega$ with an area of effect described by p_4 given by

$$f(x, y; p_2, p_3, p_4) = \exp\left(-\frac{(x - p_2)^2 + (y - p_3)^2}{p_4}\right).$$

The strength of the source term is controlled by an input u_4.

Our passive scalar field T then satisfies

$$\begin{aligned}
\dot{T}(x, y, t) &= p_1 \Delta T(x, y, t) - \mathbf{v} \cdot \nabla T(x, y, t) + u_4(t) f(x, y; p_2, p_3, p_4), \\
&\qquad (x, y, t) \in \Omega \times (0, \infty), \\
T(x, y, 0) &= T_0(x, y), \quad (x, y) \in \Omega, \\
T(x, y, t) &= u_3(t), \quad (x, y, t) \in \partial\Omega \times (0, \infty).
\end{aligned} \tag{4.59}$$

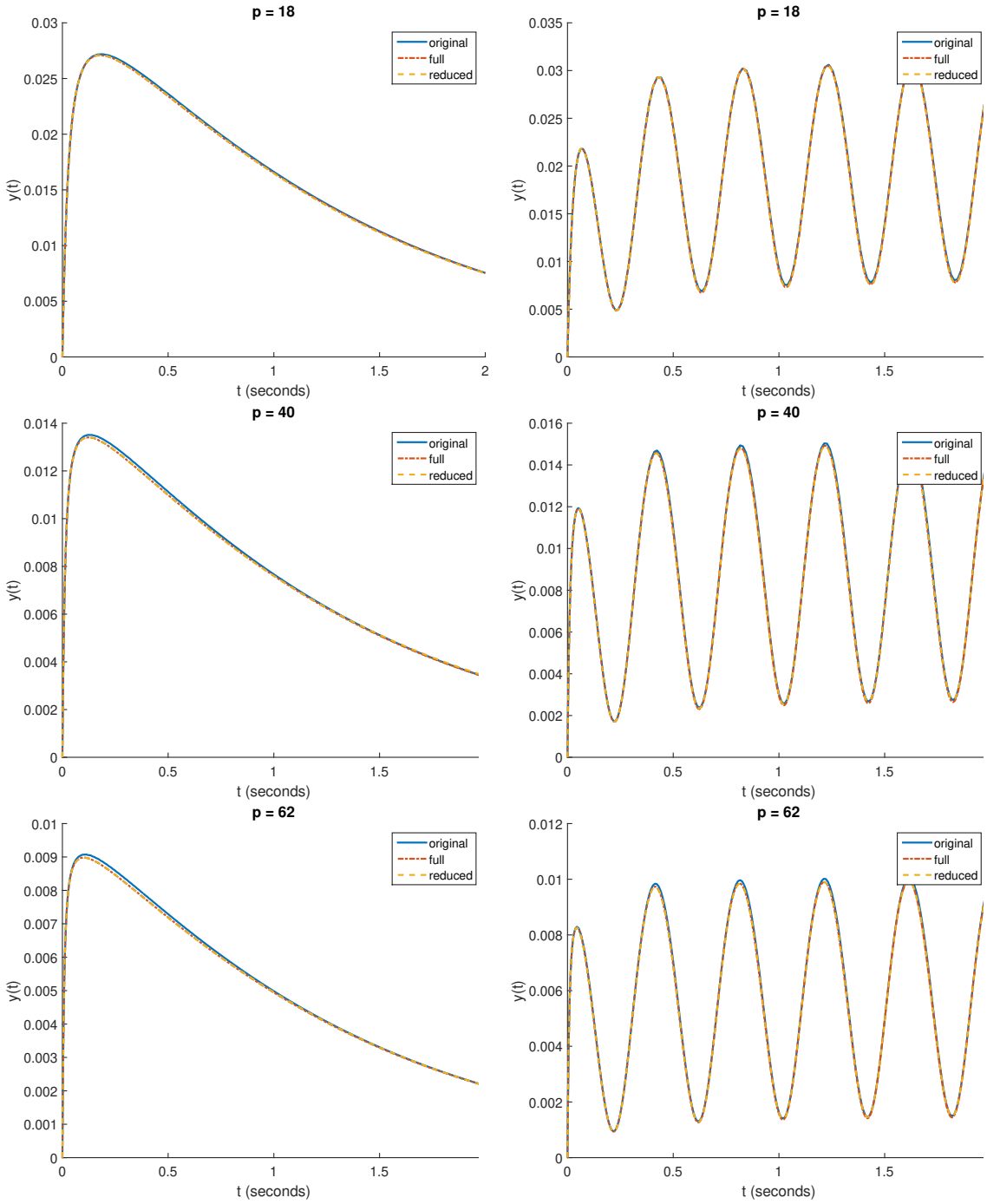


Figure 4.11: Solution to (4.53) (denoted by "original"), (4.56) (denoted by "full"), and reduced model (denoted by "reduced") for different inputs and parameter values. Left column: $u(t) = e^{-t}$. Right column: $u(t) = \frac{1}{2}(\cos(5\pi t) + 1)$.

Thus our model depends on the parameter vector

$$\mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}.$$

We will consider the following as our parameter range:

$$-3 \leq \ln p_1 \leq 1, \qquad (p_2, p_3) \in \Omega, \qquad 1 \leq p_4 \leq 10.$$

We approximate solutions to (4.59) using a finite element discretization

$$T_N(x, y, t) = \sum_{j=1}^{N} x_j(t)\varphi_j(x, y),$$

where the $\{\varphi_j\}_{j=1}^N$ arise from quadratic (P2) triangular elements. For convenience, we will split the summation above into two disjoint parts, one with indices corresponding to boundary nodes ($\mathcal{B}$) and the remainder corresponding to interior nodes ($\mathcal{I}$). Thus $\{1, 2, \ldots, N\} = \mathcal{B} \cup \mathcal{I}$. Upon substituting this into the weak form of (4.59) and suppressing function arguments, we arrive at

$$\left(\frac{\partial}{\partial t}\left[\sum_{j=1}^{N} x_j\varphi_j\right], \varphi_i\right) = -\left(p_1\nabla\left[\sum_{j=1}^{N} x_j\varphi_j\right], \nabla\varphi_i\right) - \left(\mathbf{v}\cdot\nabla\left[\sum_{j=1}^{N} x_j\varphi_j\right], \varphi_i\right) + \left(u_4 f, \varphi_i\right), \qquad \forall i \in \mathcal{I},$$

where the boundary integrals vanish, since φ_i are zero on the boundary when $i \in \mathcal{I}$. Inter-

changing integration in the L_2-inner products with the summation leads to

$$\sum_{j\in\mathcal{I}} (\varphi_j, \varphi_i)\, \dot{x}_j = -p_1 \sum_{j\in\mathcal{I}} (\nabla\varphi_j, \nabla\varphi_i)\, x_j - p_1 u_3 \sum_{j\in\mathcal{B}} (\nabla\varphi_j, \nabla\varphi_i) \\ - u_1 \sum_{j\in\mathcal{I}} (\mathbf{v}_1 \cdot \nabla\varphi_j, \varphi_i)\, x_j - u_2 \sum_{j\in\mathcal{I}} (\mathbf{v}_2 \cdot \nabla\varphi_j, \varphi_i)\, x_j + u_4 (f, \varphi_i),$$

for each $i \in \mathcal{I}$. Letting $\mathbf{x}(t) = [x_1(t)\ x_2(t)\ \ldots\ x_{|\mathcal{I}|}(t)]^\top$ and

$$[\mathbf{E}]_{ij} = (\varphi_i, \varphi_j), \qquad [\mathbf{N}_1]_{ij} = -(\mathbf{v}_1 \cdot \nabla\varphi_j, \varphi_i), \qquad [\mathbf{b}_3]_i = -p_1 \sum_{k\in\mathcal{B}} (\nabla\varphi_i, \nabla\varphi_k),$$

$$[\mathbf{A}]_{ij} = -p_1 (\nabla\varphi_i, \nabla\varphi_j), \qquad [\mathbf{N}_2]_{ij} = -(\mathbf{v}_2 \cdot \nabla\varphi_j, \varphi_i), \qquad [\mathbf{b}_4]_i = (f(\cdot; p_2, p_3, p_4), \varphi_i(\cdot)),$$

