KRYLOV PROJECTION METHODS
FOR MODEL REDUCTION

BY

ERIC JAMES GRIMME

B.S., The Ohio State University, 1992
M.S., University of Illinois at Urbana-Champaign, 1994

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1997

Urbana, Illinois

ABSTRACT

This dissertation focuses on efficiently forming reduced-order models for large, linear dynamic systems. Projections onto unions of Krylov subspaces lead to a class of reduced-order models known as rational interpolants. The cornerstone of this dissertation is a collection of theory relating Krylov projection to rational interpolation. Based on this theoretical framework, three algorithms for model reduction are proposed. The first algorithm, dual rational Arnoldi, is a numerically reliable approach involving orthogonal projection matrices. The second, rational Lanczos, is an efficient generalization of existing Lanczos-based methods. The third, rational power Krylov, avoids orthogonalization and is suited for parallel or approximate computations. The performance of the three algorithms is compared via a combination of theory and examples. Independent of the precise algorithm, a host of supporting tools are also developed to form a complete model-reduction package. Techniques for choosing the matching frequencies, estimating the modeling error, insuring the model's stability, treating multiple-input multiple-output systems, implementing parallelism, and avoiding a need for exact factors of large matrix pencils are all examined to various degrees.

DEDICATION

To my wife, Kimberly.

ACKNOWLEDGMENTS

I would like to thank all those who supported me throughout my doctoral studies. Foremost on this list are my advisors, Professors Kyle Gallivan and Paul Van Dooren. Paul deserves the credit for starting me on my work in numerical linear algebra. His insights showed me what an interesting field it can be. Many thanks go to Kyle for his support through the latter years of my doctoral research. His encouragement and humor are much appreciated.

I also thank Drs. Danny Sorensen, Steve Ashby and Eli Chiprout for working with me at different times over the past three years. Each was an excellent host who contributed significantly to my education. Danny played a key role in my introduction to iterative Krylov methods. Steve pointed me in the direction of Davidson's method, an approach that implicitly touches many parts of this dissertation. Eli patiently led me through the basics of circuit analysis and asked many stimulating questions.

My gratitude also goes out to my examination committee, Professors Bassam Bamieh, Farid Najm and M. Pai, for their time and comments. Professors Bamieh and Najm also deserve thanks for providing hours of classroom instruction.

Finally, I thank the Department of Energy for its financial support. This dissertation was supported in part by the Computational Science Graduate Fellowship Program of the Office of Scientific Computing in the Department of Energy.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

Iterative projection methods for the solution of large-scale, frequency-dependent problems are introduced in this chapter. Such problems are of growing interest in the analyses or simulations of linear dynamic systems. An outline of the dissertation is provided which highlights the goals in addressing these issues. The notation utilized throughout the dissertation is also summarized.

1.1 Motivating Trends

A surprisingly large variety of physical phenomena is modeled with linear, time-invariant (LTI) dynamic systems. The advantages of this approach include the relative ease by which both the initial model development and the eventual mathematical treatment can be achieved. Models can frequently be acquired through discretizations such as the common finite difference and finite element approaches [1]. A range of techniques from the backward Euler method to multistep methods exists for solving the ordinary differential equations (ODE) that describe the system [2]. Stable, well-understood numerical linear algebra algorithms, e.g., a reduction to Schur form by orthogonal transformations, dominate the low-level mathematical operations [3]. When combined in various fashions, techniques such as the above enable the robust analysis, control or simulation of a large class of physical applications.

Two trends, however, suggest a need for novel iterative approaches for treating LTI dynamic systems, particularly with respect to the linear algebra algorithms. First, many physical models are becoming more complex due to either increased system size or an increased desire for detail. Discretizations of three-dimensional behavior are becoming common. Sources for such applications include the modeling of off-chip (and increasingly on-chip) interconnects in high-speed circuit designs [4]. Simple two-dimensional extractions of resistance and capacitance may no longer be sufficient as minimum feature sizes drop below 0.1 microns and clock speeds exceed 1 GHz. A second example of large-scale systems is a model of the North American power grid arising from planning problems in an increasingly deregulated power industry [5]. Although such models tend to accurately describe the behavior of the underlying physical system, their complexity leads to high analysis and simulation costs with traditional numerical techniques in electrical engineering. Many dense numerical linear algebra techniques are only computationally feasible for a limited number of variables, i.e., on the rough order of hundreds at the time of writing. Popular orthogonal transformation-based approaches in control and eigenvalue computations typically grow cubically in cost and quadratically in memory. Direct sparse factorizations in simulation may be impractical as well due to a highly variable step size or an unexploitable sparsity pattern.

The unrelenting growth of problem complexity is certainly not limited to applications in electrical engineering. It is thus not surprising that a second trend, the proliferation of iterative algorithms, and in particular, Krylov iterative algorithms, has appeared in numerical linear algebra [6, 7, 8]. Suited for sparse or structured problems, these iterative methods frequently lead to approximate numerical solutions with low computational effort. For the most part, however, existing Krylov research focuses on fixed problems. For example, numerous implementations exist for solving a fully specified system of linear equations or finding the eigenvalue closest to some fixed point. Unfortunately, these approaches only extend to frequency-dependent and time-dependent problems in a limited fashion. With regards to dynamic systems, it may be desired to find the system's response over a range of frequencies or to check for unstable eigenvalues in the entire right-half plane. The introduction of time-dependent or frequency-dependent variables presents an exciting and relatively unexplored challenge for Krylov-based approaches.

For the reasons above, this dissertation explores the extension and development of iterative implementations of Krylov projection for the analyses and approximations of

LTI dynamic systems. Although certainly not the first endeavor into this area (see the historical survey in Chapter 2), it is believed that the following represents the most comprehensive treatment of the topic to date. Rather than simply adapting some existing iteration to dynamic systems, we concentrate on deriving a theoretical framework which is then utilized to develop a complete spectrum of novel and powerful Krylov-based methods for dynamic systems.

1.2 Problem Overview

This dissertation emphasizes approximate solution techniques for dynamic system problems involving the matrix pencil $(sE-A)$. The matrices $A \in \mathbb{R}^{N\times N}$ and $E \in \mathbb{R}^{N\times N}$ are assumed to be large and sparse; they typically contain lumped parameters of a large-scale system. The scalar $s \in \mathbb{C}$ denotes complex frequency. Such a matrix pencil arises in several problems in linear system theory [9, 10]. For example, finding the poles of a dynamic system entails computing the generalized eigenvalues of (A, E), i.e., finding the values $\lambda_n \in \mathbb{C}$ and vectors $\mathbf{x}_n \in \mathbb{C}^{N\times 1}$ such that

$$(A - \lambda_n E)\mathbf{x}_n = 0. \tag{1.1}$$

Computing the frequency response of a single-input single-output (SISO) dynamic system requires the solution to either of the dual systems of shifted linear equations,

$$\begin{aligned} (sE - A)\mathbf{x}_b &= b \\ (sE - A)^T\mathbf{x}_c &= c, \end{aligned} \tag{1.2}$$

for many imaginary values of s. Model reduction, finding a low-order approximation for the original dynamic system, is yet another problem involving $(sE-A)$ (see Chapter 2 for a detailed discussion of this problem). This dissertation focuses on the model reduction because it is an increasingly important problem in its own right, it is less studied in the context of iterative methods, and it encompasses many facets of the problems in (1.1) and (1.2).

Traditional techniques for problems involving $(sE-A)$ transform A and E into upper-Hessenberg or upper-triangular form. The generalized Schur decomposition, for example, determines orthogonal T_Q and T_Z such that $T_Q^T A T_Z$ and $T_Q^T E T_Z$ are both upper-triangular [11]. The eigenvalues of an upper-triangular pencil appear immediately. Solving systems of equations in upper-Hessenberg form is dominated by backwards substitutions. The pencil can be rapidly treated at many frequencies once the initial transformation is found. Yet the $O(N^3)$ cost of this initial transformation is prohibitive for large-scale problems. Thus we turn to iterative versions of projection methods. In this family of methods, one is interested in iteratively acquiring a low-order approximation to the matrix pencil, denoted $(s\hat{E} - \hat{A})$. This low-order approximation can then be treated with conventional algorithms to yield estimates for certain eigenvalues, approximate simulated responses to various inputs, or low-order controllers for the original large-scale system. For the most part, this dissertation concentrates on efficiently determining an accurate low-order approximation of a dynamic system. Examples of efforts that begin to study the insertion of the approximation into a design or simulation can be found in [12, 13, 14].

The iterative construction of the low-order approximation depends on a combination of two procedures known as preconditioning and projection (see Section 2.3 for formal definitions of these two terms). Roughly speaking and in the context of frequency-dependent problems, preconditioning speeds the approximation's convergence in chosen frequency regions. In frequency-dependent problems, preconditioning often entails an exact evaluation of the original system at a few discrete points in s. These points are denoted interpolation points. The second component, projection, can be thought of as forming an approximation through the extrapolation of the limited exact information across regions of s. Computing meaningful preconditioners and projectors leads to the problems underlying model reduction. Acquiring an accurate, low-order approximation requires choosing appropriate interpolation points and acquiring sufficient (but not excessive) exact information at each of these points. Computing this exact information from the original large-scale pencil is itself a significant constraint. Additionally, some measure of the quality of the low-order approximation is required. The formal treatment of such subproblems constitutes a successful approach for iteratively approximating dynamic systems and is the bulk of this dissertation.

1.3 Dissertation Goals

The following treatment of Krylov projection methods for dynamic systems strives to be comprehensive. A solid statement of the problems to be examined, a rigorous development of theory, a spectrum of iterative methods, and a range of tools for implementing these iterative methods are sought. Beyond the novel techniques developed in this dissertation, this comprehensive treatment is the main difference between the following and existing work. Our solution approach is not based on the direct transfer of some arbitrary existing iterative method (and its associated constraints) to a given problem at hand. Rather, a theoretical understanding and a complement of intuition are developed which hopefully provide a bigger picture. A framework for existing methods and avenues for entirely new approaches is the result. Moreover, multiple levels of sophistication are presented to balance solution accuracy against computational effort for a large variety of problems.

The cornerstone of this dissertation is a collection of new theory that relates model reduction to the topics of Krylov projection and multiple interpolation points. Out of this core theory, three novel algorithms for model reduction are proposed. Each of these algorithms is related in their ability to approximate information at multiple frequencies. However, their convergence and costs differ. The first algorithm, denoted dual rational Arnoldi, is a two-sided technique that constructs orthogonal bases for unions of Krylov subspaces. The utilized Krylov subspaces of this method are adaptations of those seen in an eigenvalue technique [15]. Due to its emphasis on orthogonalization, this method is extremely robust and is an important contribution to Krylov-based model reduction. However, this robustness is not cheap. A second algorithm, denoted rational Lanczos, is therefore proposed that constructs biorthogonal bases for unions of Krylov subspaces in a two-sided fashion. Rational Lanczos requires only short biorthogonalization recursions and is cost competitive with existing methods based on the Lanczos algorithm. This approach is not particularly suited for either parallelism or perturbations in the constructed subspaces however. A third algorithm, denoted the rational power Krylov method, is therefore also developed. The rational power Krylov method is implemented as a one-sided approach that constructs a union of Krylov subspaces without any sort of orthogonalization or biorthogonalization. Although not rigorously understood and slightly slower to converge, this novel third algorithm is low in cost, highly parallel and amenable to approximations in the constructed Krylov subspaces. The performance of this and the other algorithms are compared via examples. Trade-offs between reliability and speed exist which may be further affected by both the properties of the dynamic system and the availability of computing resources. Independent of the precise algorithm though, a host of supporting tools are also developed in Chapters 5 through 8 to aid in the implementation of a complete model-reduction package. Techniques for choosing the interpolation points (matching frequencies), estimating the modeling error, insuring model stability, treating multiple-input multiple-output (MIMO) systems, implementing parallelism, and avoiding a need for exact factors of the pencil $(A - sE)$ are all considered.

We conclude this section with summaries of the material in each of the remaining chapters.

- *Background.* Chapter 2 describes and motivates the model-reduction problem (dynamic system approximation problem) considered in this dissertation. Two important tools, projection and preconditioning, are explained in the context of dynamic systems. A much needed survey of the existing literature and existing solution approaches is provided.

- *Projection Framework.* Chapter 3 provides a clear framework for understanding all existing Krylov-based modeling methods. Sufficient conditions on Krylov projection are documented in order to achieve model reduction via rational interpolation. The treatment of unstable and/or MIMO models is considered in the projection framework.

- *Projection Method Implementations.* Chapter 4 utilizes the projection framework to present a complete spectrum of iterative algorithms that achieve rational interpolation. These algorithms are analyzed based on their efficiency and numerical stability. The rational Lanczos algorithm, a fast and low-memory iterative algorithm for implementing rational interpolation, is derived.

- *Error Analysis.* Chapter 5 develops two different schemes for approximating the error in the reduced-order model. The performance of these schemes in monitoring the modeling algorithms is analyzed and experimentally verified.

- *Interpolation Point Analysis.* Chapter 6 provides a theoretical understanding of the impact of interpolation point placement and usage on the quality of the reduced-order model. Suggestions are made for utilizing the error analysis methods of Chapter 5 to adapt the interpolation points to a specific problem's qualities. The performance of various interpolation strategies is experimentally verified. Well-defined multipoint interpolation schemes are demonstrated to robustly handle various situations.

- *Parallelism.* Chapter 7 studies the use of parallelism for speeding the construction of reduced-order models. Parallelism can be utilized in conjunction with multiple interpolation points. A version of the rational power method is proposed which avoids large-scale communications between processors. An interpolation point scheme from Chapter 6 is utilized to balance the work between processors.

- *Approximate Solves.* Chapter 8 allows for inexact rational interpolation by relaxing the need for precise factorizations of large-scale matrix pencils. A reduction of the work involved is sought without significant drops in accuracy. Approximation techniques are presented for solving linear systems of equations and connections are drawn between these techniques and the overall model-reduction process. Several examples are provided to illustrate the possibilities with approximate solves.

- *Eigenvalue Problems.* Chapter 9 surveys preconditioned iterative eigenvalue solvers and relates them to the proposed model-reduction techniques. Although providing the initial insight into many of the iterative model-reduction techniques in this dissertation, existing iterative eigenvalue methods are themselves still an active area of research. The primary aim of this chapter is to present links between eigenvalue and model-reduction techniques which can be exploited in future work.

1.4 Notation

This section summarizes the notation used throughout the following chapters. The selected symbols attempt to balance the notation used in the areas of system theory and numerical linear algebra. Many common matrix definitions and operations are summarized in Table 1.1 with respect to the generic matrix G and generic vector g. Commonly used abbreviations in this dissertation are summarized in Table 1.2.

Notation for nearly every letter in the Greek and standard alphabets is summarized in Table 1.3. Although relatively concrete in this table, the exact definition for a given symbol should be taken in the context of the surrounding text. As a general rule, uppercase letters are matrices, lowercase letters are vectors or functions, lowercase Greek letters are scalars, and calligraphic letters are subspaces. There are exceptions to these rules, though, in an attempt to match standard practice. In particular, the letters j to n and J to N correspond to indices.

With regard to functions, the Laplace transform of some time-dependent function is indicated through bold Roman rather than italicized type, e.g., $f(t)$ transforms to $\mathbf{f}(s)$. The explicit dependency of the functions on t or s is dropped where the meaning is obvious, i.e., $f(t)$ may simply be denoted by f and $\mathbf{f}(s)$ by $\mathbf{f}$, if confusion can be avoided.

Table 1.1: Matrix Notation

$\mathbb{R}^{K \times J}, \mathbb{C}^{K \times J}$	sets of real, complex matrices of size K by J
g_j	j^{th} column of the matrix G
G_j	the first j columns of the matrix G
G^T	transpose of G
G^*	complex conjugate of G
$\Lambda(G)$	spectrum of G
$\text{colsp}\,\{G\}$	column space of G
$\text{span}\,\{g_1, \ldots, g_J\}$	span of the vectors $g_1, \ldots, g_J$

Table 1.2: Abbreviations

AWE	asymptotic waveform evaluation
CD	compact disc
CFH	complex frequency hopping
DS	dynamic system
GMRES	generalized minimum residual method
LTI	linear, time-invariant
MIMO	multiple-input multiple-output
MNA	modified nodal admittance
PEEC	partial element equivalent circuit
PIES	preconditioned iterative eigenvalue solvers
QMR	quasi-minimal residual method
RA	rational Arnoldi
RC	resistor, capacitor
RL	rational Lanczos
RK	rational Krylov
RP	rational power
SISO	single-input single-output
SVD	singular value decomposition

Table 1.3: Character Notation

Uppercase Letters

A	left pencil matrix	N	initial system dimension
B	right matrix	O	order of magnitude
C	left matrix	P	preconditioner matrix
D	feed-through matrix	Q	projection matrix
E	right pencil matrix	R	residual matrix
G	generic matrix	S	structured matrix
H	transfer matrix	T	transformation matrix
I	identity matrix	V, W	projection matrices
$J : L$	upper bounds on $j : l$	X	solution matrix
M	reduced-order model size	Z	projection matrix

Lowercase Letters

b	right vector	q	projection vector
c	left vector	r	residual vector
d	feed-through term	s	complex frequency
e	exponential	t	time
f	function	u	input
g	generic vector	v, w	projection vector
h	transfer function	x	solution vector
i	column of an identity	y	output
$j : p$	indices	z	projection vector

Greek Letters

$\alpha : \gamma$	matrix elements	λ	eigenvalue
δ	perturbation	μ	moment
ϵ	error	π	$3.14\ldots$
ζ	evaluation point	ρ	residue
η	generic scalar	σ	interpolation point
ι	imaginary, $\sqrt{-1}$	ϕ, ψ	polynomial coefficients
κ	condition number	ω	real frequency

Calligraphic Letters

$\mathcal{H}$	Hardy norm	$\mathcal{S}$	search subspace
$\mathcal{K}$	Krylov subspace	$\mathcal{T}$	constraint subspace

CHAPTER 2

KRYLOV-BASED MODEL REDUCTION

The primary problem of interest in this dissertation is model reduction: efficiently computing an accurate low-order approximation to a dynamic system. Most techniques for model reduction retain certain invariant features of the original system and several are briefly surveyed in this chapter. The solution approaches utilized in Chapters 3 through 7 retain moments of the original system to yield a reduced-order model known as a rational interpolant. Preconditioning and Krylov projection, tools that are fundamental in Chapter 3 for computing rational interpolants, are explained in detail. This chapter concludes with a history and overview of existing Krylov projection methods for model reduction.