for $i, j \in \mathcal{I}$, we can then write our semi-discretized problem as

$$\mathbf{E}\dot{\mathbf{x}}(t;\mathbf{p}) = \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \mathbf{N}_1\mathbf{x}(t;\mathbf{p})u_1(t) + \mathbf{N}_2\mathbf{x}(t;\mathbf{p})u_2(t) + \mathbf{b}_3(\mathbf{p})u_3(t) + \mathbf{b}_4(\mathbf{p})u_4(t),$$

or

$$\mathbf{E}\dot{\mathbf{x}}(t;\mathbf{p}) = \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \sum_{i=1}^{4} \mathbf{N}_i\mathbf{x}(t;\mathbf{p})u_i(t) + \mathbf{B}(\mathbf{p})\mathbf{u}(t),$$

where

$$\mathbf{N} = [\mathbf{N}_1\ \mathbf{N}_2\ \mathbf{N}_3\ \mathbf{N}_4] = [\mathbf{N}_1\ \mathbf{N}_2\ \mathbf{0}\ \mathbf{0}],$$
$$\mathbf{B}(\mathbf{p}) = [\mathbf{0}\ \mathbf{0}\ \mathbf{b}_3(\mathbf{p})\ \mathbf{b}_4(\mathbf{p})], \text{ and}$$
$$\mathbf{u}(t) = [u_1(t)\ u_2(t)\ u_3(t)\ u_4(t)]^\top.$$

We also include an output $\mathbf{y}(t;\mathbf{p}) = \mathbf{c}^\top\mathbf{x}(t;\mathbf{p})$ that represents the average of our scalar field

over $[0.5, 1] \times [0.5, 1]$. In summary, we have a bilinear parametric MIMO system

$$\begin{cases} \mathbf{E}\dot{\mathbf{x}}(t;\mathbf{p}) &= \mathbf{A}(\mathbf{p})\mathbf{x}(t;\mathbf{p}) + \sum_{i=1}^{4} \mathbf{N}_i \mathbf{x}(t;\mathbf{p}) u_i(t) + \mathbf{B}(\mathbf{p})\mathbf{u}(t), \\ y(t;\mathbf{p}) &= \mathbf{c}^\top \mathbf{x}(t;\mathbf{p}), \end{cases} \tag{4.60}$$

which can be reduced using the strategy presented in the previous example.

For our simulations, we chose a 21-by-21 FEM mesh (which results in a FOM of dimension $n = |\mathcal{I}| = 361$) and velocity with

$$\mathbf{v}_1(x,y) = \begin{bmatrix} -y \\ x \end{bmatrix} \quad \text{and} \quad \mathbf{v}_2(x,y) = \frac{\cos(\pi(x-y))}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Note that the full order matrices $\mathbf{E}, \mathbf{N}_1, \mathbf{N}_2, \mathbf{c}$ are constant,

$$[\mathbf{A}(\mathbf{p})]_{ij} = -p_1 [\mathbf{A}]_{ij}, \quad \text{and} \quad [\mathbf{b}_3(\mathbf{p})]_i = -p_1 [\mathbf{b}_3]_i.$$

Hence the ROM is given by

$$\begin{cases} \widetilde{\mathbf{E}}\dot{\widetilde{\mathbf{x}}}(t;\mathbf{p}) &= \widetilde{\mathbf{A}}(\mathbf{p})\widetilde{\mathbf{x}}(t;\mathbf{p}) + \widetilde{\mathbf{N}}_1 \widetilde{\mathbf{x}}(t;\mathbf{p}) u_1(t) + \widetilde{\mathbf{N}}_2 \widetilde{\mathbf{x}}(t;\mathbf{p}) u_2(t) + \widetilde{\mathbf{b}}_3(\mathbf{p}) u_3(t) + \widetilde{\mathbf{b}}_4(\mathbf{p}) u_4(t), \\ \widetilde{y}(t;\mathbf{p}) &= \widetilde{\mathbf{c}}^\top \widetilde{\mathbf{x}}(t;\mathbf{p}), \end{cases} \tag{4.61}$$

where

$$\widetilde{\mathbf{E}} = \mathbf{W}^\top \mathbf{E} \mathbf{V}, \qquad \widetilde{\mathbf{A}}(\mathbf{p}) = -p_1 \mathbf{W}^\top \mathbf{A} \mathbf{V}, \tag{4.62}$$

$$\widetilde{\mathbf{N}}_j = \mathbf{W}^\top \mathbf{N}_j \mathbf{V}, \qquad \widetilde{\mathbf{b}}_3(\mathbf{p}) = -p_1 \mathbf{W}^\top \mathbf{b}_3, \tag{4.63}$$

$$\widetilde{\mathbf{c}}^\top = \mathbf{c}^\top \mathbf{V}, \text{ and} \qquad \widetilde{\mathbf{b}}_4(\mathbf{p}) = \mathbf{W}^\top \mathbf{b}_4(\mathbf{p}). \tag{4.64}$$

Even though the dimension in (4.61) is lower, the reduction of the vector $\mathbf{b}_4(\mathbf{p})$ cannot be done offline since it does not have an affine parametric structure like the rest of the system matrices. Hence we aim to reduce the cost of computing $\mathbf{b}_4(\mathbf{p})$ by means of Q-DEIM approximation, as we discussed in Section 4.1.2. We chose a tolerance of 10^{-5} to truncate the singular values in the POD basis, resulting in a Q-DEIM approximation of order $M = 33$. In Figure 4.12 we show the relative error of the Q-DEIM approximation of $\mathbf{b}_4(\mathbf{p})$

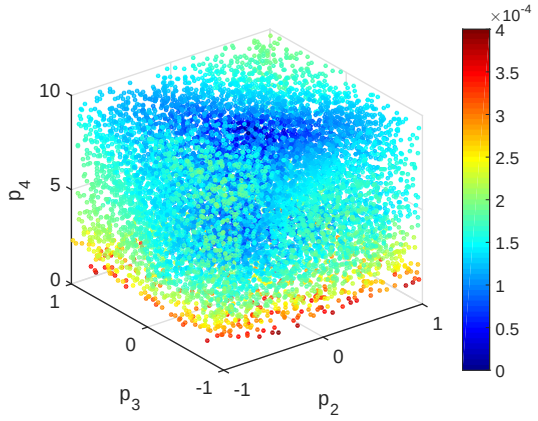


Figure 4.12: Relative error in the Q-DEIM approximation of $\mathbf{b}_4(\mathbf{p})$.

over 10^4 random parameter values in the entire parameter domain. Note that the maximum relative error is on the order of 10^{-4}, showing the accuracy of the Q-DEIM approximation. Thus, we can confidently use $\widetilde{\mathbf{b}}_4(\mathbf{p}) \approx \mathbf{W}^\top \mathbf{U}(\mathbb{S}^\top \mathbf{U})^{-1}\mathbb{S}^\top \mathbf{b}_4(\mathbf{p})$ in our reduced model.

To construct our ROM, we sample at four parameter values $\boldsymbol{\pi}^{(i)}$ for $i = 1, 2, 3, 4$ (see the leading four rows in Table 4.2). We calculate the corresponding projection matrices $\mathbf{V}_1$, $\mathbf{V}_2$, $\mathbf{V}_3$, $\mathbf{V}_4$, $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{W}_3$, and $\mathbf{W}_4$ using Theorem 4.5 that guarantee interpolation and sensitivity matching at the frequency interpolation points (generated via IRKA once again) and tangential directions corresponding to each parameter value sampled. To maintain symmetry in $\widetilde{\mathbf{E}}$ and $\widetilde{\mathbf{A}}$, we concatenate all of the projection matrices and consider a one-sided projection, i.e., $\mathbf{V} = [\mathbf{V}_1\ \mathbf{V}_2\ \mathbf{V}_3\ \mathbf{V}_4\ \mathbf{W}_1\ \mathbf{W}_2\ \mathbf{W}_3\ \mathbf{W}_4]$ and $\mathbf{W} = \mathbf{V}$. We truncate the basis (using SVD) and obtain a ROM with dimension $\widetilde{n}_{st} = 20$.

To illustrate the accuracy of the reduced model, we test it for two different inputs $\hat{\mathbf{u}}^{(i)}$ for $i = 1, 2$ (see Table 4.3) and for four different parameter samples $\hat{\boldsymbol{\pi}}^{(i)}$ for $i = 1, 2, 3, 4$ (see last four rows in Table 4.2) that were not part of the sampling set. We show the results,

Table 4.2: Example 4.3.2 (AD model): Parameter values sampled $\mathbf{p} = \boldsymbol{\pi}^{(i)}$ and parameter values tested $\mathbf{p} = \hat{\boldsymbol{\pi}}^{(i)}$.

	Parameter values		
	viscosity	source center	source reach
$\mathbf{p}$	p_1	(p_2, p_3)	p_4
$\boldsymbol{\pi}^{(1)}$	0.1	$(0.25, 0.8)$	1
$\boldsymbol{\pi}^{(2)}$	1	$(0, 0)$	9
$\boldsymbol{\pi}^{(3)}$	e^{-3}	$(1, 1)$	4
$\boldsymbol{\pi}^{(4)}$	e^{-3}	$(-0.5, -1)$	1
$\hat{\boldsymbol{\pi}}^{(1)}$	0.0529	$(0.975, 0.9275)$	1.6636
$\hat{\boldsymbol{\pi}}^{(2)}$	0.2392	$(0.6914, 0.3149)$	3.6730
$\hat{\boldsymbol{\pi}}^{(3)}$	0.1261	$(-0.7224, -0.7623)$	5.1100
$\hat{\boldsymbol{\pi}}^{(4)}$	0.0754	$(-0.3214, 0.4988)$	2.816

Table 4.3: Example 4.3.2 (AD model): Input values tested.

	Input entries			
$\mathbf{u}(t)$	$u_1(t)$	$u_2(t)$	$u_3(t)$	$u_4(t)$
$\hat{\mathbf{u}}^{(1)}$	$\sin t$	$\cos t$	-1	0.5
$\hat{\mathbf{u}}^{(2)}$	0.5	0.25	1	-1

the full-order and reduced-order outputs in Figures 4.13 and 4.14 for two different inputs (see Table 4.3). Both figures show that for each input selection (neither of which entered into our transfer function-based model reduction process), the parametric reduced bilinear model provides a high-quality approximation, only showing slight variations at the parameter values that were not sampled.

As in the previous example, to ensure a fair comparison, we performed an exhaustive search via 10^4 uniform random samples in our full parameter domain (except that we fixed the

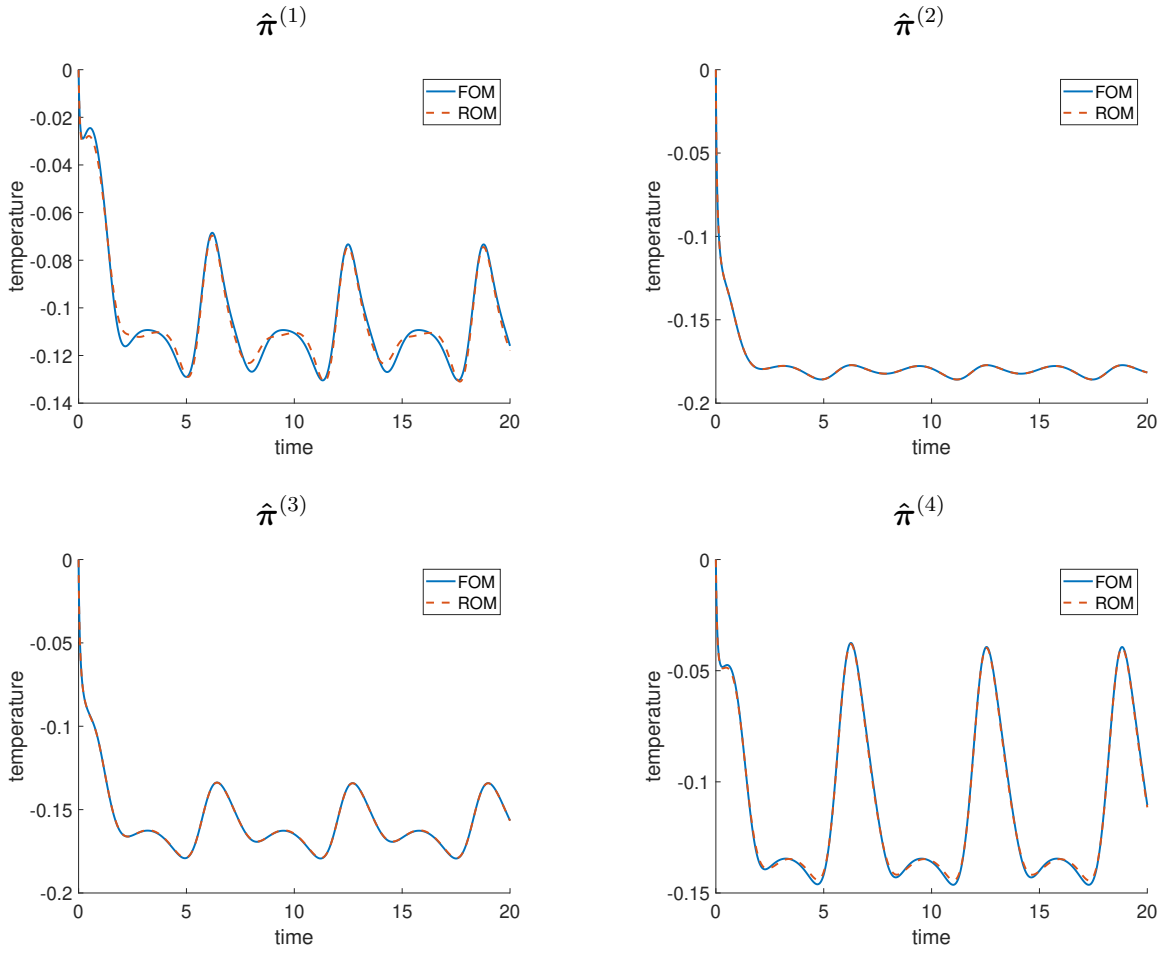


Figure 4.13: Solution of the full (FOM) and reduced-order model (ROM) corresponding to non-sampled parameter values (see values in Table 4.2) for Input 1 with entires $u_1(t) = \sin t$, $u_2(t) = \cos t$, $u_3(t) = -1$, $u_4(t) = 0.5$.