2.1 Problem Statement

This dissertation is primarily devoted to computing low-order approximations to linear dynamic systems. It is assumed that the original system is described by the generalized state-space equations

$$\begin{cases} E\dot{x}(t) = Ax(t) + bu(t) \\ y(t) = c^T x(t) + du(t). \end{cases} \tag{2.1}$$

The vector $x(t) \in \mathbb{R}^{N\times 1}$ is the vector of state variables, $b \in \mathbb{R}^{N\times 1}$ is the input vector of the system and $c \in \mathbb{R}^{N\times 1}$ is the output vector of the system. For simplicity, it is assumed until Section 3.4.2 that the system is single-input single-output so that the input $u(t)$ and output $y(t)$ are scalar functions of time. Finally, and as is the case for nearly all large-scale problems, it is assumed that the system matrix $A \in \mathbb{R}^{N\times N}$ and descriptor matrix $E \in \mathbb{R}^{N\times N}$ are large and sparse or structured.

A reduced-order approximation to (2.1) takes the corresponding form

$$\begin{cases} \hat{E}\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \hat{b}u(t) \\ \hat{y}(t) = \hat{c}^T\hat{x}(t) + du(t). \end{cases} \tag{2.2}$$

The dimension of the reduced-order model is designated as M. The output $\hat{y}(t)$ approximates the true output $y(t)$. However, in general, no simple relation exists between $\hat{x}(t)$ and the state vector $x(t)$. For instance, the tenth element of $\hat{x}(t)$ does not need to be directly related to the tenth element of $x(t)$.

The above generalized, state space expressions are merely one possible representation of linear dynamic systems. A second important representation is the transfer function of a system. Resulting from a Laplace transform of (2.1), the transfer function of the original system is

$$\mathbf{h}(s) = c^T(sE - A)^{-1}b, \tag{2.3}$$

where s represents complex frequency. Without a loss of generality, the feed-through term d of the original model is assumed to be zero (the feed-through term in (2.2) is simply that of the original system and needs no further treatment during model reduction). The function $\mathbf{h}(s)$ maps the Laplace transform of the input $\mathbf{u}(s)$ to the Laplace transform of the output $\mathbf{y}(s)$. The transfer function of the reduced-order model, $\hat{\mathbf{h}}(s)$, can be defined in a manner similar to that in (2.3).

With these preliminaries out of the way, we can now state the prime problem considered in the following: given a large-scale LTI dynamic system, rapidly compute an accurate reduced-order model. Clearly, this statement is vague so that many different solution approaches are possible. In fact, many different model-reduction techniques exist in the literature (see [16] and Section 2.2). The following paragraphs begin to formalize the problem statement, so that various model-reduction methods can be compared.

The key terms in the problem statement are reduced-order, rapidly and accurate. It is hoped that the dimension M of the reduced-order model (2.2) is significantly less than that of the original model N so that (2.2) can be analyzed and/or simulated with relative ease via conventional techniques. Of course, the reduction itself must not be too expensive. If the cost of generating the reduced-order model is comparable to that of directly analyzing (2.1), then little is saved by working with the low-order approximation. Finally, one desires that the reduced-order model is a reasonably accurate approximation of the original model. Because the behavior of the original model is of interest and yet one uses the reduced-order model in its place, the reduced-order system must match the original one in some sense. These conditions of accuracy, speed and order can be conflicting goals. One typically expects, for example, that the accuracy of the reduced-order model increases with a larger order M.

Several measures of the accuracy of the reduced-order model are possible. Formally, there tends to be an interest in the difference between the actual and low-order outputs, $y(t) - \hat{y}(t)$, given some set of inputs $u(t)$. This difference can be characterized via a system norm. The popular $\mathcal{H}_\infty$ error norm, for example, is defined as

$$\begin{aligned} \|\mathbf{h}(s) - \hat{\mathbf{h}}(s)\|_\infty &= \max_{\|u(t)\|_2 = 1} \frac{\|y(t) - \hat{y}(t)\|_2}{\|u(t)\|_2} \\ &= \sup_{\omega} \left((\mathbf{h}(\imath\omega) - \hat{\mathbf{h}}(\imath\omega)(\mathbf{h}(\imath\omega) - \hat{\mathbf{h}}(\imath\omega))^* \right)^{\frac{1}{2}}. \end{aligned}$$

In the time domain, this norm measures the worst ratio of output error energy to input energy [17]. Equivalently, but in the frequency domain, the norm represents the largest magnitude of the frequency-response error. By weighting this norm, one can emphasize the error due to a specific input class of interest.

A second measure of the accuracy of the approximation is to assess which properties of the original model are retained in the reduced-order one. Those properties of interest are said to be invariant, that is, they are independent with respect to a similarity transform. By retaining certain original properties of the system in the reduced-order model, one hopes that the resulting approximation error is small. Of course, this error depends on the selection and pertinence of the retained invariant properties.

Before leaving the problem statement, we note that model reduction is connected to both the generalized eigenvalue problem and the problem of shifted systems of linear equations. Chapter 6 explains that the accuracy of the reduced-order model can be partially connected to the quality of its pole (eigenvalue) approximations. The eigenvalue problem is also important in other areas of system analysis, perhaps most noteworthy is the stability problem. A variety of approaches already exists for computing the nontrivial solutions, eigenvalues λ_n and eigenvectors $\mathbf{x}_n$, to (1.1) over some s region (see for example [15, 18, 19]). Variations on these approaches appear in some of the model-reduction techniques proposed in this dissertation. A survey of pertinent eigenvalue techniques and their connections with model-reduction algorithms is provided in Section 9.1.

Shifted systems of linear equations arise when writing the transfer function as

$$\mathbf{h}(s) = c^T(sE - A)^{-1}(sE - A)(sE - A)^{-1}b = \mathbf{x}_c^T(sE - A)\mathbf{x}_b, \tag{2.4}$$

where $\mathbf{x}_c(\mathrm{s})$ and $\mathbf{x}_b(s)$ are solutions to the dual system of shifted linear equations in (1.2). It is demonstrated in Section 3.2 that the model-reduction approaches taken in this dissertation can be phrased in terms of finding approximate solutions to (1.2) for all values of s. Besides the model-reduction problem, the ability to efficiently solve shifted systems of equations is desirable in certain ODE solvers. A few techniques do exist for iteratively solving shifted systems of equations over a single Krylov subspace [20, 21]. However, these methods are restricted to the case $E = I$ and are limited in their choices of preconditioner. Suitable preconditioners for various regions of the shifted problem are considered in [22], when $E = I$ and A are either symmetric or diagonally dominant. Yet [22] evaluates (2.4) by treating each frequency independently; there is no effort to share information among solves at multiple points.

2.2 Solution Techniques

Many model-reduction methods are based on the retention of invariant properties. A common choice for these invariant properties are the so-called modal properties of the system [23, 24, 25]. The modal properties are based on the system's poles (eigenvalues) λ_n and residues ρ_n, which both arise in a partial fraction expansion of the frequency response,

$$\mathbf{h}(s) = \sum_{n=1}^{N} \frac{\rho_n}{s - \lambda_n}. \tag{2.5}$$

It is assumed for simplicity that the poles of the system are unique. Each of these modal components in the summation contributes a quantity $\rho_n e^{\lambda_n t}$ to the zero-state impulse response of the original system [26]. Hence, a reduced-order model that matches (or approximately matches) specific modal components of the original model retains certain time-dependent features of the original system in its response. Potentially, iterative eigenvalue techniques can be used to find these specific components so that this modal retention approach is feasible for large-scale problems. There are drawbacks to modal-based model reduction, however. It can be difficult to identify a priori which modes are the truly dominant modal components of the original system [27]. The response of the system depends on the interaction of both the poles and residues; locating only the poles near the imaginary axis may not be sufficient.

Alternative invariant properties that may be retained in model reduction are the Hankel singular values. Hankel singular values are related to the controllability and observability properties of a system [28]. Constructing a reduced-order model to retain the largest Hankel singular values is known as balanced truncation. Balanced truncation possesses the desirable feature that the $\mathcal{H}_\infty$ norm of the modeling error is bounded by the sum of the Hankel singular values not retained in the reduced-order model [29]. Unfortunately, implementing balanced truncation involves the solution of Lyapunov equations and thus, a cost of $O(N^3)$ operations [30].

The invariant properties of importance in this work are the coefficients of some power series expansion of $\mathbf{h}(s)$. The solution techniques proposed determine a reduced-order model that accurately matches the leading coefficients μ_j arising in a chosen power series. An expansion of $\mathbf{h}(s)$ about infinity takes the form

$$\mathbf{h}(s) = d + \mu_{-1}s^{-1} + \mu_{-2}s^{-2} + \mu_{-3}s^{-3} + \ldots \quad .$$

The coefficients, which are known as Markov parameters in this case, can be shown to satisfy $\mu_{-j} = c^T(E^{-1}A)^{j-1}E^{-1}b$ by making use of the Neumann expansion [31],

$$(I - \eta G)^{-1} = \sum_{j=0}^{\infty}(\eta G)^j. \tag{2.6}$$

The Markov parameters are the values of the zero-state impulse response $h(t)$ and subsequent derivatives of the impulse response at $t = 0$. A reduced-order model whose Markov parameters $\hat{\mu}_{-j}$ equal μ_{-j} for $j = 1, 2, \ldots, 2M$ is known as a partial realization [32]. Because the partial realization emphasizes behavior at $t = 0$, such a model may be dominated by the extremely rapidly decaying dynamics of the system. Regretfully, extensions of partial realizations that accurately reproduce behavior at some later time are not apparent. For this reason, a power series expansion at $s = 0$ is typically favored in the literature,

$$\mathbf{h}(s) = \mu_0 + \mu_1 s + s^2\mu_2 + s^3\mu_3 + \ldots \quad .$$

Assuming, without loss of generality that a feed-through term is absent, the coefficients, referred to as moments in this expansion, can be shown through (2.6) to satisfy $\mu_{j-1} = -c^T(A^{-1}E)^{j-1}A^{-1}b$ for $j \geq 1$. These moments are the value and subsequent derivatives of the transfer function $\mathbf{h}(s)$ evaluated at $s = 0$. A reduced-order model whose moments $\hat{\mu}_{j-1} = -\hat{c}^T(\hat{A}^{-1}\hat{E})^{j-1}\hat{A}^{-1}\hat{b}$ equal μ_{j-1} for $j = 1, 2, \ldots, 2M$ is known as a Padé approximant [33]. By replacing s in the expansion with the shifted variable $s - \sigma$, i.e.,

$$\mathbf{h}(s) = \sum_{j=1}^{\infty}(s - \sigma)^{j-1}\mu_{j-1},$$

one is led to shifted moments,

$$\mu_{j-1} = -c^T\{(A - \sigma E)^{-1}E\}^{j-1}(A - \sigma E)^{-1}b.$$

These shifted moments are the value and subsequent derivatives of $\mathbf{h}(s)$ at a user-specified interpolation point σ. A reduced-order model can typically be found that matches $2M$ moments at σ (there are $2M$ free parameters available in the numerator and denominator of $\hat{\mathbf{h}}(s)$). Beyond a single interpolation point, one may be interested in a reduced-order model that interpolates the frequency response and its derivatives at multiple points. These K possible interpolation points $\{\sigma^{(1)}, \sigma^{(2)}, \ldots, \sigma^{(K)}\}$ are differentiated by their superscripts. The first $2J_1$ moments are matched at $\sigma^{(1)}$, the next $2J_2$ moments are matched at $\sigma^{(2)}$, etc., where $J_1 + J_2 + \ldots + J_K = M$. A model meeting these constraints is denoted a multipoint Padé approximation or a rational interpolant [33, 34]. By varying the location and number of interpolation points utilized with the underlying problem in mind, one can construct accurate reduced-order models in a variety of situations. For quick reference, various moments that can be matched are summarized in Table 2.1. In each case, these moments can be computed with matrix-vector multiplies and matrix inversions (solving systems of linear equations) involving A and E. It is the relative simplicity of these two required operations that favors moment matching for sparse, large-scale problems.

Table 2.1: Moment Choices in Model Reduction

Approximation Names	Power Series Expansion of $\mathbf{h}(s)$	j^{th} Coefficient
Partial Realization Padé at ∞	$\sum_{j=1}^{\infty} \mu_{-j} s^{-j}$	$c^T(E^{-1}A)^{j-1}E^{-1}b$
Padé	$\sum_{j=1}^{\infty} \mu_{j-1} s^{j-1}$	$-c^T(A^{-1}E)^{j-1}A^{-1}b$
Shifted Padé	$\sum_{j=1}^{\infty} \mu_{j-1}(s-\sigma)^{j-1}$	$-c^T\{(A-\sigma E)^{-1}E\}^{j-1}(A-\sigma E)^{-1}b$
Rational Interpolant Multipoint Padé	$\sum_{j_k=1}^{\infty} \mu_{j_k-1}(s-\sigma^{(k)})^{j_k-1}$ $k=1,2,\ldots,K$	$-c^T\{(A-\sigma^{(k)}E)^{-1}E\}^{j_k-1}(A-\sigma^{(k)}E)^{-1}b$ $k=1,2,\ldots,K$

Example 2.1 *To understand the various moment matching possibilities, we conclude with an examination of four different* 15^{th} *order models for a* 120^{th} *order SISO system. This original system describes the dynamics between the lens actuator and the radial arm position of a portable compact disc player [35]. The frequency response corresponding to this system is shown as a solid line in Figures 2.1 and 2.2. The frequency responses of a partial realization (dotted line) and a Padé approximation (dashed line) are in Figure 2.1. The frequency responses of a shifted Padé approximation (dashed line;* $\sigma = 10^4$

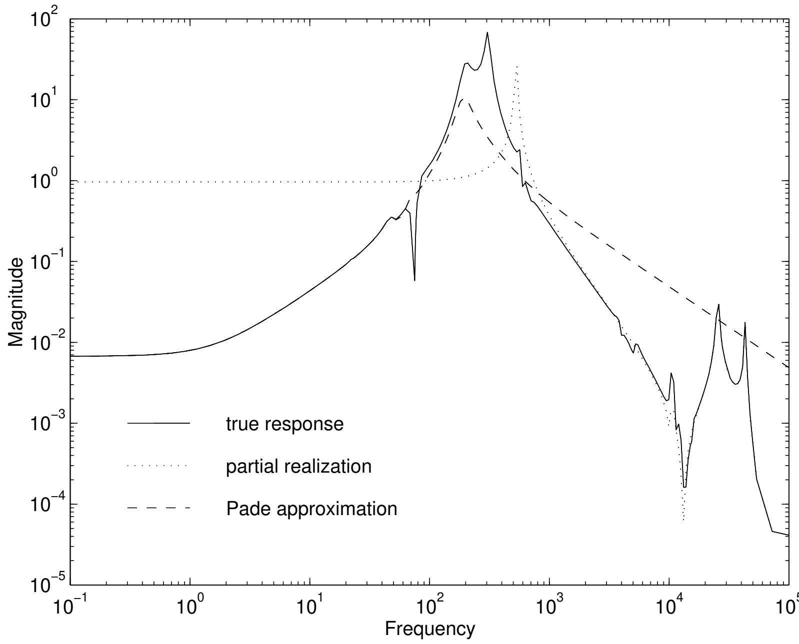

Figure 2.1: Partial Realization and Padé Approximation of Example 2.1

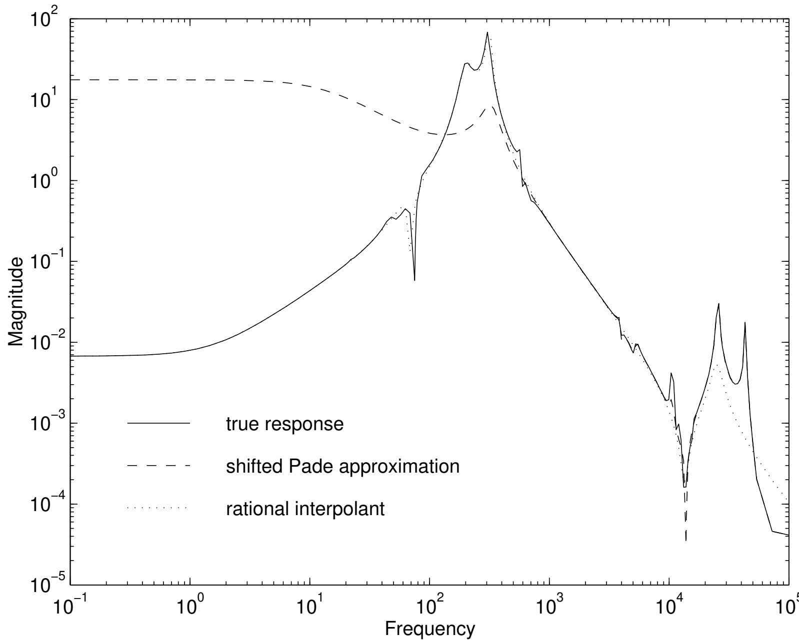

Figure 2.2: Shifted Padé Approximation and Rational Interpolation of Example 2.1

with $J = 15$) and a rational interpolation (dotted line; $\sigma^{(1)} = 1$, $\sigma^{(2)} = 100$, $\sigma^{(3)} = 10^4$ with $J_1 = J_2 = J_3 = 5$) are shown in Figure 2.2. Note that the partial realization captures only higher frequency behavior, while the accuracy of the two single-point Padé approximations is directly related to the choice of σ. The best results are acquired with the more general rational interpolant; the frequency response of the rational interpolant is nearly indistinguishable from that of the original for all but very high frequencies.