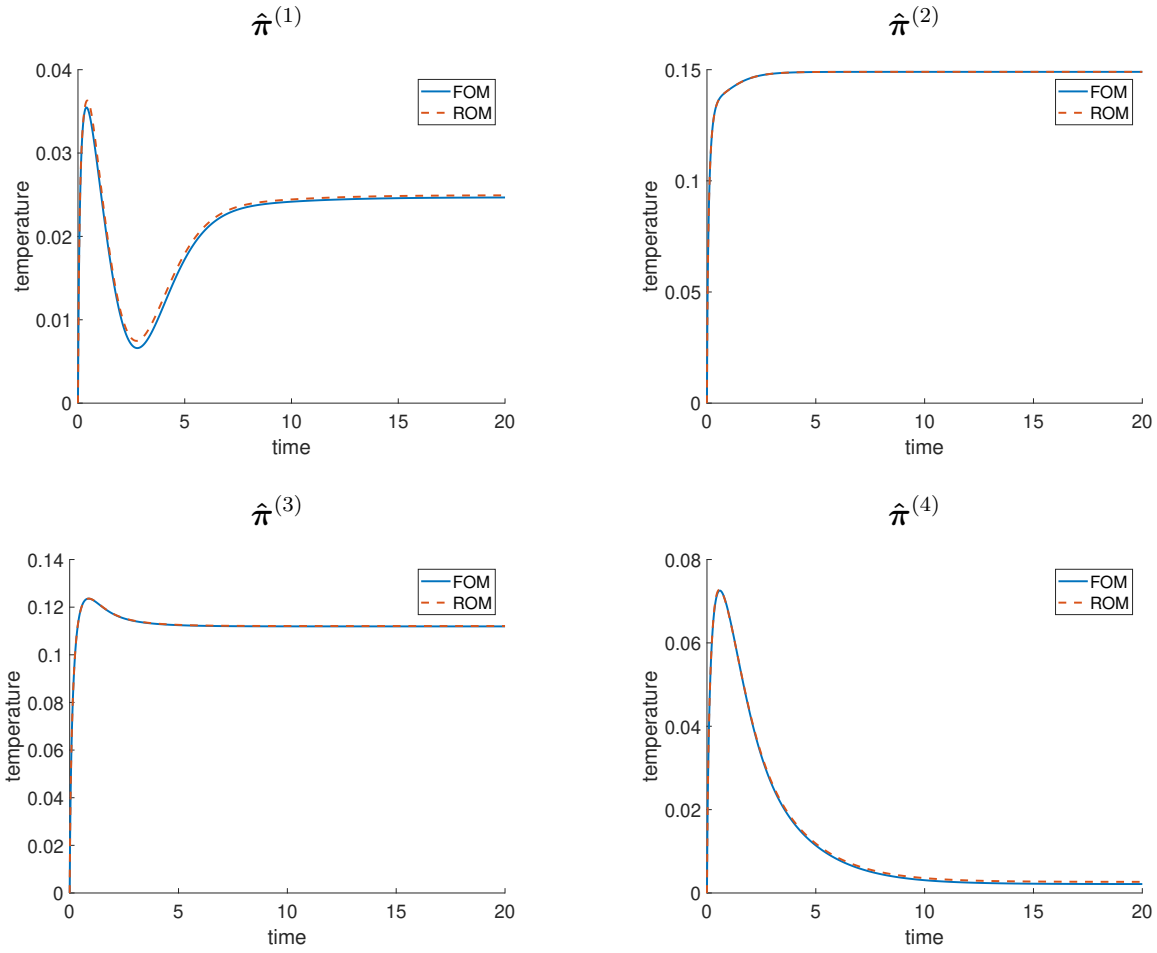


Figure 4.14: Solution of the full (FOM) and reduced-order model (ROM) corresponding to non-sampled parameter values (see values in Table 4.2) for Input 1 with entries $u_1(t) = 0.5$, $u_2(t) = 0.25$, $u_3(t) = 1$, $u_4(t) = -1$.

fourth parameter entry at $p_4 = 5$, so that we can present the results with a 3-dimensional plot) for both input selections from Table 4.3. Out of these 10^4 parameter selections, in Figure 4.15, we display in the top-left plot the relative errors at every sampling point and in the top-right plot, the outputs for the *worst* performance of the reduced model for Input 1. Note that even for the worst parameter sample, the parametric reduced model still provides an accurate approximation with a relative $L_2([0, 20])$ error of 4.79×10^{-2}. We repeat the procedure for Input 2 in the bottom of Figure 4.15 and obtain similar results.

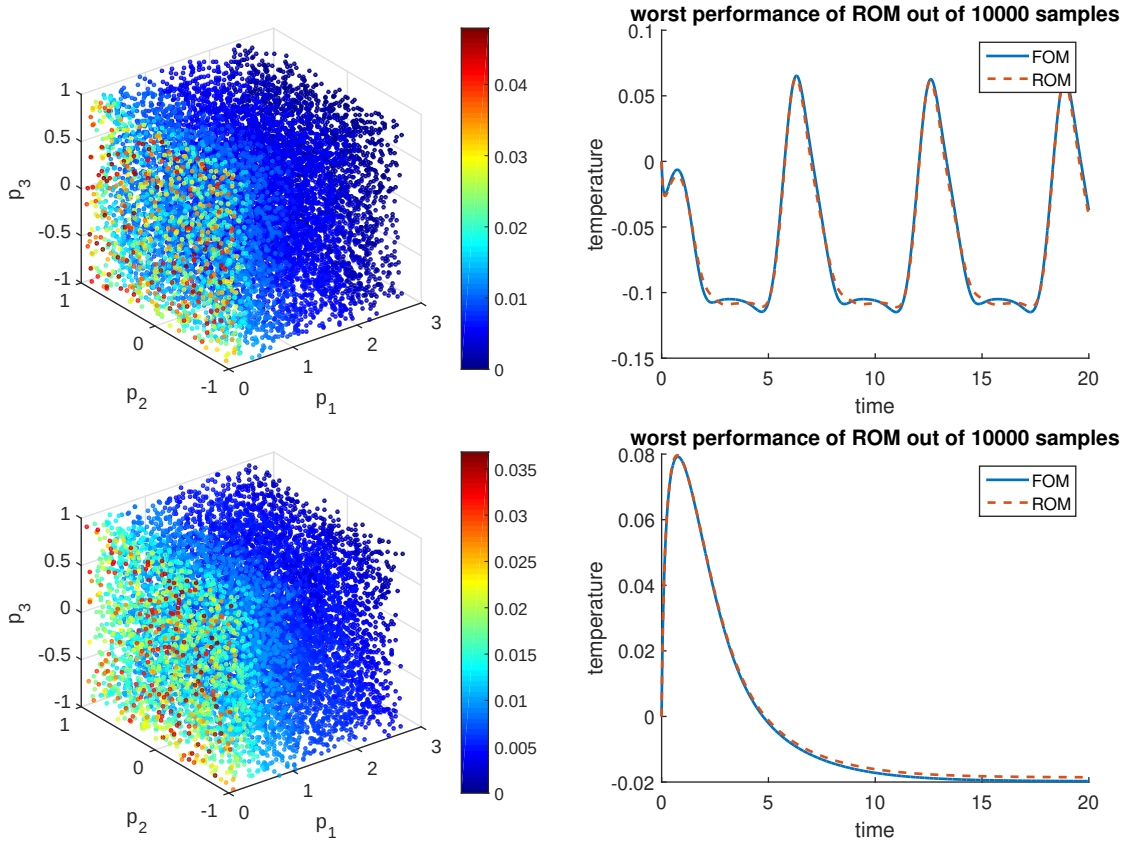


Figure 4.15: Top: Input 1. Bottom: Input 2. Left: relative L_2 output error. Right: solution of the full-order model and the reduced-order model corresponding to the highest relative L_2 error.

4.3.3 Burgers equation

As our final example, we consider the Burgers equation

$$\begin{aligned} w_t(x,t) + w(x,t)w_x(x,t) = pw_{xx}(x,t) \qquad & \text{for } (x,t) \in (0,1) \times (0,T), \\ w(x,0) = 0 \qquad & \text{for } x \in (0,1), \\ w(0,t) = u(t) \qquad & \text{for } t \in (0,T), \\ w(1,t) = 0 \qquad & \text{for } t \in (0,T), \end{aligned}$$

where the input is the time-varying boundary condition at $x = 0$ and the parameter $p > 0$ appears as a constant viscosity. A semi-discretization of the system, performed with central difference approximations of the spatial derivative operators, results in a system of 300 unknowns. We use a spatial discretization and Carleman bilinearization, as in [37], to approximate the Burgers equation as a parametric bilinear system with 90300 $(300 + 300^2)$ states. We define the output as an approximation to the average value of the solution over $(0, 1)$, leading to a SISO system of the form

$$\begin{cases} \mathbf{E}\dot{\mathbf{x}}(t;p) &= (\mathbf{A}_0 + p\mathbf{A}_1)\mathbf{x}(t;p) + (\mathbf{N}_0 + p\mathbf{N}_1)\mathbf{x}(t;p)u(t) + p\mathbf{b}u(t), \\ y(t;p) &= \mathbf{c}^\top \mathbf{x}(t;p), \end{cases}$$

where $\mathbf{E}, \mathbf{A}_0, \mathbf{A}_1, \mathbf{N}_0, \mathbf{N}_1$ are constant matrices; $\mathbf{b}, \mathbf{c}$ are constant vectors; and p represents the viscosity parameter. For details on the structure of the state-space matrices, we refer the reader to [37]. Note that the parametrization appears in the affine form; thus the projection can be applied without the need for DEIM approximation.

Our low-dimensional model reduction bases $\mathbf{V}$ and $\mathbf{W}$ were generated by sampling the vis-

cosity parameter at

$$\pi^{(1)} = 0.05, \qquad \pi^{(2)} = 0.1, \qquad \text{and} \qquad \pi^{(3)} = 1,$$

and the frequencies at

$$\sigma^{(1)} = 0, \quad \sigma^{(2)} = 1, \quad \sigma^{(3)} = 10, \quad \text{and} \quad \sigma^{(4)} = 100.$$

To avoid instability in the reduced model, we used a one-sided projection. In other words, we formed the appended matrix $\mathbf{Z} = [\mathbf{V}\ \mathbf{W}]$ and used an orthonormal basis for $\mathbf{Z}$ to apply the one-sided projection. Hence our reduced model has dimension $r = 48$ and computed in under 7.4 seconds (`tic` and `toc` in Matlab). To test the accuracy of our reduced model we perform simulations corresponding to non-sampled parameter values

$$\hat{\pi}^{(1)} = 0.025, \qquad \hat{\pi}^{(2)} = 0.5, \qquad \text{and} \qquad \hat{\pi}^{(3)} = 2,$$

for the two most challenging input cases used in [37]:

$$\hat{u}^{(2)}(t) = \frac{\sin(20t) + 1}{2}\, e^{-t} \quad \text{and} \quad \hat{u}^{(3)}(t) = \frac{\text{sign}(\sin(\pi t)) + 1}{2}.$$

Note that $\hat{\pi}^{(1)}$ and $\hat{\pi}^{(3)}$ are *outside* the sampling interval. We show our results in Figure 4.16. The figure shows excellent agreement for the outputs predicted by the full-order simulation and the reduced model, even for the extrapolated values. The relative output errors are provided in Table 4.4. As expected, the largest discrepancy occurs for the smallest viscosity parameter $\hat{\pi}^{(1)} = 0.025$. However, the output tracking is still very accurate for this extrapolated value. To emphasize that we are not necessarily claiming that the reduced model will remain accurate as we choose progressively smaller values of p and as we move further away

from the sampling interval, we test the reduced model for the viscosity value of $\hat{\pi}^{(4)} = 0.01$. In this case, we obtain relative output $L_2([0,5])$ error of 1.10×10^{-3} for $\hat{u}^{(2)}(t)$ and relative output $L_2([0,10])$ error of 1.13×10^{-2} for $\hat{u}^{(3)}(t)$. Thus, for this small, extrapolated value of $\hat{\pi}^{(4)} = 0.01$, we see the model accuracy falls off dramatically (losing two orders of accuracy) though still providing a relative error of about one percent; a rather satisfactory result considering that this value is well outside the sampling interval.

We emphasize that a full-order simulation (of the bilinear model resulting from the Carleman bilinearization) is not required to generate the reduced model, however we carried them out to verify the accuracy of the model reduction approach. The reduced model is created in under 7.4 seconds and can be simulated repeatedly for a fraction of the cost of performing even one full-order simulation. The simulation times for different parameter values and different input functions are shown in Table 4.5.

Table 4.4: Example 4.3.3 (Burgers model): Accuracy of the ROM (relative L_2 output errors) tested for three parameter values $p = \hat{\pi}^{(i)}$ and two inputs $u(t) = \hat{u}^{(i)}$.

	Relative error	
$\hat{\pi}^{(i)}$	$\hat{u}^{(2)}(t)$	$\hat{u}^{(3)}(t)$
2	3.66×10^{-5}	5.24×10^{-5}
0.5	3.52×10^{-5}	6.90×10^{-5}
0.025	2.76×10^{-4}	2.60×10^{-3}

4.4 Summary

In this chapter, we presented conditions that ensure Hermite interpolation for parametric bilinear systems. These conditions also ensure that parametric directional derivatives of the reduced-order transfer functions match the full-order transfer function at given interpolation points and directions. We demonstrate the quality of our model reduction framework using

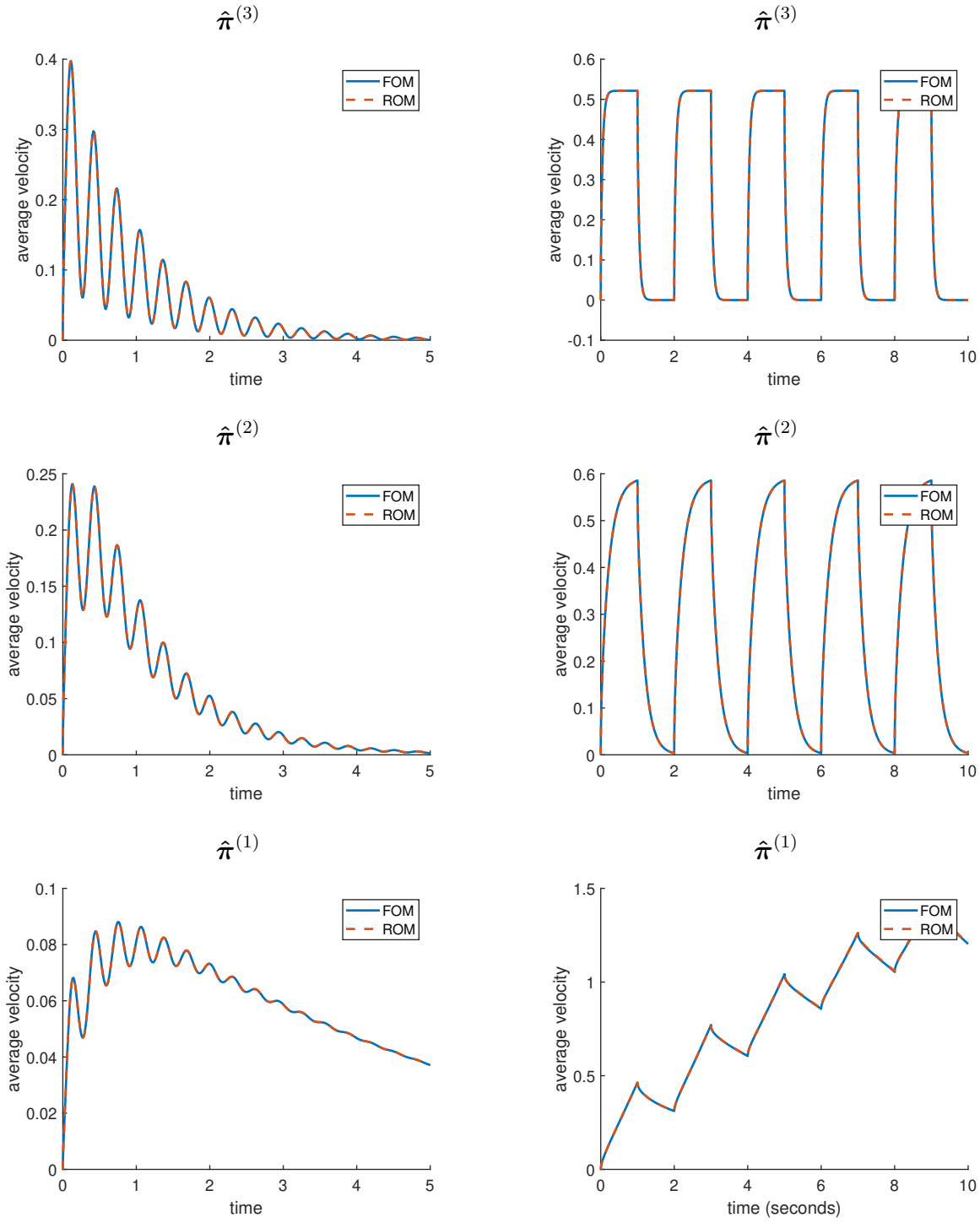


Figure 4.16: Computed outputs at different parameter values corresponding to inputs $\hat{u}^{(2)}(t)$ (left column) and $\hat{u}^{(3)}(t)$ (right column).

Table 4.5: Example 4.3.3 (Burgers model): Efficiency of the ROM vs. FOM. Simulation times shown in seconds.

	Simulation time			
	$\hat{u}^{(2)}(t)$		$\hat{u}^{(3)}(t)$	
$\hat{\pi}^{(i)}$	FOM	ROM	FOM	ROM
2	92.652718	0.150852	399.516661	0.315591
0.5	103.916925	0.100260	349.661368	0.279811
0.025	95.441945	0.071082	238.741363	0.183144
0.01	83.776454	0.067181	249.964954	0.168451

three examples: a nonlinear RC circuit; an advection-diffusion equation; and a Burgers equation with parameter dependency in the bilinear term. The method yielded high fidelity approximations, and we emphasize that no effort was made to select optimal sample points in parameter space. In fact, this approach is agnostic to the parameter choices, and can be easily embedded in well-known parameter selection schemes. The next natural steps are to test this algorithm with different schemes and more challenging problems, and develop a parametric sampling strategy to minimize a global error measure.

Chapter 5

Conclusions

In this dissertation we considered three problems related to parametric dynamical systems: multivariate rational approximation from data, parametric nonlinear eigenvalue problems, and interpolatory projection model order reduction of parametric bilinear systems.