Example 2.1 clearly exhibits the behaviors expected from each of the moment matching methods. In particular, rational interpolation matches information over a range of frequencies. This does not mean that the use of a larger number of interpolation points is always necessary or wise. Rational interpolation requires the inversion (triangular factorization) of $(A - \sigma^{(k)}E)$ at every one of the interpolation points (see Table 2.1). Thus, there are extra fixed costs involved in going to multiple interpolation points; yet these costs may be offset by the resulting improvements in convergence. Balancing the number of interpolation points, the placement of these points, and the order M, can be crucial for the efficient calculation of an accurate, low-order model.

2.3 Implementation Techniques

The previous section defined several model-reduction techniques including the family of moment matching methods. Actually implementing algorithms to yield the desired reduced-order models remains. For moment matching, there are in fact several different avenues of implementation. One path is the explicit approach in Section 2.4.2. However, this dissertation concentrates on the utilization of projection and preconditioning in the proposed implementations. The obtainable benefits of this path include numerical stability and the opportunity for iterative implementations.

2.3.1 Projection

A primary tool in Chapters 3 and 4 is projection. Projection extracts an approximate solution of dimension M from a search subspace $\mathcal{S}$. In order to be precisely defined, this approximation is chosen from $\mathcal{S}$ so that M constraints are satisfied. The subspace $\mathcal{T}$ is associated with these constraints. For example, we typically require that the approximate solution is chosen from $\mathcal{S}$ so that its residual is orthogonal to a specified $\mathcal{T}$. Such constraints are known as Petrov-Galerkin conditions. If this $\mathcal{T}$ equals $\mathcal{S}$, then the projection is orthogonal; otherwise, the projection is said to be oblique. For a more detailed review of the projection technique, refer to [8].

The subspaces $\mathcal{S}$ and $\mathcal{T}$ can be represented via rectangular matrices $V \in \mathbb{R}^{N\times M}$ and $Z \in \mathbb{R}^{N\times M}$, whose columns form bases for the respective subspaces. It is important to note that there are infinitely many V and Z whose columns are acceptable bases for given $\mathcal{S}$ and $\mathcal{T}$. For now, it is only necessary to know that V and Z satisfy colsp $\{V\} = \mathcal{S}$ and colsp $\{Z\} = \mathcal{T}$. In fact, this knowledge is sufficient in theory, because all possible choices for the bases lead to identical results up to a similarity transformation. In Chapter 4, we begin to analyze specific choices for V and Z in light of the issues of numerical efficiency and accuracy.

In terms of linear dynamic systems, the projection technique is associated with the transform and truncate operations [36]. If nonsingular left and right transformation matrices $T_l^T \in \mathbb{R}^{N\times N}$ and $T_r \in \mathbb{R}^{N\times N}$ are partitioned as

$$T_l = \begin{bmatrix} Z & Z_+ \end{bmatrix} \quad \text{and} \quad T_r = \begin{bmatrix} V & V_+ \end{bmatrix}$$

and applied to (2.1), then the original model can be expressed as

$$\left[\begin{array}{c|c} Z^T EV & Z^T EV_+ \\ \hline Z_+^T EV & Z_+^T EV_+ \end{array}\right] \left[\begin{array}{c} \dot{\hat{x}} \\ \hline \dot{\hat{x}}_+ \end{array}\right] = \left[\begin{array}{c|c} Z^T AV & Z^T AV_+ \\ \hline Z_+^T AV & Z_+^T AV_+ \end{array}\right] \left[\begin{array}{c} \hat{x} \\ \hline \hat{x}_+ \end{array}\right] + \left[\begin{array}{c} Z^T b \\ \hline Z_+^T b \end{array}\right] u$$

and

$$y = \left[\begin{array}{c|c} c^T V & c^T V_+ \end{array}\right] \begin{bmatrix} \hat{x} \\ \hat{x}_+ \end{bmatrix} + du.$$

The reduced-order model is determined by retaining only the leading M by M subsystem of this transformed original system. Hence, the transformation and truncation operations (denoted model reduction by projection) lead to a reduced-order model with components

$$\hat{A} = Z^T AV, \quad \hat{b} = Z^T b, \quad \hat{c} = V^T c, \quad \hat{d} = d, \quad \hat{E} = Z^T EV. \tag{2.7}$$

The quantities in (2.7) are said to be the restrictions of the original system matrices by Z and V.

The concept of model reduction via projection can be connected to the formal concept of projection through (2.4). Analogous to (2.4), the transfer function for the reduced-order model can be written as

$$\hat{\mathbf{h}}(s) = \hat{c}^T(s\hat{E} - \hat{A})^{-1}(s\hat{E} - \hat{A})(s\hat{E} - \hat{A})^{-1}\hat{b}. \tag{2.8}$$

Defining $\hat{\mathbf{x}}_b$ and $\hat{\mathbf{x}}_c$ to be the solutions of the dual reduced-order, shifted systems of equations

$$\begin{aligned}(s\hat{E} - \hat{A})\hat{\mathbf{x}}_b &= \hat{b}\\ (s\hat{E} - \hat{A})^T\hat{\mathbf{x}}_c &= \hat{c},\end{aligned} \tag{2.9}$$

the transfer function (2.8) can be written as

$$\hat{\mathbf{h}}(s) = \hat{\mathbf{x}}_c^T Z^T(sE - A)V\hat{\mathbf{x}}_b. \tag{2.10}$$

Comparing (2.4) to (2.10), one sees that the transfer functions of the original and reduced-order models differ only in that the latter approximates $\mathbf{x}_b$ and $\mathbf{x}_c$ with $V\hat{\mathbf{x}}_b$ and $Z\hat{\mathbf{x}}_c$. In fact, these approximations $V\hat{\mathbf{x}}_b$ and $Z\hat{\mathbf{x}}_c$ satisfy the Petrov-Galerkin conditions for any chosen V and Z. Model reduction via projection computes the same approximations to $\mathbf{x}_b$ and $\mathbf{x}_c$ that would be computed by projection onto $\mathcal{S}$ and $\mathcal{T}$ with Petrov-Galerkin constraints. For all s, the approximate solution vector $V\hat{\mathbf{x}}_b$ lies in colsp$\{V\} = \mathcal{S}$ and has a residual

$$\mathbf{r}_b(s) = b - (sE - A)V\hat{\mathbf{x}}_b \tag{2.11}$$

that is orthogonal to colsp$\{Z\} = \mathcal{T}$, i.e.,

$$Z^T b - Z^T(sE - A)V\hat{\mathbf{x}}_b = \hat{b} - (s\hat{E} - \hat{A})\hat{\mathbf{x}}_b = 0. \tag{2.12}$$

The column spaces of V and Z play a reversed role in acquiring the approximate solution to $\mathbf{x}_c$. The approximate solution vector $Z\hat{\mathbf{x}}_c$ lies in colsp$\{Z\}$ and its residual,

$$\mathbf{r}_c(s) = c - (sE - A)^T Z\hat{\mathbf{x}}_c, \tag{2.13}$$

is orthogonal to colsp$\{V\}$ for all s. Thus, there is a connection between the formal definition of a projection method and model reduction via projection.

Almost all popular model-reduction methods utilize projection. In balanced truncation, V and Z are chosen to correspond to the so-called Hankel singular vectors (and associated largest Hankel singular values) of the system. In modal model-reduction methods, V and Z are chosen to correspond to M left and right eigenvectors of the pencil (A,E) relative to some ordering of the eigenvalues. Thus, the reduced-order model retains exactly M modal components of the original system. Rational interpolation, the method of choice in this dissertation, can also be phrased in terms of projection. Section 3.1 proves in detail that rational interpolation is achieved if the column spaces of V and Z span unions of Krylov subspaces.

A j^{th} dimensional Krylov subspace corresponding to some matrix G and vector g is denoted $\mathcal{K}_j(G,g)$ and is defined as

$$\mathcal{K}_j(G,g) = \text{span}\left\{g, Gg, G^2g, \ldots, G^{j-1}g\right\}.$$

A basis for a Krylov subspace can be quickly computed if G can be rapidly applied to g, e.g., due to sparsity. This fact gives Krylov-based model reduction the potential for cost savings. Yet specifying which Krylov subspace(s) are desirable for the best reduced-order model remains. Extremely simple but not particularly effective choices for V and Z are

$$\begin{aligned}
\text{colsp}\{V(s)\} &= \mathcal{K}_M((A-sE),b) \\
&= \text{span}\left\{b,(A-sE)b,\ldots,\{(A-sE)\}^{M-1}b\right\}, \\
\text{colsp}\{Z(s)\} &= \mathcal{K}_M((A-sE)^T,c) \\
&= \text{span}\left\{c,(A-sE)^Tc,\ldots,\{(A-sE)^T\}^{M-1}c\right\}.
\end{aligned} \tag{2.14}$$

For the time being, we overlook the dependence of the subspaces in (2.14) on s (for now, think of fixing s at some value). The content of these individual Krylov subspaces are classic choices when approximately solving the dual systems of equations in (1.2). For example, the biconjugate gradient method, a well-known iterative Krylov solver, would lead to (2.14) when applied to (1.2) at a fixed s [8]. A second motivation for the form of (2.14) and one more consistent with the model-reduction history is its simplicity. The use of (2.14) for model reduction emphasizes simple sparse matrix-vector products. Finally, the structure of these two Krylov subspaces is consistent with the dual input and output structure of a LTI dynamic system. If E is the identity matrix, then the subspaces in (2.14) are closely related to the controllability and observability spaces of a dynamic LTI system [37].

Chapters 3 and 4 motivate and evaluate alternative choices for Krylov subspaces composing V and Z. Cost, approximation quality, and the ability to handle frequency dependence in the problem are significant concerns which must be addressed by the forms of V and Z.

2.3.2 Preconditioning

Preconditioning is an important partner with projection in iterative solution techniques. In fact, it is frequently credited with being the most important component in efficiently computing an accurate solution. In broad terms, the goal of preconditioning is to generate better projection subspaces (yield faster model convergence) without drastically complicating the construction of the Krylov subspaces.

The preconditioner $P \in \mathbb{R}^{N \times N}$ is traditionally introduced as a fixed left or right transformation. Rather than solving the problem $Ax = b$, for example, one might consider the left-transformed problem $PAx = Pb$. If P exactly equals A^{-1}, then the solution x is trivially Pb. In general, P can be any matrix that transforms the original problem to a description that is hopefully easier to (iteratively) solve. In frequency-dependent problems, approaches very similar to traditional preconditioning can be used. For example, one can transform the matrix pencil $A - sE$ to $P(A - sE)$. However, this fixed P cannot approximate $(A - sE)^{-1}$ for all s in general. The introduction of P may only be helpful over certain frequency ranges. To emphasize this difference with the traditional fixed case, we use the term dynamic system (DS) preconditioner in the following to describe the matrix P. Left transformations are utilized so that the DS preconditioned system is

$$\begin{cases} PE\dot{x}(t) = PAx(t) + Pbu(t) \\ y(t) = c^T x(t) + du(t). \end{cases} \tag{2.15}$$

It is stressed that (2.1) and (2.15) both describe the same system. The generalized eigenvalues of (PA, PE) and (A, E) are identical. Starting with this new description in (2.15), but utilizing the mapping seen in going from (2.1) to (2.14), it is consistent to define a new reduced-order model

$$\begin{cases} W^T PEV\dot{\hat{x}}(t) = W^T PAV\hat{x}(t) + W^T Pbu(t) \\ \hat{y}(t) = c^T V\hat{x}(t) + du(t), \end{cases} \tag{2.16}$$

where the matrix $V \in \mathbb{R}^{N\times M}$ now satisfies

$$\text{colsp}\,\{V(s)\} = \mathcal{K}_M(P(A - sE), Pb)$$

and the matrix $W \in \mathbb{R}^{N\times M}$ satisfies

$$\text{colsp}\,\{W(s)\} = \mathcal{K}_M((A - sE)^T P^T, c).$$

Although (2.1) and (2.15) describe the same original system, (2.16) describes a reduced-order system that is different from the one previously constructed according to (2.14). Both the presence of P in (2.16) and the modifications of the projection matrices lead to a new reduced-order model. In particular, the definition of V has changed from that in Section 2.3.1.

Model reduction of DS preconditioned dynamic systems is generally derived in the above fashion in the existing literature. The matrices V and W are explicitly computed and applied to the transformed version of the original dynamic system (2.15). Yet it is possible to obtain the new reduced-order model (2.16) directly from the original description in (2.1). If one defines Z to be $P^T W$ rather than (2.14), then the new reduced-order model in (2.16) can be written in the desired form of (2.7). By associating new projection subspaces with V and Z, new reduced-order models are possible. To obtain the reduced-order model in (2.16), choose V and Z according to

$$\begin{aligned} \text{colsp}\,\{V(s)\} &= \text{span}\left\{Pb, P(A - sE)Pb, \ldots, \{P(A - sE)\}^{M-1}Pb\right\} \\ \text{colsp}\,\{Z(s)\} &= \text{span}\left\{P^T c, P^T(A - sE)^T P^T c, \ldots, \{P^T(A - sE)^T\}^{M-1}P^T c\right\} \end{aligned} \tag{2.17}$$

and apply these matrices as in (2.7). Thus, DS preconditioning can be concisely defined in our context as the introduction of a matrix P, which takes us from the old V and Z choices in (2.14) to those of (2.17). This definition of DS preconditioning is equivalent to the above use of a left-transformation P followed by the application of V and W in (2.16). We favor the point of view in (2.17) over (2.16), however. Focusing on V, Z rather than V, W avoids the need for left transformations on the original description. Additional benefits of the V, Z point of view become apparent in Section 3.2.

Clearly, the introduction of P into V and Z provides additional freedoms for specifying the contents of the constraint and solution subspaces. Section 2.4.3 shows how a P can be chosen to acquire a reduced-order model that matches moments about a single interpolation point. The choice for the DS preconditioner determines where the frequency response of the original and reduced-order systems agrees.

A noteworthy class of DS preconditioners in frequency-dependent problems is the so-called exact DS preconditioners. An exact DS preconditioner is the exact inverse of the matrix pencil, $P = (A - \sigma E)^{-1}$, at a fixed scalar σ. We see in Chapter 3 that exact DS preconditioners are required if rational interpolation is to be achieved. An important property of exact DS preconditioning is presented as Lemma 2.1. The proof for this and all other lemmas in this dissertation may be found in Appendix A.

Lemma 2.1 *For any value of s and σ,*

$$(A - \sigma E)^{-1}(A - sE) = I + (\sigma - s)(A - \sigma E)^{-1}E. \tag{2.18}$$

Applying $P = (A - \sigma E)^{-1}$ to the pencil $(A - sE)$ leads to the simpler transformed pencil which consists of a scaled matrix PE shifted by the identity matrix. However, scalings and shifts by the identity matrix are not important in Krylov subspaces.

Lemma 2.2 (Krylov subspace shift-invariance) *For any matrix G, vector g and nonzero scalar η,*

$$\mathcal{K}_j(\eta G + I, g) = \mathcal{K}_j(G, g). \tag{2.19}$$

Combining Lemmas 2.1 and 2.2 leads to the equivalences

$$\mathcal{K}((A - \sigma E)^{-1}(A - sE), (A - \sigma E)^{-1}b) = \mathcal{K}((A - \sigma E)^{-1}E, (A - \sigma E)^{-1}b) \tag{2.20}$$

and

$$\mathcal{K}((A-\sigma E)^{-T}(A-sE)^T,(A-\sigma E)^{-T}c)=\mathcal{K}((A-\sigma E)^{-T}E^T,(A-\sigma E)^{-T}c). \quad (2.21)$$

Thus, when an exact DS preconditioner is utilized, the preconditioned projection subspaces in (2.17) are equivalent to the frequency-independent subspaces on the right sides of (2.20) and (2.21). Exact DS preconditioning causes the V and Z of (2.17) to be invariant with respect to s. This fact is important, because frequency-dependent $V(s)$ and $Z(s)$ do not generally lead to LTI reduced-order models. If exact preconditioning is not utilized, the only option is to fix the s in V and Z to be some value ζ. Yet specifying s to be ζ in a frequency-dependent $V(s)$ and $Z(s)$ tends to favor an accurate solution at ζ over other frequencies. This issue is discussed further in Section 8.1.