As the first major contribution, we developed our p-AAA algorithm to build two-variable rational functions from data. Our algorithm depends only on the data. The only decision the user needs to make to use p-AAA is the relative error tolerance that they want the approximation to satisfy, given the data provided. The algorithm automatically chooses the order of the rational approximation, as well as the interpolation points (by greedy search) and the weights (by LS). We discussed the influence of parameter scaling and an approach to apply p-AAA to matrix-valued data, and showed that it provides an approximation whose weights solve a tangential LS problem. We illustrated the success of our algorithm via various numerical examples, including parametric dynamical systems and multivariate parametric stationary problems.

As a second major contribution, we employed our p-AAA algorithm to build rational approximations to solve PNLEVPs. For these approximations, we defined a corresponding CORK linearization. We proved that indeed this linearization has the same eigenvalues as the rational approximation. Furthermore, we proved that the eigenvectors of the rational approximation can be easily recovered from the eigenvectors of the linearization. We illustrated our method with two small examples and showed its performance for a large example

where we applied the CORK algorithm.

As our final and third major contribution, we introduced our generalization of interpolatory projection MOR for parametric bilinear dynamical systems. We proved the necessary conditions to guarantee tangential interpolation of the transfer functions, their first derivatives with respect to frequencies, and their first and second derivatives with respect to parameters. We showed the performance of our method for various numerical examples, including a multi-output example and a system with parameter dependency on the bilinear matrix.

Future directions

It would be interesting to apply our interpolatory MOR method for bilinear systems in the field of optimization, since our construction guarantees interpolation on the sensitivities and these are used to inform optimization steps.

The p-AAA algorithm has much potential since it can be applied to approximate *any* function for which data is available or can be generated, and the approximation it builds, a rational function, is built in a numerically efficient way for computations.

We showed an application of p-AAA to parametric nonlinear eigenvalue problems, where we simply approximate the eigenvalues for a parameter value. It would be interesting to see how this method performs when applied to the study of eigenvalue sensitivity to parameter alterations. For more efficient reevaluations, one could also generalize the CORK algorithm to the parametric case and incorporate the parameter dependency directly in the algorithm (instead of evaluating the approximation first and then applying CORK like we did in Chapter 3).

Continuing in the topic of eigenvalue approximation, p-AAA could maybe be applied to other

eigenvalue problems such as the multi-parameter eigenvalue problem.

Generalizations of the AAA algorithm to cases such as matrix-valued data have been developed recently. One could now combine these generalizations for the p-AAA algorithm as well.

Bibliography

[1] M.I. Ahmad, P. Benner, and P. Goyal. Krylov subspace-based model reduction for a class of bilinear descriptor systems. *J. Comput. Appl. Math.*, 315:303–318, 2017.

[2] S. Al-Baiyat, A.S. Farag, and M. Bettayeb. Transient approximation of a bilinear two-area interconnected power system. *Electric Power Systems Research*, 26(1):11–19, 1993.

[3] S.A. Al-Baiyat and M. Bettayeb. A new model reduction scheme for k-power bilinear systems. In *Decision and Control, 1993., Proceedings of the 32nd IEEE Conference on*, pages 22–27. IEEE, 1993.

[4] D. Amsallem and C. Farhat. An online method for interpolating linear parametric reduced-order models. *SIAM J. Sci. Comput.*, 33(5):2169–2198, 2011.

[5] B.D.Q. Anderson and A.C. Antoulas. Rational interpolation and state-variable realizations. *Lin. Alg. Appl.*, 137–138(0):479–509, September 1990.

[6] A.C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM Publications, Philadelphia, PA, 2005.

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

[8] A.C. Antoulas, D.C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. *Contemporary Mathematics*, 280:193–219, 2001.

[9] A.C. Antoulas, C.A. Beattie, and S. Gugercin. Interpolatory model reduction of large-scale dynamical systems. In *Efficient Modeling and Control of Large-Scale Systems*, pages 3–58. Springer, 2010.

[10] A.C. Antoulas, A.C. Ionita, and S. Lefteriu. On two-variable rational interpolation. *Lin. Alg. Appl.*, 436(8):28890–2915, April 2012.

[11] A.C. Antoulas, I.V. Gosea, and A.C. Ionita. Model reduction of bilinear systems in the Loewner framework. *SIAM J. Sci. Comput.*, 38(5):B889–B916, 2016.

[12] A.C. Antoulas, S. Lefteriu, and A.C. Ionita. A tutorial introduction to the Loewner framework for model reduction. In *Model Reduction and Approximation*, chapter 8, pages 335–376. SIAM, 2017.

[13] A.C. Antoulas, C. Beattie, and S. Gugercin. *Interpolatory methods for model reduction*. Computational Science and Engineering 21. SIAM, Philadelphia, 2020.

[14] Z. Bai and D. Skoogh. A projection method for model reduction of bilinear dynamical systems. *Lin. Alg. Appl.*, 415:406–425, 2006.

[15] U. Baur, C. Beattie, P. Benner, and S. Gugercin. Interpolatory projection methods for parameterized model reduction. *SIAM J. Sci. Comput.*, 33(5):2489–2518, 2011.

[16] U. Baur, P. Benner, C.A. Beattie, and S. Gugercin. Interpolatory projection methods for parameterized model reduction. *SIAM J. Sci. Comput.*, 33(5):2489–2518, October 2011.

[17] U. Baur, P. Benner, and L. Feng. Model order reduction for linear and nonlinear systems: A system-theoretic perspective. *Arch. Comput. Methods Eng.*, 21(4):331–358, 2014.

[18] C.A. Beattie and S. Gugercin. Model reduction by rational interpolation. In P. Benner, A. Cohen, M. Ohlberger, and K. Willcox, editors, *Model Reduction and Approximation: Theory and Algorithms. Available as http://arxiv.org/abs/1409.2140*. SIAM, Philadelphia, PA, USA, 2017.

[19] B. Benner, C. Himpe, and T. Mitchell. On reduced input-output dynamic mode decomposition. *Adv. Comput. Math.*, 44:1751–1768, 2018.

[20] P. Benner and T. Breiten. Interpolation-based $\mathcal{H}_2$-model reduction of bilinear control systems. *SIAM J. Matrix Anal. Appl.*, 33(3):859–885, 2012.

[21] P. Benner and T. Damm. Lyapunov equations, energy functionals, and model order reduction of bilinear and stochastic systems. *SIAM J. Control Optim.*, 49(2):686–711, 2011.

[22] P. Benner and L. Feng. A robust algorithm for parametric model order reduction based on implicit moment matching. In A. Quarteroni and G. Rozza, editors, *Reduced Order Methods for Modeling and Computational Reduction*, volume 9 of *MS&A – Modeling, Simulation and Applications*, pages 159–185. Springer, 2014.

[23] P. Benner and P. Goyal. Interpolation-based model order reduction for polynomial parametric systems. *arXiv:1904.11891*, 2019.

[24] P. Benner and T. Stykel. Model order reduction for differential-algebraic equations: a survey. In *Surveys in Differential-Algebraic Equations IV*, pages 107–160. Springer, 2017.

[25] P. Benner, T. Breiten, and T. Damm. Generalised tangential interpolation for model reduction of discrete-time MIMO bilinear systems. *International Journal of Control*, 84(8):1398–1407, 2011.

[26] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Rev.*, 57(4):483–531, November 2015.

[27] P. Benner, A. Cohen, M. Ohlberger, and K. Willcox. *Model Reduction and Approximation: Theory and Algorithms*. SIAM, 2017.

[28] P. Benner, T. Damm, and Y.R. Rodriguez Cruz. Dual pairs of generalized Lyapunov inequalities and balanced truncation of stochastic linear systems. *IEEE Transactions on Automatic Control*, 62(2):782–791, 2017.

[29] 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. *arXiv:2002.09726*, 2020.

[30] P. Benner, S. Gugercin, and S. Werner. Structure-preserving interpolation of bilinear control systems. *arXiv:2005.00795*, 2020.

[31] G. Berkooz, P. Holmes, and J.L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[32] M. Berljafa and S. Güttel. The RKFIT algorithm for nonlinear rational approximation. *SIAM J. Sci. Comput.*, 39(5):2049–2071, 2017.

[33] J.P. Berrut and L.N. Trefethen. Barycentric Lagrange interpolation. *SIAM Rev.*, 46 (3):501–517, August 2004.

[34] T. Betcke, N.J. Higham, V. Mehrmann, C. Schroder, and F. Tisseur. NLEVP: A collection of nonlinear eigenvalue problems. *ACM Trans. Math. Softw.*, 39:7:1–7:28, 2013.

[35] W.J. Beyn. An integral method for solving nonlinear eigenvalue problems. *Lin. Alg. Appl.*, 436:3839–3963, 2012.

[36] W.J. Beyn and V. Thummler. Continuation of invariant subspaces for parameterized quadratic eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 31:1361–1381, 2009.

[37] T. Breiten and T. Damm. Krylov subspace methods for model order reduction of bilinear control systems. *Systems & Control Letters*, 59:443–450, 2010.

[38] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2019.

[39] T. Bui-Thanh, K. Willcox, and O. Ghattas. Model reduction for large-scale systems with high-dimensional parametric input space. *SIAM J. Sci. Comput.*, 30(6):3270–3288, 2008.

[40] T. Carleman. Application de la theorie des equations integrales lineaires aux systemes d'equations differentielles non lineaires. *Acta Mathematica*, 59:63–87, 1932.

[41] A. Carracedo Rodriguez and S. Gugercin. The p-AAA algorithm for data driven modeling of parametric dynamical systems. *arXiv:2003.06536*, 2020.

[42] A. Carracedo Rodriguez, S. Gugercin, and J. Borggaard. Interpolatory model reduction of parameterized bilinear dynamical systems. *Adv. Comput. Math.*, 44(6):1887–1916, 2018.

[43] S. Chaturantabut and D. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.*, 32(5):2737–2764, 2010.

[44] Y. Chen, J. Jiang, and A. Narayan. A robust error estimator and a residual-free error indicator for reduced basis methods. *Computers & Mathematics with Applications*, 77 (7):1963–1979, April 2019.

[45] L. Daniel, O.C. Siong, S.C. Low, K.H. Lee, and J. White. A multiparameter moment matching model reduction approach for generating geometrically parameterized interconnect performance models. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(5):678–693, 2004.

[46] J. Degroote, J. Vierendeels, and K. Willcox. Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis. *International Journal for Numerical Methods in Fluids*, 63(2):207–230, 2010.

[47] Z. Drmač and S. Gugercin. A new selection operator for the discrete empirical interpolation method – improved a priori error bound and extensions. *SIAM J. Sci. Comput.*, 38(2):A631–A648, 2016.

[48] Z. Drmač, S. Gugercin, and C.A. Beattie. Vector fitting for matrix-valued rational approximation. *SIAM J. Sci. Comput.*, 37(5):A2346–A2379, 2015.

[49] Z. Drmač, I. Mezić, and R. Mohr. Data driven modal decompositions: Analysis and enhancements. *SIAM J. Sci. Comput.*, 40(4):A2253–2A2285, 2018.

[50] V. Druskin, C. Lieberman, and M. Zaslavsky. On adaptive choice of shifts in rational Krylov subspace reduction of evolutionary problems. *SIAM J. Sci. Comput.*, 32(5): 2485–2496, August 2010.

[51] C. Effenberger. Robust successive computation of eigenpairs for nonlinear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 34:1231–1256, 2013.

[52] S. Elsworth and S. Güttel. Conversions between barycentric, RKFUN, and Newton representations of rational interpolants. *Lin. Alg. Appl.*, 576(0):246–257, September 2019.

[53] M. Embree and A.C. Ionita. Pseudospectra of Loewner matrix pencils. *arXiv:1910.12153*, 2019.

[54] L. Feng and P. Benner. A new error estimator for reduced-order modeling of linear parametric systems. *IEEE Transactions on Microwave Theory and Techniques*, 67(12): 4848–4859, 2019.

[55] L. Feng and P. Benner. On error estimation for reduced-order modeling of linear non-parametric and parametric systems. *arXiv:2003.14319v2*, 2020.

[56] S. Filip, Y. Nakatsukasa, L.N. Trefethen, and B. Beckermann. Rational minimax approximation via adaptive barycentric representations. *SIAM J. Sci. Comput.*, 40(4): A2427–A2455, August 2018.

[57] G. Flagg and S. Gugercin. Multipoint Volterra series interpolation and $\mathcal{H}_2$ optimal model reduction of bilinear systems. *SIAM J. Matrix Anal. Appl.*, 2015.

[58] G. Golub and V. Pereyra. Separable nonlinear least squares: the variable projection method and its applications. *Inverse Problems*, 19(2):R1–R26, feb 2003.

[59] P. Gonnet, R. Pachón, and L.N. Trefethen. Robust rational interpolation and least-squares. *Electronic Transactions on Numerical Analysis*, 38:146–167, 2011.

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

[61] V. Gosea and S. Gugercin. The AAA framework for modeling linear dynamical systems with quadratic output. *arXiv:2005.10316v1*, 2020.

[62] V. Gosea and S. Güttel. Algorithms for the rational approximation of matrix-valued functions. *arXiv:2003.06410*, 2020.

[63] W.S. Gray and J. Mesko. Energy functions and algebraic gramians for bilinear systems. *IFAC Proceedings Volumes*, 31(17):101–106, 1998.

[64] M.A. Grepl, Y. Maday, N.C. Nguyen, and A.T. Patera. Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. *Mathematical Modelling and Numerical Analysis (M2AN)*, 41(3):575–605, 2007.

[65] S. Grivet-Talocia and B. Gustavsen. *Passive Macromodeling: Theory and applications*, volume 239. John Wiley & Sons, 2015.

[66] S. Gugercin, A.C. Antoulas, and C. Beattie. $\mathcal{H}_2$ model reduction for large-scale linear dynamical systems. *SIAM J. Matrix Anal. Appl.*, 30(2):609–638, 2008.

[67] S. Gugercin, T. Stykel, and S. Wyatt. Model reduction of descriptor systems by interpolatory projection methods. *SIAM J. Sci. Comput.*, 35(5):B1010–B1033, 2013.

[68] P.K. Gunupudi, R. Khazaka, M.S. Nakhla, T. Smy, and D. Celo. Passive parameterized time-domain macromodels for high-speed transmission-line networks. *IEEE Transactions on Microwave Theory and Techniques*, 51(12):2347–2354, 2003.

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

[70] S. Güttel and F. Tisseur. The nonlinear eigenvalue problem. *Acta Numerica*, 26:1–94, 2017.

[71] S. Güttel, R. VanBeeuumen, K. Meerbergen, and W. Michiels. NLEIGS: A class of fully rational Krylov methods for nonlinear eigenvalue problems. *SIAM J. Sci. Comput.*, 36:A2842–A2864, 2014.

[72] Hernandez H., J.E. Roman, and V. Vidal. SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Trans. Math. Software*, 31(3):351–362, 2005.

[73] Voss H. Nonlinear eigenvalue problems. *Handbook of Linear Algebra*, pages 115:1–115:24, 2014.

[74] K.P. Hadeler. Mehrparametrige und nichtlineare eigenwertaufgaben. *Arch. Rational Mech. Anal.*, 27(4):306—328, 1967.

[75] C. Hartmann, B. Schäfer-Bung, and A. Thöns-Zueva. Balanced averaging of bilinear systems with applications to stochastic control. *SIAM J. Control Optim.*, 51(3):2356–2378, 2013.

[76] A. Hay, J. Borggaard, and D. Pelletier. Local improvements to reduced-order models using sensitivity analysis of the proper orthogonal decomposition. *Journal of Fluid Mechanics*, 629:41–72, 2009.

[77] J.S. Hesthaven, G. Rozza, and B. Stamm. *Certified reduced basis methods for parametrized partial differential equations*. Springer Briefs in Mathematics. Springer, Switzerland, 2016. ISBN 978-3-319-22469-5.

[78] J.M. Hokanson. Projected nonlinear least squares for exponential fitting. *SIAM J. Sci. Comput.*, 39(6):A3107–A3128, 2017.

[79] J.M. Hokanson and C.C. Magruder. Least squares rational approximation. *arXiv:1811.12590*, 2018.

[80] A.C. Ionita and A.C. Antoulas. Data-driven parametrized model reduction in the Loewner framework. *SIAM J. Sci. Comput.*, 36(3):A984–A1007, December 2014.

[81] E. Jarlebring, W. Michiels, and K. Meerbergen. A linear eigenvalue algorithm for the nonlinear eigenvalue problem. *Numerische Mathematik*, 122:169–195, 2012.

[82] D. S. Karachalios, I. V. Gosea, Q. Zhang, and A.C. Antoulas. Case study: Approximations of the bessel function. *arXiv:1801.03390*, 2017.