DS preconditioning, especially exact DS preconditioning, can significantly improve the accuracy of the reduced-order model. However, there are two significant limitations to preconditioning in a frequency-dependent problem. First, the introduction of DS preconditioners can significantly complicate the computation of V and Z. Rather than multiplying by the sparse pencil $(A-sE)$ at each step, one must work with $P(A-sE)$. If P is an exact DS preconditioner, the inverse of $(A-\sigma E)$ appears in the generation of V and Z. Solving large-scale systems of linear equations to implicitly enact this inverse may be costly. Second, one may wonder how to choose good DS preconditioners. The Petrov-Galerkin constraints insure that the reduced-order model converges in, at most, N steps, but one must be able to specify DS preconditioners that achieve significantly faster results. Poorly chosen DS preconditioners are hardly better than no DS preconditioners at all. Both the cost and choice of DS preconditioners are addressed in subsequent chapters.

2.4 Existing Approaches

The number of papers proposing, exploring or utilizing Krylov-based projection for model reduction is approaching one hundred. In this section, a hopefully complete history of these and related works is presented. Subsections 2.4.1 through 2.4.3 present typical examples of the methods utilized in these existing efforts. However, it is certainly not claimed that these subsections precisely capture every one of the many variations present in the literature.

2.4.1 History

The methods forming the foundation for this work are relatively old. The history of Padé approximation, for example, spans more than one hundred years [38]. The algorithm of Lanczos, an important Krylov-based iteration, is nearing its fiftieth anniversary [39]. Yet, as evident by this dissertation and its many recent references, the understanding and application of these concepts is certainly not a closed topic.

A large number of the moment-matching methods, particularly the early ones, form a reduced-order model from an explicit knowledge of the desired moments of the original system (see for example Section 2.4.2 and [40]). Explicit methods such as [41] were utilized to construct Padé approximants in the area of control in the early 1970s. Extensions of these techniques to multiple interpolation points followed [42, 43, 44]. Of more recent interest, circa 1990, is a class of explicit moment-matching methods known as asymptotic waveform evaluation (AWE) [45, 46]. Although the AWE methods themselves vary little in basic concept from the earlier control implementations, the AWE techniques are applied for interconnect model reduction in the area of circuits. The methods received attention for their ability to reduce RC interconnect models involving tens of thousands of variables. A multipoint version of AWE, complex frequency hopping (CFH), is available as well [46]. Unfortunately, all of these explicit moment-matching methods are known to exhibit numerical instabilities, particularly as the dimension of the reduced-order model M grows. The source of these difficulties was pointed out in [47] and in the independent work of [48]. Both efforts point out that moment-matching via the Lanczos method (and more generally (bi)orthogonalized Krylov-based projection) is a preferred numerical implementation.

The first significant mathematical connection between the Lanczos algorithm, a Krylov-based technique, and model reduction occurred in the early 1980s. It was shown that partial realizations could be generated through the Lanczos algorithm [32]. Adaptations of Krylov subspaces were proposed in 1987 to generate Padé approximations and shifted Padé approximations [49]. Beyond the mathematical connections, the Lanczos method was utilized for model reduction in many application areas. The first of these areas chronologically was apparently structural dynamics. Even prior to the knowledge of the moment-matching connections, the Lanczos method was utilized in structural dynamics for model reduction based on eigenvalue analysis [50, 51, 52]. Later work in the field utilized the Lanczos method for Padé approximation [53] including MIMO systems [54, 55]. The next wave of application work took place in the control literature [56, 57, 58]. A large amount of existing work was repeated, although new results did appear in the areas of error analysis [59] and stability retention [60]. Very recently, Lanczos-based model reduction has become a popular topic in the area of high-speed circuits. Existing Lanczos algorithms were applied to the standard [47, 48], MIMO [61] and symmetric problems [62]. New algorithms were proposed for stability retention [63, 64]. However, through all of these application areas, the approaches remained closely tied to the classical Lanczos algorithm. These approaches did not emphasize or exploit the fundamental structure in projection techniques for rational interpolation.

2.4.2 Explicit moment-matching

Explicit moment-matching is a straightforward approach for constructing Padé approximations. It is typically a two-step process. First, $2M$ selected moments μ_j of the original system are explicitly computed. These moments are frequently the leading coefficients of a power series expansion about $s = 0$ or $s = \infty$ (see the last column of Table 2.1 for assorted moment definitions). In the second step, the reduced-order frequency response

$$\hat{\mathbf{h}}(s) = \frac{\hat{\phi}_{M-1}s^{M-1} + \ldots + \hat{\phi}_1 s + \hat{\phi}_0}{s^M + \hat{\psi}_{M-1}s^{M-1} + \ldots + \hat{\psi}_1 s + \hat{\psi}_0}$$

is forced to correspond to the selected moments. That is, the numerator parameters $\hat{\phi}$ and denominator parameters $\hat{\psi}$ are chosen so that the moments of the reduced-order system $\hat{\mu}_j$ equal those of the original system μ_j for $j = 1, 2, \ldots, 2M$. This parameter selection requires the solution of a linear systems of equations involving Hankel matrices. In the partial realization problem, for example, one solves the equation

$$\begin{bmatrix} \mu_{-1} & \mu_{-2} & \iddots & \mu_{-M} \\ \mu_{-2} & \iddots & \iddots & \mu_{-M-1} \\ \iddots & \iddots & \iddots & \iddots \\ \mu_{-M} & \mu_{-M-1} & \iddots & \mu_{-2M+1} \end{bmatrix} \begin{bmatrix} \hat{\psi}_0 \\ \hat{\psi}_1 \\ \vdots \\ \hat{\psi}_{M-1} \end{bmatrix} = \begin{bmatrix} \mu_{-M-1} \\ \mu_{-M-2} \\ \vdots \\ \mu_{-2M} \end{bmatrix} \tag{2.22}$$

to determine the $\hat{\psi}$ coefficients. Another equation is solved for the $\hat{\phi}$ coefficients. Similar Hankel equations arise in the cases of Padé and shifted Padé approximations. For the rational interpolation problem, equations involving the more general Loewner matrix must be solved [34]. In all cases, it is important to note that the system matrices and vectors only enter the modeling problem through the moments. Given the definition of the moments in Table 2.1, A and E only enter the problem through sparse matrix-vector multiplies and sparse linear system solves.

Unfortunately, numerical implementations of explicit moment-matching experience several difficulties. We consider only the most serious of these problems, ill-conditioned Hankel matrices, through an example. The reader is referred to [47] for a discussion of the shortcomings of explicit moment-matching.

Example 2.2 *Consider a simple hundredth-order dynamic system defined by*

$$A = 10^{-5} \begin{bmatrix} 10^5 & 0 & 0 & \cdots & 0 \\ 0 & 99 & 0 & \ddots & \vdots \\ 0 & 0 & 98 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & 1 \end{bmatrix}, \quad b = c = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix},$$

and E is an identity matrix. It is not difficult to see that given 16 *digits of finite precision, the computed $\mu_{-(j+1)} = c^T A^j b$ is equal to μ_{-j} for $j > 10$. For even moderate values of j, the change in consecutive moments is determined by only the largest eigenvalue of A,* 1, *in finite precision. The information corresponding to the other eigenvalues is rapidly* *lost in practice during the computation of higher order moments. The condition number of the Hankel matrix in (2.22) is on the order of* 10^{18} *when* M *is only five.*

With repetitive multiplications by a fixed matrix, it no longer becomes possible in finite precision to introduce additional new information into the reduced-order model. This loss of information due to repetitive multiplications manifests itself through ill-conditioned Hankel matrices in the explicit moment-matching equations. Regardless of the number of moments matched, the computed model never converges to the actual. Although a partial realization example was presented, the same difficulties occur for the various Padé schemes. In practice, bounds must be placed, i.e., $j < 10$, on the number of moments computed about a given expansion point if explicit moment-matching is utilized.

2.4.3 Lanczos-based moment-matching

Due to its relative numerical elegance and reliability, the nonsymmetric Lanczos algorithm has a become a popular choice for moment-matching, model-reduction methods. The nonsymmetric Lanczos method is due to Cornelius Lanczos and was originally proposed as a method for solving linear systems of equations and eigenvalue problems [39, 65]. Because we focus on nonsymmetric matrices throughout the following, the nonsymmetric designation of the Lanczos method (versus the symmetric Lanczos method) is dropped, but assumed.

The algorithm of Lanczos computes rectangular matrices V and $W \in \mathbb{R}^{N \times M}$ that restrict a specified matrix G to tridiagonal form,

$$S = W^T G V = \begin{bmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \gamma_2 & \alpha_2 & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_M \\ 0 & \cdots & 0 & \gamma_M & \alpha_M \end{bmatrix},$$

and that satisfy

$$\text{colsp}\,\{V\} \in \mathcal{K}_M(G, \hat{v}_1) \quad \text{and} \quad \text{colsp}\,\{W\} \in \mathcal{K}_M(G^T, \hat{w}_1). \tag{2.23}$$

The vectors $\hat{v}_1$ and $\hat{w}_1$ are user-specified starting vectors which lie in the direction of the first columns of V and W. Alternatively and equivalently, the Lanczos method can be viewed as an approach for constructing biorthogonal V and W, i.e., $W^TV = I$, that satisfy the same Krylov subspace conditions (2.23). The columns of V and W satisfying these constraints can be iteratively computed via the three-term recursions

$$\begin{aligned} \gamma_{m+1} v_{m+1} &= G v_m - \alpha_m v_m - \beta_m v_{m-1} \\ \beta_{m+1} w_{m+1} &= G^T w_m - \alpha_m w_m - \gamma_m w_{m-1}. \end{aligned} \tag{2.24}$$

Choosing the α, β and γ parameters in (2.24), so that $W^TV = I$, leads to a tridiagonal $S = W^TGV$ and vice versa. An implementation of (2.24) with the appropriate parameter selections is the nonsymmetric Lanczos algorithm in Algorithm 2.1. The interested reader is referred to [66, 67] for a recent and detailed study of the nonsymmetric Lanczos method.

Algorithm 2.1 Nonsymmetric Lanczos

Initialize: $\hat{v}_1$ and $\hat{w}_1$.

For $m = 1$ to M,

(S2.1.1) $v_m = \hat{v}_m / \gamma_m$ where $\gamma_m = \sqrt{|\hat{w}_m^T \hat{v}_m|}$;

(S2.1.2) $w_m = \hat{w}_m / \beta_m$ where $\beta_m = \text{sign}(\hat{w}_m^T \hat{v}_m)\gamma_m$;

(S2.1.3) $\alpha_m = w_m^T A v_m$;

(S2.1.4) $\hat{v}_{m+1} = G v_m - \alpha_m v_m - \beta_m v_{m-1}$;

(S2.1.5) $\hat{w}_{m+1} = G^T w_m - \alpha_m w_m - \gamma_m w_{m-1}$;

end

Actual implementations of the Lanczos method may encounter numerical difficulties including a loss of biorthogonality and so-called serious breakdowns [66, 67, 68, 69]. However, these breakdowns are less drastic and/or rarer than the breakdowns occurring in explicit moment-matching. Additionally, remedies are possible [68, 69] and are discussed in Section 3.3.3.

As noted in Section 2.4.1, the Lanczos method can be utilized to realize Padé approximants or partial realizations. A proof of these statements follows from [49], as well as in Section 3.1. Table 2.2 summarizes the appropriate Lanczos input choices for G, $\hat{v}_1$ and $\hat{w}_1$ in order to achieve various reduced-order models. The constructed V and W of the Lanczos method lead to the reduced-order model in (2.16). The differences in the reduced-order models of Table 2.2 are due entirely to the choice of exact DS preconditioners in V and W. Partial realizations utilize $P = E^{-1}$, Padé approximations involve $P = A^{-1}$, and the shifted Padé approximants utilize $P = (A - \sigma E)^{-1}$. It is important to observe that the reduced-order models in Table 2.2 are restricted to moment matching about a single interpolation point. The Lanczos method cannot generate a rational interpolant.

Table 2.2: Modeling Choices in the Lanczos Algorithm

Model Type	Lanczos Quantities			Model Quantities		
	G	$\hat{w}_1$	$\hat{v}_1$	Z	$\hat{A}$	$\hat{E}$
Partial Realize	$E^{-1}A$	c	$E^{-1}b$	$E^{-T}W$	S	I
Padé	$A^{-1}E$	c	$A^{-1}b$	$A^{-T}W$	I	S
Shifted Padé	$(A-\sigma E)^{-1}E$	c	$(A-\sigma E)^{-1}b$	$(A-\sigma E)^{-T}W$	$I+\sigma S$	S

Table 2.2 also defines Z in terms of the Lanczos matrix W, so that the reduced-order model can be generated via (2.7). In practice, the reduced-order system matrices $\hat{A}$ and $\hat{E}$ can be directly generated from the tridiagonal matrix S, according to the last two columns of Table 2.2. In the case of a partial realization, for example, $\hat{A}$ equals $Z^T AV = W^T E^{-1} AV = W^T GV = S$ and $\hat{E}$ equals $Z^T EV = W^T V = I$. Constructing the reduced-order model according to (2.7) is an explicit projection. In the Lanczos method, one implicitly constructs the reduced-order model based on the assumed biorthogonality of V and W. The validity of this assumption is studied further in Section 4.1.4.

The reader should note the involvement of the inverses of A, E or combinations of the two in the formation of the various moment-matching models. Hence, for a particular problem, a given model may be unrealizable due to singularities. Even when possible, the inclusion of inverses in the Lanczos method is not standard. Traditionally, the Lanczos algorithm assumes only matrix-vector products with easily accessible matrices. For the modeling approaches in Table 2.2, inverses must also be treated; they are actually implemented through solving systems of linear equations. These inverses are, in fact, examples of the exact DS preconditioner discussed in Section 2.3.2.

Finally, it should be mentioned that the Lanczos method avoids the difficulties encountered with explicit moment-matching. It does so by storing its modeling information in two biorthogonal matrices V and W rather than in moments. Recall that explicit moment-matching eventually fails because a certain direction (corresponding to the largest eigenvalue of G) quickly dominates the generated moments. On the other hand, the biorthogonality condition of Lanczos, $W^T V = I$, insures that new information is introduced into the projector at every step. Directions already present in $v_1, v_2, \ldots, v_m$ and $w_1, w_2, \ldots, w_m$ are theoretically kept orthogonal to w_{m+1} and v_{m+1} and, therefore, do not dominate the new information. Unfortunately, the Lanczos method does not maintain precise biorthogonality given limited numerical precision. The convergence of the reduced-order model in practice is slightly clouded. We frequently discuss the role of biorthogonality/orthogonality on model reduction in the following (see Sections 3.3.2 and 4.1.4).

CHAPTER 3

PROJECTION FRAMEWORK FOR RATIONAL INTERPOLATION

In this chapter, a set of sufficient conditions on the projection technique is derived to guarantee rational interpolation. These conditions are intuitively meaningful from several points of view. This projection approach is also far more general than unraveling a preselected iterative implementation. Although the latter is frequently emphasized in the existing literature, a projection approach provides a general framework for deriving and contrasting both new and existing implementations. The validity of this projection framework (and the corresponding algorithms) does depend on certain assumptions, which are covered in detail. The extension of the projection techniques to the cases of stable and/or MIMO systems also requires and receives special attention.

3.1 Rational Interpolation Theory

Under mild assumptions, dual conditions on the projection matrices V and Z are sufficient to produce a rational interpolant for a reduced-order model. A concise summary of these conditions and pertinent intuition can be found in Section 3.2. This section formally connects Krylov subspace projection and rational interpolation. Many of the statements and arguments made while working towards this goal are reminiscent of those in [49]. Our analysis of rational interpolation from a projector point of view is apparently the first to be completely general, however, in that we allow for an arbitrary number of interpolation points K, an arbitrary number of values and derivatives J_k about a given interpolation point, and a direct projection involving V and Z. No restrictions are placed on the (bi)orthogonality of V and Z.

Is is assumed throughout this section that the large-scale matrix pencil $(A - \sigma E)$ and the reduced-order pencil $(\hat{A} - \sigma\hat{E})$ are both nonsingular when σ is an interpolation point. The assumption on $(A - \sigma E)$ must hold if the moments corresponding to σ are to exist. The nonsingularity of $(\hat{A} - \sigma\hat{E})$ is considered in detail in Section 3.3. For now, it is sufficient to know that both assumptions typically hold.

The connection between rational interpolation and projection begins from a simple property of oblique projection. Lemma 3.1 shows under what conditions a projection can be applied to a direction without changing it. This behavior is generalized in subsequent lemmas to model reduction. One desires a V and Z which when applied to the original system do not modify the moments to be retained. The proof for Lemma 3.1 and all lemmas in this dissertation may be found in Appendix A.

Lemma 3.1 *If* $v \in \operatorname{colsp}\{V\}$, *then*

$$v = V(W^T v)$$

when V *and* $W \in \mathbb{R}^{N \times M}$ *are biorthogonal.*

The moments of the original system and the model are essentially two-sided, i.e., containing both an input and output direction. We proceed to present a fundamental lemma corresponding to each of these directions and then combine them in the desired result.

Lemma 3.2 *If* $\mathcal{K}_{J_b}\left((A - \sigma E)^{-1}E, (A - \sigma E)^{-1}b\right) \subseteq \operatorname{colsp}\{V\}$, *then*

$$\left\{(A - \sigma E)^{-1}E\right\}^{j-1}(A - \sigma E)^{-1}b = V\left\{(\hat{A} - \sigma\hat{E})^{-1}\hat{E}\right\}^{j-1}(\hat{A} - \sigma\hat{E})^{-1}\hat{b}$$

for $j = 1, 2, \ldots, J_b$.