[83] D.S. Karachalios, I.V. Gosea, and A.C. Antoulas. Data-driven approximation methods applied to non-rational functions. *PAMM*, 18(1):e201800368, 2018.

[84] K. Kowalski and W.H. Steeb. *Nonlinear Dynamical Systems and Carleman Linearization*. World Scientific, 1991.

[85] V.N. Kublanovskaya. On an approach to the solution of the generalized latent value problem for lambda-matrices. *SIAM J. Numer. Anal*, 7:532–537, 1970.

[86] J.N. Kutz, S.L. Brunton, B.W. Brunton, and J.L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. SIAM, 2016.

[87] P. Lancaster. Lambda-matrices and vibrating systems. *Pergamon Press. Reprinted by Dover, 2002*, 1966.

[88] C.L. Lawson. Contribution to the theory of linear least maximum approximation. *Ph. D. dissertation, Univ. Calif.*, 1961.

[89] S. Lefteriu and A.C. Antoulas. A new approach to modeling multiport systems from frequency-domain data. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(1):14–27, 2010.

[90] S. Lefteriu and A.C. Antoulas. On the convergence of the vector-fitting algorithm. *IEEE Transactions on Microwave Theory and Techniques*, 61(4):1435–1443, 2013.

[91] E.C. Levy. Complex curve fitting. *IRE Transactions on Automatic Control*, AC-4(1): 37–43, May 1959.

[92] P. Lietaert, J. Perez, B. Vandereycken, and K. Meerbergen. Automatic rational approximation and linearization of nonlinear eigenvalue problems. *arXiv:1801.08622*, 2018.

[93] J.L. Lumley. The Structures of Inhomogeneous Turbulent Flow. *Atmospheric Turbulence and Radio Wave Propagation*, pages 166–178, 1967.

[94] D.S. Mackey, N. Mackey, Mehl, C., and V. Mehrmann. Structured polynomial eigenvalue problems: Good vibrations from good linearizations. *SIAM J. Matrix Anal. Appl.*, 28:1029–1051, 2006.

[95] D.S. Mackey, N. Mackey, and F. Tisseur. Polynomial eigenvalue problems: Theory, computation, and structure. *Numerical Algebra, Matrix Theory, Differential-Algebraic Equations and Control Theory (P. Benner et al., eds), Springer*, pages 319–348, 2015.

[96] A.J. Mayo and A.C. Antoulas. A framework for the solution of the generalized realization problem. *Lin. Alg. Appl.*, 425(2):634–662, 2007.

[97] V. Mehrmann and T. Stykel. Balanced truncation model reduction for large-scale systems in descriptor form. In *Dimension Reduction of Large-Scale Systems*, pages 83–115. Springer, Berlin, 2005.

[98] V. Mehrmann and H. Voss. Nonlinear eigenvalue problems: a challenge for modern eigenvalue methods. *GAMM Mitt.*, 27(2):121–152, 2004.

[99] R.R. Mohler. Natural bilinear control processes. *IEEE Transactions on Systems Science and Cybernetics*, 6(3):192–197, 1970.

[100] R.R. Mohler. Bilinear control processes: with applications to engineering, ecology and medicine. *Mathematics in Science and Engineering*, 106, 1973.

[101] R.R. Mohler. *Nonlinear systems (vol. 2): applications to bilinear control*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1991.

[102] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Transactions on Automatic Control*, 26(1):17–32, 1981.

[103] C. Mullis and R. Roberts. Synthesis of minimum roundoff noise fixed point digital filters. *IEEE Transactions on Circuits and Systems*, 23(9):551–562, 1976.

[104] Y. Nakatsukasa and L.N. Trefethen. An algorithm for real and complex rational minimax approximation. *arXiv:1908.06001*, 2019.

[105] Y. Nakatsukasa, O. Sète, and L.N. Trefethen. The AAA algorithm for rational approximation. *SIAM J. Sci. Comput.*, 40(3):A1494–A1522, December 2018.

[106] Y. Ou. Optimal control of a class of nonlinear parabolic PDE systems arising in fusion plasma current profile dynamics. *PhD Thesis, Lehigh University, Bethlehem, Pennsylvania, USA*, 2010.

[107] H. Panzer, J. Mohring, R. Eid, and B. Lohmann. Parametric model order reduction by matrix interpolation. *at–Automatisierungstechnik*, 58(8):475–484, 2010.

[108] B. Peherstorfer and K. Willcox. Online adaptive model reduction for nonlinear systems via low-rank updates. *SIAM J. Sci. Comput.*, 37(4):A2123—A2150, 2015.

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

[110] B. Peherstorfer, D. Butnaru, K. Willcox, and H.J. Bungartz. Localized discrete empirical interpolation method. *SIAM J. Sci. Comput.*, 36(1):A168—A192, 2014.

[111] B. Peherstorfer, Z. Drmač, and S. Gugercin. Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points. *arXiv:1808.10473*, 2020.

[112] T. Penzl. Algorithms for model reduction of large dynamical systems. *Lin. Alg. Appl.*, 415(2–3):322–343, June 2006.

[113] J.R. Phillips. Projection-based approaches for model reduction of weakly nonlinear, time-varying systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 22:171–187, 2003.

[114] K. Qian and Y. Zhang. Bilinear model predictice control of plasma keyhole pipe welding process. *J. Manuf. Sci. Eng.*, 136(3):031002, 2014.

[115] A. Quarteroni, A. Manzoni, and F. Negri. *Reduced basis methods for partial differential equations: an introduction*. UNITEXT. Springer Cham, 2016. ISBN 978-3-319-15430-5.

[116] W.J. Rugh. *Nonlinear System Theory*. Johns Hopkins University Press Baltimore, MD, 1981.

[117] C. Sanathanan and J. Koerner. Transfer function synthesis as a ratio of two complex polynomials. *IEEE Transactions on Automatic Control*, 8(1):56–58, January 1963.

[118] J. Saputra, R. Saragih, and D. Handayani. Robust $\mathcal{H}_\infty$ controller for bilinear system to minimize HIV concentration in blood plasma. *J.Phys.: Conf. Ser.*, page 1245:012055, 2019.

[119] T. Siu and M. Schetzen. Convergence of Volterra series representation and BIBO stability of bilinear systems. *International Journal of Systems Science*, 22(12):2679–2684, 1991.

[120] S.I. Solov'ëv. Preconditioned iterative methods for a class of nonlinear eigenvalue problems. *Lin. Alg. Appl.*, 415:210–229, 2006.

[121] D. Sorensen and M. Embree. A DEIM induced CUR factorization. *SIAM J. Sci. Comput.*, 38(3):A1454–A1482, 2016.

[122] R. Stefanescu and A. Sandu. A goal-oriented adaptive discrete empirical interpolation method. *arXiv:1901.05343*, 2019.

[123] Y. Su and Z. Bai. Solving rational eigenvalue problem via linearization. *SIAM J. Matrix Anal. Appl.*, 32:201–216, 2011.

[124] F Tisseur and K. Meerbergen. The quadratic eigenvalue problem. *SIAM Rev.*, 43: 235–286, 2001.

[125] R. van Beeumen, K. Meerbergen, and W. Michiels. Compact rational Krylov methods for nonlinear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 2015.

[126] Z. Wang, I. Akhtar, J. Borggaard, and T. Iliescu. Proper orthogonal decomposition closure models for turbulent flows: A numerical comparison. *Computer Methods in Applied Mechanics and Engineering*, 237-240:10–26, 2012.

[127] D.D. Weiner and J.F. Spina. *Sinusoidal Analysis and Modeling of Weakly Nonlinear Circuits: With Application to Nonlinear Interference Effects*. Van Nostrand Reinhold, 1980.

[128] X. Xie, M. Mohebujjaman, L.G. Rebholz, and T. Iliescu. Data-driven filtered reduced order modeling of fluid flows. *SIAM J. Sci. Comput.*, 40(3):B834–B857, 2018.

[129] R. Zimmermann. A locally parametrized reduced-order model for the linear frequency domain approach to time-accurate computational fluid dynamics. *SIAM Journal on Scientific Computing*, 36(3):B508–B537, 2014.

[130] R. Zimmermann. Local parametrization of subspaces on matrix manifolds via derivative information. *SIAM J. Sci. Comput.*, 36(3):B508–B537, 2014.