Lemma 3.3 *If* $\mathcal{K}_{J_c}\left((A - \sigma E)^{-T}E^T, (A - \sigma E)^{-T}c\right) \subseteq \operatorname{colsp}\{Z\}$, *then*

$$c^T(A - \sigma E)^{-1}\left\{E(A - \sigma E)^{-1}\right\}^{j-1} = \hat{c}^T(\hat{A} - \sigma\hat{E})^{-1}\left\{\hat{E}(\hat{A} - \sigma\hat{E})^{-1}\right\}^{j-1}Z^T$$

for $j = 1, 2, \ldots, J_c$.

Theorem 3.1 *If*

$$\bigcup_{k=1}^{K} \mathcal{K}_{J_{b_k}} \left((A - \sigma^{(k)} E)^{-1} E, (A - \sigma^{(k)} E)^{-1} b \right) \subseteq \text{colsp}\{V\} \tag{3.1}$$

and

$$\bigcup_{k=1}^{K} \mathcal{K}_{J_{c_k}} \left((A - \sigma^{(k)} E)^{-T} E^T, (A - \sigma^{(k)} E)^{-T} c \right) \subseteq \text{colsp}\{Z\} \tag{3.2}$$

then the moments of (2.1) and (2.2) satisfy

$$-c^T \left\{ (A - \sigma^{(k)} E)^{-1} E \right\}^{j_k - 1} (A - \sigma^{(k)} E)^{-1} b = -\hat{c}^T \left\{ (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \hat{E} \right\}^{j_k - 1} (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \hat{b}$$

for $j_k = 1, 2, \ldots, J_{b_k} + J_{c_k}$ *and* $k = 1, 2, \ldots K$.

Proof: We develop the case where $2 \leq j_k \leq J_{b_k} + J_{c_k}$; the $j_k = 1$ case follows trivially from Lemma 3.2. For the $j_k \geq 2$ case, nonnegative integers j_{b_k} and j_{c_k} can always be found that satisfy $j_{b_k} \leq J_{b_k}$, $j_{c_k} \leq J_{c_k}$ and $j_{b_k} + j_{c_k} = j_k$. Given such a j_{b_k} and j_{c_k},

$$\begin{aligned} & c^T \left\{ (A - \sigma^{(k)} E)^{-1} E \right\}^{j_k - 1} (A - \sigma^{(k)} E)^{-1} b \\ = \quad & c^T (A - \sigma^{(k)} E)^{-1} \left\{ E (A - \sigma^{(k)} E)^{-1} \right\}^{j_{c_k} - 1} E \left\{ (A - \sigma^{(k)} E)^{-1} E \right\}^{j_{b_k} - 1} (A - \sigma^{(k)} E)^{-1} b. \end{aligned} \tag{3.3}$$

By Lemmas 3.2 and 3.3, expression (3.3) is equivalent to

$$\hat{c}^T (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \left\{ \hat{E} (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \right\}^{j_{c_k} - 1} Z^T E V \left\{ (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \hat{E} \right\}^{j_{b_k} - 1} (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \hat{b}$$

and may be further simplified to

$$\hat{c}^T \left\{ (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \hat{E} \right\}^{j_k - 1} (\hat{A} - \sigma^{(k)} \hat{E})^{-1} \hat{b}. \tag{3.4}$$

The quantity (3.4) is the corresponding moment of the reduced-order model. The above relations hold for any value of k between 1 and K, because V and Z contain Krylov subspaces corresponding to all the k in this range. ■

3.2 Interpretations of the Theory

To acquire a rational interpolant that matches the first $J_{b_k} + J_{c_k}$ moments about the interpolation points $\sigma^{(k)}$ for $k = 1, 2, \ldots K$, we propose selecting V and Z according to

(3.1) and (3.2). Assuming the nonsingularity of the pencils $(\hat{A} - \sigma^{(k)}\hat{E})$ at these points (see Section 3.3), Theorem 3.1 guarantees that the desired rational interpolant is acquired. We stress that any pair of projection bases satisfying (3.1) and (3.2) is sufficient to achieve the desired rational interpolant. Restrictions on V or Z, such as biorthogonality or orthogonality, are purely implementation specific choices. The question then is what is so special about the forms of (3.1) and (3.2)? In particular, do these forms suggest a family of projection-based approaches for computing the reduced-order model?

There are some clear connections to be made between the V and Z of (3.1) and (3.2) and the desired moments of the original system,

$$\mu_{j_k} = c^T(A - \sigma^{(k)}E)^{-1}E(A - \sigma^{(k)}E)^{-1}E \ldots (A - \sigma^{(k)}E)^{-1}E(A - \sigma^{(k)}E)^{-1}b.$$

An obvious correlation exists between the repetitive multiplication by $(A - \sigma^{(k)}E)^{-1}E$ in both the Krylov spaces and the moments. The sum, $J_{b_k} + J_{c_k}$, of the dimensions of the Krylov subspaces corresponding to $\sigma^{(k)}$ is exactly equal to the number of moments matched about $\sigma^{(k)}$ by the reduced-order model. This direct connection between a given pair of Krylov subspaces and the values/derivatives matched at a given frequency suggests a great deal of potential parallelism when forming the reduced-order model. Independent of the rest of V, only the basis for $\mathcal{K}_{J_{b_k}}\left((A - \sigma^{(k)}E)^{-1}E, (A - \sigma^{(k)}E)^{-1}b\right)$ is pertinent to the moments at $\sigma^{(k)}$. Matching moments about multiple points requires multiple Krylov subspaces. Constructing these multiple subspaces in parallel is addressed in Chapter 7.

In addition to moment-matching, it was demonstrated in Section 2.3.1 that the proposed model-reduction approach can be phrased in terms of solving the dual system of equations (1.2). Recall that the reduced-order frequency response can be written as $\hat{x}_c^T Z^T(sE - A)V\hat{x}_b$, where $V\hat{x}_b$ and $Z\hat{x}_c$ are approximate solutions to (1.2) in a Petrov-Galerkin sense. Rational interpolation is connected to the approximate solutions to (1.2) through the concept of varying DS preconditioners. Understanding what is meant by a varying DS preconditioner follows by using Lemmas 2.1 and 2.2 to obtain the equality

$$\begin{aligned} &\textstyle\bigcup_{k=1}^{K} \mathcal{K}_{J_{b_k}}\left((A - \sigma^{(k)}E)^{-1}E, (A - \sigma^{(k)}E)^{-1}b\right) \\ &= \textstyle\bigcup_{k=1}^{K} \mathcal{K}_{J_{b_k}}\left((A - \sigma^{(k)}E)^{-1}(A - sE), (A - \sigma^{(k)}E)^{-1}b\right). \end{aligned} \tag{3.5}$$

A dual result can be obtained for the subspace on the left of (3.2). The subspace on the right of (3.5) is simply the union of several preconditioned Krylov subspaces of the type in (2.17). In fact, the subspaces making up this union vary only in the value of the matrix $(A - \sigma^{(k)}E)^{-1}$, which we denote as an exact varying DS preconditioner, P_k. The choices of Z and V leading to rational interpolation simply combine several exactly preconditioned Krylov subspaces in (2.17). By introducing multiple DS preconditioners P_k into V and Z, one hopes to obtain accurate solutions to the dual equations (1.2) in each of the neighborhoods surrounding the interpolation points σ_k. The impact of various DS preconditioners (various choices for the $\sigma^{(k)}$) is considered in Chapter 6.

The use of multiple varying preconditioners has appeared in the literature [70, 71] for solving fixed systems of linear equations. Similarities can be seen between these algorithms and the model-reduction algorithms developed in Chapter 4. The model-reduction problem is frequency dependent; adapting multiple preconditioners to cover a range of frequencies is novel.

One may question the advantages of phrasing the rational interpolation problem in terms of shifted systems of linear equations and varying DS preconditioners. The strength of this new point of view becomes apparent as one relaxes the constraint that P_k equals $(A - \sigma^{(k)}E)^{-1}$, e.g., if an iterative method is used and not converged to working precision. Avoiding the need for exact inverses of the matrix pencil may significantly cut the costs involved in generating V and Z. Unfortunately, the concept of rational interpolation becomes clouded without exact inverses; inexact DS preconditioners cannot exactly match moments. On the other hand, the use of inexact DS preconditioners is accepted and is, in fact, standard for treating systems of linear equations. An array of inexact DS preconditioning techniques and iterative solvers can be considered for solving dual systems of linear equations. The use of approximations for $(A - \sigma^{(k)}E)^{-1}$ in model reduction is considered in Chapter 8.

We finish this section by noting that the matrices V and Z of Theorem 3.1 are not computed by existing Krylov-based model-reduction methods. Rather, the existing literature consistently employs the matrices V and W. Already seen in Sections 2.3.2

and 2.4.3, the columns of the W and Z matrices satisfy the simple relation

$$z_m = (A - \sigma_m E)^{-T} w_m \tag{3.6}$$

for some value of σ_m. Theoretically, we can approach the rational interpolation problem in terms of computing either appropriate V and W or V and Z. The V, W choice is more naturally suited to Lanczos-based implementations of Krylov projection. However, it has been decided that the V, Z choice yields a clearer presentation. The column spaces of V and Z are exact duals to each other. The reduced-order model follows trivially from V and Z, as in (2.7). Lastly, the appearance of varying DS preconditioners in V and Z is straightforward. Yet, the choice of a fixed left transformation and associated W as in (2.16) is no longer clear when the DS preconditioners vary.

3.3 Limits of the Theory

The connections developed in Section 3.1 between rational interpolation and projection depend on the assumed nonsingularity of $Z^T(A - \sigma^{(k)}E)V$ for $k = 1, 2, \ldots K$. Singularities must occur if any of the matrices V, Z or $(A - \sigma^{(k)}E)$ are singular. Even if these individual components are all nonsingular, $Z^T(A - \sigma^{(k)}E)V$ may still be ill-conditioned. This section explores various sources and remedies for singular $\hat{A} - \sigma^{(k)}\hat{E}$.

With regards to notation, recall that the matrix Z_m consists of the first m columns of $Z \in \mathbb{R}^{N \times M}$. Likewise, V_m is the first m columns of V. These first m columns form bases for some arbitrarily chosen m-dimensional subspaces contained in (3.2) and (3.1), respectively. One can define a reduced-order model of size $m < M$ by simply replacing V and Z in (2.7) with V_m and Z_m. Finally, note that V and Z are always V_M and Z_M.

3.3.1 A singular large-scale pencil

Matrix inverses of the form $(A - \sigma E)^{-1}$ are a dominant presence in the construction of the Krylov subspaces in colsp$\{V\}$ and colsp$\{Z\}$. However, if σ is a generalized eigenvalue of (A, E), this inverse does not exist, as $(A - \sigma E)$ is singular. An eigenvalue of the original system should not be chosen as an interpolation point. By itself, this restriction is an insignificant constraint; only a finite number of discrete points in a continuous plane need be avoided. Yet one may wonder about interpolation points in the neighborhood of these discrete singularity points. As the interpolation point nears an eigenvalue, the conditioning of $(A - \sigma E)$ worsens. Fortunately, and perhaps surprisingly, a poorly conditioned $(A - \sigma E)$ does not lead to catastrophic results when attempting to match information at $s = \sigma$. This conditioning issue was examined in [72] with respect to related concerns in the method of inverse iteration for eigenvalue problems. Quoting [72],

> The period when inverse iteration was first considered was notable for exaggerated fears concerning the instability of direct methods for solving linear systems and ill-conditioned systems were a source of particular anxiety. For this reason, it was widely held to be inadvisable to use a σ which was too accurate; it was thought that such an eigenvalue should be debased a little so that the resulting matrix $(A - \sigma I)$ would not be too ill-conditioned. Although it is now generally recognized that this is not necessary, and indeed is not to be recommended....

An ill-conditioned $(A - \sigma E)$ can be utilized for an accurate interpolation at σ due to the form of the error in $(A - \sigma E)^{-1}b$. In short (the reader is referred to [72] for a more rigorous analysis), this error is dominated by the eigenvector $\mathbf{x}_n$ as σ approaches the corresponding eigenvalue λ_n. However, this eigenvector is in the direction of the desired solution. Although the desired vector is $x = (A - \sigma E)^{-1}b$, the computed vector is $(1 + \epsilon)(A - \sigma E)^{-1}b$, where ϵ may be large. Such a scaling error is unimportant in a projection technique; one requires only an accurate subspace basis to acquire the desired reduced-order model. Scaling a basis vector by $(1 + \epsilon)$ does not perturb the resulting subspace.

Existing theory and practical experience show that accurate interpolation can occur at a frequency σ, where $(A - \sigma E)$ is ill-conditioned, but care must be taken when matching higher order moments at σ. Directions not corresponding to $\lambda_n \approx \sigma$ are significantly damped by $(A - \sigma E)^{-1}$ multiplication. Therefore, one can expect a loss of precision at frequencies away from σ, which is directly proportional to the number of digits shared by σ and λ_n. Acquiring information away from λ_n is better achieved by moving to a different interpolation point.

Ill-conditioned $(A - \sigma E)$ are, in fact, a rare concern in many applications for a more practical reason. If the original LTI system is dynamically stable, its eigenvalues are obviously restricted to the left-half plane. Yet we see in Chapter 6, that positive real or purely imaginary interpolation points are preferred for model reduction. Thus, unless unstable or lightly damped modes occur in the dynamic system, ill-conditioned $(A-\sigma E)$ cannot occur at all.

3.3.2 Rank-deficient projection matrices

The use of a V or Z that is not full rank also leads to a singular $Z^T(A - \sigma^{(k)}E)V$. In fact, this case leads to a singular $(\hat{A} - s\hat{E})$ for all s. There are multiple sources for a rank-deficient V or Z.

Theoretically, the loss of full rank in V or Z corresponds to the occurrence of an invariant subspace. For example, consider letting M go to N with a fixed interpolation point. Then, the column space of V is the Krylov subspace

$$\mathcal{K}_N((A - \sigma E)^{-1}E, (A - \sigma E)^{-1}b).$$

This subspace is, in fact, the controllability subspace of (2.1), because the eigenvectors of $(A - \sigma E)^{-1}E$ and (A, E) can be shown to be identical. If the original system is not completely controllable, the dimension of the controllability subspace is less than N and the column rank of V must be less than N. Of course, the fact that V is not full rank in this case is not really disturbing. Uncontrollability in the original system implies the existence of a completely accurate reduced-order model of order less than N, known as a minimal realization [31]. One has no need for a V of size N. In the context of the Lanczos method, a loss of rank in V or Z due to the computation of an invariant subspace is aptly termed a fortuitous breakdown.

More alarming than the theoretical possibility of an invariant subspace, though, is a rank loss in finite machine precision. It was seen in Section 2.4.2 that finite precision can lead to problems for naive implementations of moment-matching. Although the desired constraint and search subspaces may be theoretically acceptable, the computed V or Z may not possess full rank. Similar to Section 2.4.2, one should avoid forming v_{m+1}, for example, by simply multiplying v_m by $(A - \sigma^{(k)}E)^{-1}E$. Such explicitly constructed columns of V quickly become dependent in finite precision. In the fashion of inverse iteration, repeated multiplications by the matrix $(A-\sigma^{(k)}E)^{-1}E$ emphasize only a single eigendirection, e.g., Example 2.2.

Rather than explicitly constructing columns of V, it is common to construct v_{m+1} so as to force it to be orthogonal or biorthogonal against previous directions in V_m or Z_m. The columns of Z may be handled similarly. In this manner, the new columns of V and Z are kept independent from the old. One makes sure that new information is added as the size of V and Z grows. It is stressed that the placement of orthogonality or biorthogonality type constraints on V and Z is purely an implementational decision. Various biorthogonality/orthogonality possibilities are explored in Table 4.1 of Chapter 4. Yet these orthogonalization choices are in no way fundamental to model reduction via projection. Orthogonality/biorthogonality is one possible tool for avoiding rank-deficient V and Z in finite precision.

Whatever the choice for orthogonality or biorthogonality, a variety of numerical approaches exist for its enforcement. In order of increasing numerical robustness, these techniques include selective classical Gram-Schmidt, classical Gram-Schmidt, modified Gram-Schmidt, classical Gram-Schmidt with reorthogonalization, and Householder reflectors [3]. Some version of classic Gram-Schmidt is the most common choice. Given a vector $\tilde{g}_{m+1}$ to be orthogonalized against an orthogonal matrix G_m, classical Gram-Schmidt computes

$$g_{m+1} = \tilde{g}_{m+1} - \sum_{l=1}^{m} g_l (g_l^T \tilde{g}_{m+1}). \tag{3.7}$$

Each component in the summation of (3.7) orthogonalizes a column of G_m against g_{m+1}.

In some cases, certain past directions (columns of G_m) are known to be already orthogonal to $\tilde{g}_{m+1}$ due to the structure of the problem. In Lanczos-type methods for example, one knows a priori that $(g_l^T \tilde{g}_{m+1})$ are zero in theory for all $l < m-1$. Utilizing this knowledge and avoiding the computation of the zero terms leads to simple updates by short recursions. However, round-off error always perturbs these $(g_l^T \tilde{g}_{m+1})$ away from zero in practice. Short term recursions are less accurate than explicitly computing all m terms in the summation of classical Gram-Schmidt. Computing all m terms is classical Gram-Schmidt. Computing only certain terms in the summation of (3.7) while assuming the others to be zero is known as selective orthogonalization [73]. More robust than classic Gram-Schmidt is two passes of Gram-Schmidt, known as reorthogonalization [74]. In this approach, one computes g_{m+1} via (3.7), sets $\tilde{g}_{m+1} = g_{m+1}$, and computes the left-hand side of (3.7) once more.

3.3.3 A singular reduced-order pencil

Even if the matrices $(A - \sigma^{(k)}E)$, V and Z are nonsingular, $(\hat{A} - \sigma^{(k)}\hat{E})$ may still be ill-conditioned. This final subsection explores this situation in detail and, hence, assumes $(A - \sigma^{(k)}E)$, V and Z to be nonsingular. Our key insight into this situation is stated and proven as Theorem 3.2. If a singular $Z^T(A - \sigma^{(k)}E)V$ arises when forming an order M model that matches $J_{b_k} + J_{c_k}$ moments at $\sigma^{(k)}$, then there exists a reduced-order model of a size less than M that matches $J_{b_k} + J_{c_k} - 1$ moments at $\sigma^{(k)}$.

Theorem 3.2 *Consider a V and Z where $Z^T(A - \sigma^{(k)}E)V$ is singular and*

$$\mathcal{K}_{J_{b_k}}\left((A - \sigma^{(k)}E)^{-1}E, (A - \sigma^{(k)}E)^{-1}b\right) \subseteq \operatorname{colsp}\{V\} \tag{3.8}$$

$$\mathcal{K}_{J_{c_k}}\left((A - \sigma^{(k)}E)^{-T}E^T, (A - \sigma^{(k)}E)^{-T}c\right) \subseteq \operatorname{colsp}\{Z\}. \tag{3.9}$$

If $\tilde{V}_{M-1}, \tilde{Z}_{M-1} \in \mathbb{R}^{N\times(M-1)}$ are full-rank matrices satisfying the conditions

$$\operatorname{colsp}\left\{\tilde{V}_{M-1}\right\} \subset \operatorname{colsp}\{V\}, \tag{3.10}$$

$$\operatorname{colsp}\left\{\tilde{Z}_{m-1}\right\} \subset \operatorname{colsp}\{Z\}, \tag{3.11}$$

$$v_+ \equiv \left\{(A - \sigma^{(k)}E)^{-1}E\right\}^{J_{b_k}-1}(A - \sigma^{(k)}E)^{-1}b \notin \operatorname{colsp}\left\{\tilde{V}_{M-1}\right\}, \tag{3.12}$$

$$z_+ \equiv \left\{(A - \sigma^{(k)}E)^{-T}E^T\right\}^{J_{c_k}-1}(A - \sigma^{(k)}E)^{-T}c \notin \operatorname{colsp}\left\{\tilde{Z}_{M-1}\right\}, \tag{3.13}$$

and $\tilde{Z}_{m-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1}$ *is singular, then the* $(J_{b_k}+J_{c_k}-1)^{st}$ *moment of the dimension* $M-1$ *reduced-order model*

$$\begin{cases} (\tilde{Z}_{M-1}^T E\tilde{V}_{M-1})\dot{\tilde{x}} = (\tilde{Z}_{M-1}^T A\tilde{V}_{M-1})\tilde{x} + (\tilde{Z}_{M-1}^T b)u \\ \tilde{y} = (c^T\tilde{V}_{M-1})\tilde{x} + du \end{cases}$$

about $\sigma^{(k)}$ *equals the* $(J_{b_k}+J_{c_k}-1)^{st}$ *moment of the original system (2.1) about* $\sigma^{(k)}$.

Proof: Due to the conditions (3.8) through (3.13), the vectors

$$\begin{aligned} \tilde{v}_M &= v_+ - \tilde{V}_{M-1}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-1}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)v_+) \\ \tilde{z}_M &= z_+ - \tilde{Z}_{M-1}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-T}(\tilde{V}_{M-1}^T(A-\sigma^{(k)}E)^T z_+) \end{aligned} \tag{3.14}$$

form completed matrices

$$\tilde{V} = \begin{bmatrix} \tilde{V}_{M-1} & \tilde{v}_M \end{bmatrix} \quad \text{and} \quad \tilde{Z} = \begin{bmatrix} \tilde{Z}_{M-1} & \tilde{z}_M \end{bmatrix}$$

satisfying $\operatorname{colsp}\{V\} = \operatorname{colsp}\left\{\tilde{V}\right\}$ and $\operatorname{colsp}\{Z\} = \operatorname{colsp}\left\{\tilde{Z}\right\}$. Moreover, by the inclusion of classical Gram-Schmidt biorthogonalization in (3.14), one has the relation

$$\begin{bmatrix} \tilde{Z}_{M-1}^T \\ \tilde{z}_M^T \end{bmatrix}(A-\sigma^{(k)}E)\begin{bmatrix} \tilde{V}_{M-1} & \tilde{v}_M \end{bmatrix} = \begin{bmatrix} \tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1} & 0 \\ 0 & \alpha_M \end{bmatrix}. \tag{3.15}$$

The value of $\alpha_M = \tilde{z}_M^T(A-\sigma^{(k)}E)\tilde{v}_M$ in (3.15) must be zero due to the assumed singularity of $Z^T(A-\sigma^{(k)}E)V$ and nonsingularity of $\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_k$. Using (3.14), this α_M can be written as

$$z_+^T(A-\sigma^{(k)}E)v_+ - (z_+^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-1}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)v_+).$$

This expression further simplifies to

$$\begin{aligned} &c^T\{(A-\sigma^{(k)}E)^{-1}E\}^{(J_{b_k}+J_{c_k}-2)}(A-\sigma^{(k)}E)^{-1}b \\ &-c^T\{(A-\sigma^{(k)})^{-1}E\}^{(J_{c_k}-1)}\tilde{V}_{M-1}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-1}\tilde{Z}_{M-1}^T\{E(A-\sigma^{(k)}E)^{-1}\}^{(J_{b_k}-1)}b \end{aligned}$$

due to (3.12) and (3.13). However, because $\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1}$ is nonsingular, Lemmas 3.2 and 3.3 can be used to write this most recent expression for α_m as

$$\begin{aligned} &\mu_{(J_{b_k}+J_{c_k}-1)} \\ &\quad -c^T\tilde{V}_{M-1}\{(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-1}\tilde{Z}_{M-1}^T E\tilde{V}_{M-1}\}^{(J_{c_k}-1)}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-1}\cdot \\ &\qquad\qquad \{\tilde{Z}_{M-1}^T E\tilde{V}_{M-1}(\tilde{Z}_{M-1}^T(A-\sigma^{(k)}E)\tilde{V}_{M-1})^{-1}\}^{(J_{b_k}-1)}b. \end{aligned}$$

This quantity, α_m, is the difference between $(J_{b_k} + J_{c_k} - 1)^{st}$ moments of the original and reduced-order models. Yet α_m is known to be zero. ∎

Certainly there are many conditions involved in the statement of Theorem 3.2. Moreover, Theorem 3.2 is the typical, but not the most general, description of a singular $Z^T(A - \sigma^{(k)}E)V$ (by assuming $\tilde{Z}_{M-1}^T(A - \sigma^{(k)}E)\tilde{V}_{M-1}$ to be nonsingular, one guarantees that the rank of $Z^T(A - \sigma^{(k)}E)V$ is at least $M - 1$). Rather than treating all these conditions and cases in detail, we simply reiterate the main concept and provide an example. The reader should simply keep in mind that a singular $Z^T(A - \sigma^{(k)}E)V$ implies the existence of a system of order less than M that matches more than $2M - 2$ of the desired moments. A singular $Z^T(A - \sigma^{(k)}E)V$ implies the existence of a lesser approximation that is nearly as good or as good at matching the desired moments of the original system. Such behavior is illustrated by Example 3.1.

Example 3.1 *Consider a third-order system with*

$$A = \begin{bmatrix} -1 & 2 & -2 \\ 0 & -1 & 2 \\ 0 & 0 & -1 \end{bmatrix} \quad b = \begin{bmatrix} 8 \\ 2 \\ -1 \end{bmatrix} \quad c = \begin{bmatrix} -2 \\ 0 \\ 0 \end{bmatrix} \quad E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.2 & 0 & 1 \end{bmatrix}.$$

This system is stable, controllable and observable. In an attempt to obtain a second-order model that matches two moments at $\sigma^{(1)} = 0$ and two at $\sigma^{(2)} = 1$, one can follow Theorem 3.1 and choose

$$V = \begin{bmatrix} A^{-1}b & (A-E)^{-1}b \end{bmatrix} \quad \text{and} \quad Z = \begin{bmatrix} A^{-T}c & (A-E)^{-T}c \end{bmatrix}.$$

Unfortunately, the matrix $Z^T(A - \sigma^{(1)}E)V$ is

$$\begin{bmatrix} 20 & 10 \\ 10 & 5 \end{bmatrix},$$

which is singular. In agreement with Theorem 3.2, one can check that the first-order model described by $\{z_2^T A v_2,\ z_2^T b,\ v_2^T c,\ z_2^T E v_2\}$ matches not only the first and second moments of the original system about $\sigma^{(2)}$ (as expected), but also the first moment of the *original system about $\sigma^{(1)}$ (not expected). Note that this first-order model does not match the second moment of the original system at $\sigma^{(1)}$.*

The presence of singular $Z^T(A - \sigma^{(k)}E)V$ and the fortuitous matching of extra moments can be tied to other phenomenon. For example, a nonminimal, reduced-order model of size M implies the presence of a less than M^{th} order approximation that matches all of the moments of the M^{th} order model. Hence, a nonminimal, reduced-order model has a singular $Z^T(A - \sigma^{(k)}E)V$ for one or more k. This case is not much of a concern, because the generation of nonminimal, reduced-order models (beyond a minimal one) is a wasted effort. Perhaps a more common scenario for a singular $Z^T(A - \sigma^{(k)}E)V$ is the lack of any M^{th} order system that meets the specified $2M$ moment constraints. A rational interpolant of size M cannot always be found that meets the specified $2M$ constraints; a singular $Z^T(A-\sigma^{(k)}E)V$ is consistent with this situation. For example, no approximation with $M = 1$ exists that matches the moments $\mu_0 = 0$ and $\mu_1 = 1$ at some σ, because a first-order system has no zeros. The $M = 0$ approximation, $\hat{\mathbf{h}}(s) = 0$ for all s, does match the first moment $\mu_0 = 0$, though, agreeing with Theorem 3.2. Similarly, in Example 3.1, one can check that no reduced-order model of order less than or equal to two exists which matches the specified moments $\mu_0 = -20$, $\mu_1 = 24$ at $\sigma^{(1)} = 0$ and $\mu_0 = -10$, $\mu_1 = 5$ at $\sigma^{(2)} = 1$.

Fortunately, as Theorem 3.2 suggests, the environment surrounding a singular or nearly singular $(\hat{A} - \sigma^{(k)}\hat{E})$ is not a common one in practice. This is not to say that such difficulties never arise. Given that the cause of a singular $(\hat{A} - \sigma^{(k)}\hat{E})$ is now better understood, it is possible to characterize various approaches for working around the problem. Each of these remedies modifies the size of and/or the moment constraints on the reduced-order model in hopes that a valid rational interpolant can be realized. Each of these remedies also assumes that V, Z and $(A - \sigma^{(k)}E)$ are nonsingular.

Modified Model Dimension. One possible remedy is to simply increase the model size until all $(\hat{A} - \sigma^{(k)}\hat{E})$ are nonsingular, i.e., until the model is large enough to meet all required constraints. If a model of size M is lacking, then one simply skips over it and augments V and Z until a valid reduced-order model is found. This process is known as look-ahead in the Lanczos literature [68, 69]. Unfortunately, one rarely knows a priori the required increase in model size for a valid approximation to be found. Usually, this increase is only one or two iterations. This approach is appropriate for fast implementations; although the required implementation may be complicated and heuristic.

Revised Moment Constraints. A second remedy is to choose the interpolation points to avoid difficulties. Rather than fixing the number of moments matched J_k, a priori, one adaptively selects from among the interpolation points as the model reduction proceeds. For example, assume one has a valid model of size $M-1$. Then, the next interpolation point utilized (the next two moments matched) is selected to avoid singular $(\hat{A}-\sigma^{(k)}\hat{E})$. Theorem 3.2 and intuition tell us to avoid matching two new moments at an interpolation point $\sigma^{(k)}$ if the first of these new moments is already matched by the model of size $M-1$. Unfortunately, it is not known a priori how many interpolation points must be inspected before a valid one is found. Usually, this number is only one. This approach is suited for implementations that already match moments about multiple interpolation points. It is also consistent with choosing the interpolation points in an attempt to generate the most accurate reduced-order model possible. One avoids choosing an interpolation point to match data that are already included in the existing reduced-order model. Adaptively choosing interpolation points according to the modeling error is further considered in Chapter 6.

Reduced Moment Constraints. A final approach for avoiding a singular $(\hat{A}-\sigma^{(k)}\hat{E})$ is to construct a reduced-order model that matches fewer than $2M$ moment constraints. That is, if a reduced-order model of size M cannot meet the specified $2M$ moment constraints, drop one or more of the constraints. A well-known algorithm in this class is the Arnoldi method [75]. Arnoldi methods are guaranteed to avoid singular pencils by choosing $Z^T = V^T(A-\sigma E)^{-1}$. As long as V and $(A-\sigma E)$ are nonsingular (assumed in this section), $Z^T(A-\sigma E)V = V^TV$ must be nonsingular. However, with this choice for Z, the value(s) of the J_{c_k} in Theorem 3.1 are zero, so that an Arnoldi-type approach generally meets only M moment constraints. The advantages of the Arnoldi-type approach is its immunity to certain breakdowns. The disadvantage of the approach is its inability to match as many moments in the reduced-order model. It should be noted that approaches matching more than M, but less than $2M$ moments, still need to be studied.

3.4 Further Issues

In this dissertation, the focus is on the computation of a rational interpolant for a SISO system. Other system-related issues exist, however. In the following the issues of dynamic stability and MIMO systems are considered in the projection framework.

3.4.1 Stable models

Although the introduced projection approach matches moments of the original system, a different set of invariant quantities, the system's eigenvalues, are less regulated. It is known that the eigenvalues of $(\hat{A}, \hat{E})$, known as Ritz values, may lie in the right half of the complex plane even though the original system is stable. A discussion of instabilities in partial realizations is presented in [76]. Unstable reduced-order models of stable systems are frequently unacceptable, e.g., if the approximation is to be used for simulations.

To address these stability concerns, the author proposed techniques in [60, 77] to discard the unstable modes of the reduced-order model. Because finite Ritz values in the right-half plane cannot correspond to true eigenvalues of a stable system, eliminating the unstable Ritz values is a reasonable approach. Similar strategies also recently appeared in [48, 78]. Explicitly, one need find only order-M orthogonal left and right matrices to transform the reduced-order model into the form

$$\begin{bmatrix} \hat{E}_s & 0 \\ 0 & \hat{E}_u \end{bmatrix} \begin{bmatrix} \dot{\hat{x}}_s \\ \dot{\hat{x}}_u \end{bmatrix} = \begin{bmatrix} \hat{A}_s & 0 \\ 0 & \hat{A}_u \end{bmatrix} \begin{bmatrix} \hat{x}_s \\ \hat{x}_u \end{bmatrix} \tag{3.16}$$

and

$$\hat{y} = \begin{bmatrix} \hat{c}_s & \hat{c}_u \end{bmatrix}^T \begin{bmatrix} \hat{x}_s \\ \hat{x}_u \end{bmatrix} + du,$$

where the eigenvalues of $(\hat{A}_s, \hat{E}_s)$ are the stable ones of the initial reduced-order model. The leading subsystem is retained as the final reduced-order model. A straightforward approach for acquiring the form in (3.16) is discussed in [78]. For the Lanczos algorithm, an efficient implementation based on hyperbolic rotations can produce (3.16) with only $O(M)$ operations [60]. This approach is known as the implicitly restarted Lanczos algorithm. It implicitly edits the Lanczos iteration to rapidly acquire a new Lanczos iteration that generates the purely stable reduced-order subsystem.

Under the special conditions that A and E are normal matrices and V equals Z, stable reduced-order models can always be obtained. Under these conditions, a relation between the field of values of (A, E) and the convex hull of (A, E) exists that bounds the spectrum of $(\hat{A}, \hat{E})$ [7]. The guaranteed stability of partial realizations when A is normal, E is an identity matrix, and V equals Z is discussed in [77]. A projection involving the Cholesky factorization of E is utilized in [64] to acquire guaranteed stable Padé approximations when A and E are symmetric. Certainly such results are desirable, although they require normal matrices and limit the choices for V and Z.

3.4.2 Multiple-input multiple-output models

For a MIMO system, the rectangular matrices $B \in \mathbb{R}^{N \times L_b}$ and $C \in \mathbb{R}^{N \times L_c}$ take the place of the vectors b and c. Corresponding to these new input and output matrices, the corresponding reduced-order model becomes

$$\hat{A} = Z^T AV, \quad \hat{B} = Z^T B, \quad \hat{C} = V^T C, \quad \hat{D} = D, \quad \hat{E} = Z^T EV.$$

Similarly, the moments of the original system are now the matrices

$$C^T \{(A - \sigma^{(k)} E)^{-1} E\}^{j-1} (A - \sigma^{(k)} E)^{-1} B. \tag{3.17}$$

Trivially, the element in the (l_c, l_b) position of (3.17) is the j^{th} moment of the SISO system, whose input vector is the l_b^{th} column of B (denoted b_l) and whose output vector is the l_c^{th} column of C (denoted c_l). Thus, the following corollary to Theorem 3.1 follows readily for the MIMO case.

Corollary 3.1 *If*

$$\bigcup_{l_b=1}^{L_b}\left\{\bigcup_{k=1}^{K}\mathcal{K}_{J_{b_{k,l}}}\left((A-\sigma^{(k)}E)^{-1}E,(A-\sigma^{(k)}E)^{-1}b_l\right)\right\}\subseteq \mathrm{colsp}\left\{V\right\} \tag{3.18}$$

and

$$\bigcup_{l_c=1}^{L_c}\left\{\bigcup_{k=1}^{K}\mathcal{K}_{J_{c_{k,l}}}\left((A-\sigma^{(k)}E)^{-T}E^T,(A-\sigma^{(k)}E)^{-T}c_l\right)\right\}\subseteq \mathrm{colsp}\left\{Z\right\} \tag{3.19}$$

then the moments of original and reduced-order model satisfy

$$-c_l^T\left\{(A-\sigma^{(k)}E)^{-1}E\right\}^{j_{k,l}-1}(A-\sigma^{(k)}E)^{-1}b_l=-\hat{c}_l^T\left\{(\hat{A}-\sigma^{(k)}\hat{E})^{-1}\hat{E}\right\}^{j_{k,l}-1}(\hat{A}-\sigma^{(k)}\hat{E})^{-1}\hat{b}_l$$

for $j_{k,l}=1,2,\ldots,J_{b_{k,l}}+J_{c_{k,l}}$, $k=1,2,\ldots K$, $l_b=1,2,\ldots,L_b$ *and* $l_c=1,2,\ldots,L_c$.

Two important points arise from Corollary 3.1. First, although the number of scalar elements in (3.17) grows as the product L_cL_b, the changes in the size of (3.18) versus (3.1) and (3.19) versus (3.2) are linear with respect to L_b and L_c. To understand this difference, consider the special, but not uncommon case where $L_b=1$ and $L_c\gg 1$. The number of subsystems in the overall problem is L_c. Because $L_b=1$, the MIMO version of V in (3.18) simplifies to (3.1). Thus, for any Z of size M, the reduced-order model formed with this V is guaranteed to match $J_{b_1}+\ldots+J_{b_K}=M$ moments of every one of the L_c subsystems. For the case where $L_b=1$, it is possible to match M moments in every one of the L_c subsystems with projection matrices of only size M. The model-reduction problem is two-sided; but at times it is more cost-effective to concentrate on only one of the two sides.

A second point of importance in the MIMO problem is that the same number of moments need not be matched for every subsystem. The standard Krylov-based approaches to the MIMO problem assume $K=1$ and fix $J_{b_l}=J_{c_l}=J$ for all l [54, 55, 61, 79]. Such an approach is known as a block method, because the individual vectors in B or C are treated identically during the construction of the projection matrices. Although a block approach is perhaps the most straightforward to implement, situations can and do frequently arise where the complexity among the subsystems varies significantly. Corollary 3.1 provides a tremendous amount of flexibility for treating MIMO systems. The projection matrices may be weighted to achieve greater accuracy in specific subsystems.

Corollary 3.1 is a natural extension for rational interpolation in the MIMO case, yet the user should avoid going to extremes. Allowing K, L_b, L_c to each become even moderately large automatically precludes a small model size M. Moreover, there may be a large amount of overlap between the various individual Krylov subspaces, particularly with respect to variations in the l indices. For example, even though the number of theoretically required subspaces may grow large with L_b, a few subspaces may come very close to covering the entire union. It is unclear at this time as to how one might locate the most globally appropriate individual subspaces. Experiments replacing the KL_b subspaces in (3.19) with K Krylov subspaces that are independent of l (e.g., replace b_l with a random starting vector, replace b_l with the summed vector $\sum_l b_l$, etc.) led to only limited success in practice. More research is needed in the case where both L_b and L_c are large to determine an appropriate reduced-order model with a practical size.

CHAPTER 4

PROJECTION METHODS FOR RATIONAL INTERPOLATION

Chapter 3 demonstrates that a rational interpolant can be acquired with straightforward conditions on the column spaces of V and Z. The rational Krylov algorithm developed in Section 4.1 provides a great deal of freedom in computing bases for these subspaces. Specifically, the type of orthogonality imposed on the columns of the projection matrices differentiates the algorithms proposed in this chapter. A significant amount of attention is therefore given to the impact of various orthogonalization schemes in both theory and practice. One particularly elegant version of the rational Krylov method, the rational Lanczos method, leads to a generalization of the Lanczos method for treating multiple interpolation points. The development of the rational Lanczos is significant as it can compute rational interpolants with short iterative recursions. The chapter concludes with an example demonstrating the behavior of various rational Krylov approaches.

4.1 The Rational Krylov Method

Theorem 3.1 provides simple conditions on the column spaces of V and Z for acquiring a reduced-order model through rational interpolation. Specific choices for implementing V and Z that meet these conditions remains. The goal is to compute V and Z in both an efficient and a numerically stable manner. In this section, a general method, denoted the rational Krylov method, is introduced for computing V and Z. An infinite number of appropriate V and Z can, in fact, be generated with this method by simply varying a few well-defined parameters. Once this broad method is available, specific implementations follow readily. Additionally, the presence of a broad method provides a clear framework for comparing each version's particular attributes.

The rational Krylov (RK) method is presented in Algorithm 4.1. Its name follows from the description in [80] of the subspaces in (3.1) and (3.2) as rational Krylov subspaces. Additionally and not by accident, the title is fitting from the perspective that the RK algorithm generates rational interpolants.

Algorithm 4.1 Rational Krylov (General Version)

Initialize: $q_1 = (\gamma_1^q)^{-1}b$ and $w_1 = (\beta_1^w)^{-1}c$;

For $m = 1$ to M,

(S4.1.1) Input: σ_m, the interpolation point for m^{th} iteration;

(S4.1.2) $\tilde{v}_m = (A - \sigma_m E)^{-1} q_{p_m+1}$ and $\tilde{z}_m = (A - \sigma_m E)^{-T} w_{p_m+1}$;

(S4.1.3) $\gamma_m^v v_m = \tilde{v}_m - V_{m-1}\bar{v}_m$ and $\beta_m^z z_m = \tilde{z}_m - Z_{m-1}\bar{z}_m$;

(S4.1.4) $\tilde{q}_{m+1} = Ev_m$ and $\tilde{w}_{m+1} = E^T z_m$;

(S4.1.5) $\gamma_{m+1}^q q_{m+1} = \tilde{q}_{m+1} - Q_m \bar{q}_{m+1}$ and $\beta_{m+1}^w w_{m+1} = \tilde{w}_{m+1} - W_m \bar{w}_{m+1}$;

end

Two simplifying assumptions are made in going from Theorem 3.1 to the RK algorithm. First, the column spaces of V and Z are constructed to equal ($=$) rather than contain ($\supseteq$) the union of Krylov subspaces on the left sides of (3.1) and (3.2). Yet this assumption does not prevent rational interpolation; it is the Krylov subspaces which contain the desired moment information. Second, it is assumed that the dimensions of the dual Krylov subspaces are consistent, i.e., $J_{b_k} = J_{c_k} = J_k$ for all k. These choices allow the matching of the maximum number of possible moments for a given model size M.

One should immediately notice the presence of q_m and w_m in addition to v_m and z_m in Algorithm 4.1. The constructed Q and W are related in direct fashions with V and Z via (S4.1.4). The choice between primarily working with Q and W versus V and Z can be mainly based on ease of notation and point of view. By initially incorporating all four matrices (rather than only V and Z) into the RK algorithm, more options become apparent.

An iteration of the RK algorithm consists of executing (S4.1.1) through (S4.1.5). A user-specified interpolation point, σ_m, is associated with each iteration. The value of σ_m must be one of the K possible interpolation points $\sigma^{(1)}$ through $\sigma^{(K)}$. Although the ordering of the interpolation points in the RK algorithm is arbitrary, the number of times σ_m is chosen to be some $\sigma^{(k)}$ during the M iterations determines the number of moments matched at $\sigma^{(k)}$ by the final reduced-order model. Mainly, it is shown below that the number of moments matched at $\sigma^{(k)}$ is twice the number of times that σ_m is chosen to be $\sigma^{(k)}$ in the M iterations.

Steps (S4.1.2) through (S4.1.5) of the RK algorithm generate the new columns of the projection matrices. Step (S4.1.2) introduces new information into the column spaces of V and Z, while (S4.1.4) introduces new information into the column spaces of Q and W. The actual bases used to represent these columns spaces are determined in (S4.1.3) and (S4.1.5). The updates in these two steps correspond to the classical Gram-Schmidt procedure described by (3.7). The choices for the vectors $\bar{q}_m$, $\bar{v}_m$, $\bar{w}_m$ and $\bar{z}_m$ in these updates determine what type of biorthogonality or orthogonality is produced among Q, V, W and Z. Furthermore, it is these vectors that primarily distinguish the specific implementations. Several important options (but certainly not all) are summarized in Table 4.1. The first, second and fourth cases in Table 4.1 are implemented in detail in Sections 4.1.1 through 4.1.3. The study of these specific cases further clarifies the breadth of the RK Algorithm 4.1.

An additional component to (bi)orthogonalization is the specification of the scaling parameters β^w, β^z, γ^q and γ^v. The last column of Table 4.1, titled β, γ restriction, lists the conditions that must be met in each case by the choice of the four γ^q, γ^v, β^w and β^z parameters. In the second row, for example, orthogonal V and Z are required, $V^TV = Z^TZ = I$. One might therefore choose $\beta_m^w = \gamma_m^q = 1$ and

$$\gamma_m^v = \|\tilde{v}_m - V_{m-1}\bar{v}_m\|_2 \quad \text{and} \quad \gamma_m^z = \|\tilde{z}_m - Z_{m-1}\bar{z}_m\|_2$$

Table 4.1: Orthogonalization Choices in the Rational Krylov Algorithm

Case	$\bar{q}_m$	$\bar{v}_m$	$\bar{w}_m$	$\bar{z}_m$	β, γ Restrictions
none	0	0	0	0	—
V, Z orthogonal	0	$V_{m-1}^T \tilde{v}_m$	0	$Z_{m-1}^T \tilde{z}_m$	$\|z_m\|_2 = \|v_m\|_2 = 1$
V, Z biorthogonal	0	$Z_{m-1}^T \tilde{v}_m$	0	$V_{m-1}^T \tilde{z}_m$	$z_m^T v_m = 1$
V, W biorthogonal	0	$W_{m-1}^T \tilde{v}_m$	$V_{m-1}^T \tilde{w}_m$	0	$w_m^T v_m = 1$

to insure that $v_m^T v_m = z_m^T z_m = 1$. In all cases, there are fewer constraints than β, γ parameters.

There is one other parameter of interest in the RK algorithm, mainly the "previous" index p_m. This scalar subscript appears in (S4.1.2). The value of p_m locates the most recent iteration prior to iteration m that employed the same interpolation point used in the m^{th} iteration. If σ_m was not used as an interpolation point in the first $m-1$ iterations, one sets $p_m = 0$. If $\sigma_m = \sigma_{m-1}$, then σ_m was last used in the $(m-1)^{st}$ iteration and the value of p_m is $m-1$. The new directions generated in (S4.1.2) of the current iteration are founded on the directions computed the last time that σ_m was used. Example 4.1 sheds some light on both the role of p_m and the overall RK algorithm.

Example 4.1 *Consider a 5^{th} order model with $\sigma_1 = \sigma^{(2)}$, $\sigma_2 = \sigma^{(1)}$, $\sigma_3 = \sigma^{(2)}$, $\sigma_4 = \sigma^{(2)}$ and $\sigma_5 = \sigma^{(1)}$. Thus, the $m = 1$ iteration is said to be associated with $\sigma^{(2)}$, the $m = 2$ iteration is associated with $\sigma^{(1)}$, etc. Based on the above definition of p_m, one has $p_1 = 0$, $p_2 = 0$, $p_3 = 1$, $p_4 = 3$ and $p_5 = 2$. The value of p_5, for example, follows from the fact that prior to $m = 5$, $\sigma^{(1)}$ was last utilized in the second iteration.*

Choosing $\beta_m^w = \beta_m^z = \gamma_m^q = \gamma_m^v = 1$ and $\bar{q}_m = \bar{v}_m = \bar{w}_m = \bar{z}_m = 0$ for all m, one can check that the V constructed in Algorithm 4.1 takes the form

$$V = \left[\begin{array}{ccccc} P_2 b & P_1 b & (P_2 E) P_2 b & (P_2 E)^2 P_2 b & (P_1 E) P_1 b \end{array} \right],$$

where $P_k = (A - \sigma^{(k)} E)^{-1}$. A dual result holds for Z. The use of p_m in (S4.1.2) leads directly to V and Z that correspond to the desired Krylov subspaces in an order *determined by the selection of the σ_m. New columns of V and Z that correspond to the Krylov subspaces involving $\sigma^{(k)}$ are added whenever σ_m is chosen to be $\sigma^{(k)}$.*

Example 4.1 demonstrates that the RK algorithm with special parameter choices leads to the desired column spaces for V and Z. Prior to leaving this general version of the RK algorithm, it is important to show that the computed V and Z lead to the desired rational interpolant in all cases. The following results prove this fact by demonstrating that the column spaces of V and Z fit the required form of Theorem 3.1. The key to this proof is Lemma 4.1, which is proven in Appendix A.

Lemma 4.1 *If $\sigma^{(k)}$ and $\sigma^{(k+1)}$ are two arbitrary, distinct interpolation points, then*

$$\begin{aligned}&(A-\sigma^{(k)}E)^{-1}E\{(A-\sigma^{(k+1)}E)^{-1}E\}^{j-1}(A-\sigma^{(k+1)}E)^{-1}b \in \\ &\qquad\left\{\operatorname{span}\{(A-\sigma^{(k)}E)^{-1}b\}\bigcup\mathcal{K}_j\left((A-\sigma^{(k+1)}E)^{-1}E,(A-\sigma^{(k+1)}E)^{-1}b\right)\right\}\end{aligned}\tag{4.1}$$

and

$$\begin{aligned}&(A-\sigma^{(k)}E)^{-T}E^T\{(A-\sigma^{(k+1)}E)^{-T}E^T\}^{j-1}(A-\sigma^{(k+1)}E)^{-T}c \in \\ &\qquad\left\{\operatorname{span}\{(A-\sigma^{(k)}E)^{-T}c\}\bigcup\mathcal{K}_j\left((A-\sigma^{(k+1)}E)^{-T}E^T,(A-\sigma^{(k+1)}E)^{-T}c\right)\right\}\end{aligned}\tag{4.2}$$

for any value of $j \geq 1$.

Theorem 4.1 *Assume that V and Z are the results of M steps of the general RK algorithm with nonzero scaling parameters γ_m^q, γ_m^v, β_m^w and β_m^z. For any $m \leq M$, the relations*

$$\operatorname{colsp}\{V_m\} = \bigcup_{k=1}^{K}\mathcal{K}_{j_{k,m}}\left((A-\sigma^{(k)}E)^{-1}E,(A-\sigma^{(k)}E)^{-1}b\right)\tag{4.3}$$

and

$$\operatorname{colsp}\{Z_m\} = \bigcup_{k=1}^{K}\mathcal{K}_{j_{k,m}}\left((A-\sigma^{(k)}E)^{-T}E^T,(A-\sigma^{(k)}E)^{-T}c\right)\tag{4.4}$$

hold, where $j_{k,m}$ equals the number of times σ_m was chosen to be $\sigma^{(k)}$ in the first m iterations.

Proof: We begin by inductively proving that

$$\text{colsp}\,\{Q_{m+1}\} \subseteq \text{colsp}\left\{\left[\begin{array}{c|c} EV_m & b \end{array}\right]\right\} \tag{4.5}$$

for any $m \leq M$. The $m = 1$ case is straightforward because $\tilde{q}_1 = \gamma_1^q q_1$ is initialized to b. Assume (4.5) holds for $m = M - 1$ as well. Due to steps (S4.1.4) and (S4.1.5) of the RK algorithm,

$$\gamma_{M+1}^q q_{M+1} = Ev_M - Q_M \bar{q}_{M+1}.$$

Combining this expression with the inductive assumption demonstrates that (4.5) holds for $m = M$ as well. Hence, (4.5) holds by induction.

We next show that for any $m \leq M$ the relation

$$v_m = \alpha_m \{(A - \sigma_m E)^{-1} E\}^{j_{\tilde{k},m}-1} (A - \sigma_m E)^{-1} b + V_{m-1} g_m \tag{4.6}$$

holds where α_m is nonzero and σ_m equals $\sigma^{(\tilde{k})}$ (the subscript $\tilde{k}$ identifies which of the K interpolation points was used in the m^{th} iteration). The equality in (4.3) follows trivially from (4.6).

Induction can be utilized to prove (4.6). For $m = 1$, the desired relation

$$\gamma_1^v v_1 = (A - \sigma_1 E)^{-1} b$$

follows directly from (S4.1.2) and (S4.1.3) of the RK algorithm. Assume (4.6) also holds for iterations 2 to $m - 1$. Based on (S4.1.2) and (S4.1.3), v_m can be written as

$$\begin{aligned} v_m &= (\gamma_m^v)^{-1} \{\tilde{v}_m - V_{m-1} \bar{v}_m\} \\ &= (\gamma_m^v)^{-1} \{(A - \sigma_m E)^{-1} q_{p_m+1} - V_{m-1} \bar{v}_m\}. \end{aligned} \tag{4.7}$$

If $j_{\tilde{k},m}$ is one, then p_m equals zero and q_{p_m+1} lies in the direction of b. For this case, (4.6) follows directly from (4.7). If $j_{\tilde{k},m}$ is greater than one, then (S4.1.4) and (S4.1.5) must be utilized to express (4.7) as

$$\begin{aligned} v_m &= (\gamma_m^v)^{-1} \left\{(A - \sigma_m E)^{-1} (\gamma_{p_m+1}^q)^{-1} \{\tilde{q}_{p_m+1} - Q_m \bar{q}_{m+1}\} - V_{m-1} \bar{v}_m\right\} \\ &= (\gamma_m^v)^{-1} \left\{(A - \sigma_m E)^{-1} (\gamma_{p_m+1}^q)^{-1} \{Ev_{p_m} - Q_m \bar{q}_{m+1}\} - V_{m-1} \bar{v}_m\right\}. \end{aligned}$$

Using (4.5), this last expression can be rewritten as

$$\begin{aligned} v_m &= (\gamma_m^v)^{-1}(\gamma_{p_m+1}^q)^{-1}(A-\sigma_m E)^{-1}Ev_{p_m} + (A-\sigma_m E)^{-1}\left[\begin{array}{cc} EV_{m-1} & b \end{array}\right]\tilde{g}_m + V_{m-1}\hat{g}_m \\ &= \alpha_{p_m}(\gamma_m^v)^{-1}(\gamma_{p_m+1}^q)^{-1}\{(A-\sigma_m E)^{-1}E\}^{j_{\tilde{k},m}}(A-\sigma_m E)^{-1}b \\ &\quad +(A-\sigma_m E)^{-1}\left[\begin{array}{cc} EV_{m-1} & b \end{array}\right]\bar{g}_m + V_{m-1}\hat{g}_m, \end{aligned} \tag{4.8}$$

where $\tilde{g}_m$, $\bar{g}_m$ and $\hat{g}_m$ are some vectors. The second equality follows from the inductive assumption and the fact that $j_{\tilde{k},p_m} = j_{\tilde{k},m} - 1$. From (4.8), Lemma 4.1 leads to the desired result (4.6). The value of α_m is simply a product of the inverses of γ^q and γ^v, terms which by assumption are nonzero,

$$\alpha_m = \alpha_{p_m}(\gamma_{p_m}^v)^{-1}(\gamma_{p_m+1}^q)^{-1}.$$

Expression (4.3) follows directly from (4.6). The portion of the proof corresponding to (4.4) is the dual to that presented above. ■

As a side note, it is an interesting fact that the proof of Theorem 4.1 is not strongly dependent on the specific value p_m in the subscript of (S4.1.3). The replacement of p_m in (S4.1.3) with the value $m-1$ arises in Section 4.1.4. Insights into alternatives to p_m are provided by Example 4.2.

Example 4.2 *Consider the construction of an orthogonal V_3 (see second row of Table 4.1) with the interpolation point-ordering $\sigma_1 = \sigma^{(1)}$, $\sigma_2 = \sigma^{(2)}$ and $\sigma_3 = \sigma^{(1)}$. The first two columns of V are therefore*

$$V_2 = \left[\begin{array}{c|c} \gamma_1(A-\sigma^{(1)}E)^{-1}b & \gamma_2(A-\sigma^{(2)}E)^{-1}b + \alpha_{2,1}v_1 \end{array}\right],$$

where the parameters γ_1, γ_2 and α_2 are chosen so that V_2 is orthogonal. Using the prescribed subscript $p_m = 1$ in step (S4.1.2) of the third RK iteration, one obtains a new direction

$$\tilde{v}_3 = (A-\sigma^{(1)}E)^{-1}q_{1+1} = (A-\sigma^{(1)}E)^{-1}Ev_1 = \gamma_1(A-\sigma^{(1)}E)^{-1}E(A-\sigma^{(1)}E)^{-1}b.$$

This new direction is always acceptable for augmenting the subspace.

If, on the other, the subscript p_m+1 is replaced with m in the subscript of (S4.1.2), the third direction is

$$\begin{aligned}
\tilde{v}_3 &= (A-\sigma^{(1)}E)^{-1}q_{2+1} = (A-\sigma^{(1)}E)^{-1}Ev_2 \\
&= \gamma_2(A-\sigma^{(1)}E)^{-1}E(A-\sigma^{(2)}E)^{-1}b \\
&\quad +\alpha_{2,1}(A-\sigma^{(1)}E)^{-1}E(A-\sigma^{(1)}E)^{-1}b. \qquad (4.9)
\end{aligned}$$

Due to Lemma 4.1, the $(A-\sigma^{(1)}E)^{-1}E(A-\sigma^{(2)}E)^{-1}b$ vector in (4.9) is a combination of the vectors $(A-\sigma^{(1)}E)^{-1}b$ and $(A-\sigma^{(2)}E)^{-1}b$, so that $\tilde{v}_3$ takes the form

$$\alpha_{2,1}(A-\sigma^{(1)}E)^{-1}E(A-\sigma^{(1)}E)^{-1}b+\alpha_{3,1}v_1+\alpha_{3,2}v_2. \qquad (4.10)$$

The vector in (4.10) is an acceptable third direction as long as $\alpha_{2,1}$ is nonzero. However, if $(A-\sigma^{(1)}E)^{-1}b$ and $(A-\sigma^{(2)}E)^{-1}b$ are (nearly) orthogonal, $\alpha_{2,1}$ is extremely small and $\tilde{v}_3$ fails to introduce a new direction.

Consistent with Example 4.2 and the proof of Theorem 4.1, we claim that the V and Z satisfy

$$\operatorname{colsp}\{V_m\} \subseteq \bigcup_{k=1}^{K} \mathcal{K}_{j_{k,m}}\left((A-\sigma^{(k)}E)^{-1}E,(A-\sigma^{(k)}E)^{-1}b\right) \qquad (4.11)$$

and

$$\operatorname{colsp}\{Z_m\} \subseteq \bigcup_{k=1}^{K} \mathcal{K}_{j_{k,m}}\left((A-\sigma^{(k)}E)^{-T}E^T,(A-\sigma^{(k)}E)^{-T}c\right), \qquad (4.12)$$

when a value is substituted for p_m in (S4.1.2) that is greater than p_m, but less than m. If equality does not hold in (4.11) or (4.12) (as it always must in Theorem 4.1 when p_m is utilized), then V and/or Z are rank-deficient. In this case, the substituted index for p_m is not appropriate. An approach to handling rank-deficient V or Z is presented at the end of Section 4.1.1.

4.1.1 A rational power Krylov algorithm

The various orthogonalization parameters and vectors in the RK algorithm provide a great deal of flexibility in computing valid V and Z. We consider in this section the simple case where $\gamma_m^q = \gamma_m^v = \beta_m^w = \beta_m^z = 1$ and $\bar{q}_m = \bar{v}_m = \bar{w}_m = \bar{z}_m = 0$. No orthogonalization is used in this approach; the Krylov subspaces are generated by directly multiplying previous vectors with $(A - \sigma^{(k)}E)^{-1}E$. With these choices, a simplified version of the RK algorithm, the rational power algorithm, Algorithm 4.2, results. We denote this approach as the RP (rational power) algorithm for consistency with the other abbreviations.

Algorithm 4.2 Rational Krylov (RP Version)

Initialize: $m = 0$

For $k = 1$ to K,

  For $j_k = 1$ to J_k,

    (S4.2.1) If $j_k = 1$,

      $\tilde{v}_m = (A - \sigma^{(k)}E)^{-1}b$ and $\tilde{z}_m = (A - \sigma^{(k)}E)^{-T}c$;

    else

      $\tilde{v}_m = (A - \sigma^{(k)}E)^{-1}Ev_{m-1}$ and $\tilde{z}_m = (A - \sigma^{(k)}E)^{-T}E^T z_{m-1}$;

    end

    (S4.2.2) $v_m = \tilde{v}_m / \|\tilde{v}_m\|_2$ and $z_m = \tilde{z}_m / \|\tilde{z}_m\|_2$.

    (S4.2.3) $m = m + 1$;

  end

end

Several choices were made in going from Algorithm 4.1 to the concrete implementation of Algorithm 4.2. First, the interpolation points used in the iterations, denoted σ_m, were utilized in a consecutive fashion. That is, the first J_1 iterations involved $\sigma^{(1)}$, the next J_2 iterations utilized $\sigma^{(2)}$, etc. Of course, the σ_m could theoretically be selected in any order. Different possibilities for ordering the interpolation points were illustrated in the algorithms developed in Sections 4.1.2 and 4.1.3. Due to the interpolation point-ordering in the RP algorithm, $p_m + 1$ takes on either the value 1 (a new interpolation point) or m (same interpolation point as the previous iteration). This fact leads to the two possible decision branches in (S4.2.1).

One should also note that the RP implementation lacks the q_m and w_m vectors present in the general RK algorithm. These vectors can be buried in the RP algorithm due to the choice $\bar{q}_m = \bar{w}_m = 0$. This simplification leads to $q_m = \tilde{q}_m$ and $w_m = \tilde{w}_m$, so that (S4.1.5) of the general version yields

$$q_m = Ev_{m-1} \quad \text{and} \quad w_m = E^T z_{m-1}.$$

These last expressions are substituted into (S4.1.2) of the general RK algorithm to remove the explicit presence of q_m and w_m from the RP implementation. By taking this step, one need only store two rather than four sequences of vectors in memory. In general, one or more of the q_m, v_m, w_m, or z_m equals the corresponding vectors $\tilde{q}_m$, $\tilde{v}_m$, $\tilde{w}_m$, and $\tilde{z}_m$ (see Table 4.1). When these equalities occur, it is possible that the corresponding vector sequence need not be stored in memory. We must compute at least two of the four projection matrices in general nonsymmetric problems to obtain a rational interpolant.

A relatively brief analysis of the RP implementation reveals that the constructed V takes the explicit form

$$V = \left[\begin{array}{ccc|ccc|cc} P_1 b & \ldots & (P_1 E)^{J_1 - 1} P_1 b & P_2 b & \ldots & (P_2 E)^{J_2 - 1} P_2 b & P_3 b & \ldots\ldots \end{array}\right],$$

where $P_k = (A - \sigma^{(k)} E)^{-1}$. The matrix Z takes a dual form. The RP algorithm realizes appropriate bases for colsp $\{V\}$ and colsp $\{Z\}$ in the most direct manner possible.

Of course, it has already been pointed out in Section 3.3.3 that a direct approach (straightforward and repeated multiplications by $(A - \sigma^{(k)} E)^{-1} E)$ may lead to numerical difficulties in practice. The lack of any sort of (bi)orthogonalization in the algorithm immediately causes concern. Indeed, unless the J_k are all extremely small, problems are sure to arise. Numerous repeated multiplications by a fixed matrix in finite precision no longer introduce new information into V and Z. Yet some hope arises for the RP algorithm because the values of $\sigma^{(k)}$ can be varied. If the interpolation point is frequently altered, the J_k are kept small. Changing to a new interpolation point where the problem has yet to converge guarantees that new information is added. That is, one places information into the projection matrices by repeated changes of $\sigma^{(k)}$, rather than repeated multiplications. This approach is similar in spirit to the complex frequency hopping improvements of AWE [46]. As in CFH, a successful RP implementation tends to require factors of $(A - sE)$ at numerous points. Unlike CFH, the projection technique of the RP algorithm generates a single reduced-order model that is a rational interpolant. A projection approach may also enable the use of techniques such as approximate solves or parallelism to reduce the matrix factorization costs (see Chapters 7 and 8).

Even with a frequent change of $\sigma^{(k)}$, unacceptable dependent columns can still appear in V or Z. Consider, for example, the redundancy in the V columns which results when two different interpolation points are very near to each other. Unfortunately, what constitutes "nearness" is problem dependent and difficult to characterize. In practice, one may try to adapt the point placement to the problem as the model reduction proceeds. In any situation, one cannot rule out the appearance of a few dependencies in the columns the RP-generated V and Z. As long as the values of J_k are kept small, one can expect this number of dependencies to be only a small fraction of M.

The dependent portion of V and Z can be discarded through a singular value decomposition. Assume that the ranks of V and Z are $M - \delta_V$ and $M - \delta_Z$, respectively, so that the rank of $(\hat{A} - s\hat{E})$ is at most the lesser of $(M - \delta_v, M - \delta_z)$ for any s. Then, by the singular value decomposition [3, 81], there exist orthogonal matrices $T_l = [T_{l_{\tilde{M}}} | T_{l_+}]$ and $T_r = [T_{r_{\tilde{M}}} | T_{r_+}]$ such that for a given matrix $(\hat{A} - \eta\hat{E})$,

$$\left[\begin{array}{c} T_{l_{\tilde{M}}}^T \\ \hline T_{l_+} \end{array}\right] (\hat{A} - \eta\hat{E}) \left[\begin{array}{c|c} T_{r_{\tilde{M}}} & T_{r_+} \end{array}\right] = \left[\begin{array}{c|c} \Sigma_\eta & 0 \\ \hline 0 & 0 \end{array}\right]. \tag{4.13}$$

The matrix Σ_η is nonsingular and square with a rank $\tilde{M}$ that is less than or equal to $\min(M - \delta_V, M - \delta_Z)$. By Theorem 4.2 below, the lower-order model consisting of $\tilde{A} = (T_{l_{\tilde{M}}} Z)^T A (V T_{r_{\tilde{M}}})$ and $\tilde{E} = (T_{l_{\tilde{M}}} Z)^T E (V T_{r_{\tilde{M}}})$ is an appropriate one for model reduction.

Theorem 4.2 . *If T_l and T_r are the left and right singular vectors of $(\hat{A} - \eta\hat{E})$ that lead to (4.13), then the matrices $V T_{r_{\tilde{M}}}$ and $Z T_{l_{\tilde{M}}}$ are full rank.*

Proof: In the singular value decomposition, the columns of T_{r_+} and T_{l_+} are known to form bases for the null spaces of $(\hat{A} - \eta\hat{E})$ and $(\hat{A} - \eta\hat{E})^T$, respectively [3]. Because $\hat{A}$ equals Z^TAV and $\hat{E}$ equals Z^TEV, the null spaces of V and Z must be respectively contained in the column spaces of T_{r_+} and T_{l_+} as well (if g is a vector such that $V^Tg = 0$, then clearly Z^TAVg and Z^TEVg are also zero). Yet T_{r_+} and T_{l_+} are orthogonal to $T_{r_{\tilde{M}}}$ and $T_{l_{\tilde{M}}}$, implying that $VT_{r_{\tilde{M}}}$ and $ZT_{l_{\tilde{M}}}$ are full rank. ■

The matrices $\tilde{V} = VT_{l_{\tilde{M}}}$ and $\tilde{Z} = ZT_{r_{\tilde{M}}}$ are full rank and thus, serve as suitable projection matrices. In fact, if $\tilde{M} = M - \delta_V$, then the column spaces of V and $\tilde{V}$ are equivalent. A dual result holds for $\tilde{Z}$. Otherwise, $\operatorname{colsp}\left\{\tilde{V}\right\} \subset \operatorname{colsp}\{V\}$, so that the column spaces of V and $\tilde{V}$ differ slightly (in a manner depending on the choice of η, which is typically one of the interpolation points). In this case, the reduced-order model involving $\tilde{A}$ and $\tilde{E}$ differs slightly from the desired rational interpolant. Additionally, the matrices V and Z are typically not exactly singular in practice. As a result, the lower-right corner of the rightmost matrix in (4.13) differs slightly from zero. For these reasons, projection with $\tilde{V}$ and $\tilde{Z}$ rather than V and Z yields better conditioned, yet slightly perturbed, reduced-order models.

In summary, the RP implementation is a simple version of the RK framework which avoids orthogonalization. Due to this simplicity, the issue of dependent projection directions must be addressed through the use of multiple interpolation points and possibly postprocessings. The RP approach certainly does not promote the level of understanding and elegance which follows from the inclusion of orthogonalization. Even so, the simplicity of the rational power Krylov approach allows for interesting possibilities in Chapter 7.

4.1.2 A dual rational Arnoldi algorithm

Arguably, the best approach for avoiding difficulties in the construction of V and Z is to insure that both of these quantities are orthogonal matrices. This technique is implemented as Algorithm 4.3. Algorithm 4.3 is denoted a dual rational Arnoldi (RA) version, because the steps taken to independently construct V and Z are each similar to the steps of the rational Arnoldi method of [15].

In Algorithm 4.3, the interpolation points are interspersed. The first iteration uses $\sigma^{(1)}$, the second iteration uses $\sigma^{(2)}$, ..., the K^{th} iteration involves $\sigma^{(K)}$, the $(K+1)^{st}$ iteration uses $\sigma^{(1)}$ again, etc. This alternating strategy fixes p_m to be $m-K$ and, therefore, determines the decision and indices in (S4.3.1). Again, it is stressed that ordering of the interpolation points is not theoretically a factor in the resulting reduced-order model. The approach in Algorithm 4.3 simply provides another example of an interpolation point selection strategy.

Algorithm 4.3 Rational Krylov (Dual RA Version)

Initialize: $m = 0$
For $j = 1$ to J,
  For $k = 1$ to K,
    (S4.3.1) If $j = 1$,
        $\tilde{v}_m = (A - \sigma^{(k)}E)^{-1}b$ and $\tilde{z}_m = (A - \sigma^{(k)}E)^{-T}c$;
      else
        $\tilde{v}_m = (A - \sigma^{(k)}E)^{-1}Ev_{m-K}$ and $\tilde{z}_m = (A - \sigma^{(k)}E)^{-T}E^T z_{m-K}$;
      end
    (S4.3.2) $\hat{v}_m = \tilde{v}_m - V_{m-1}V_{m-1}^T\tilde{v}_m$ and $\hat{z}_m = \tilde{z}_m - Z_{m-1}Z_{m-1}^T\tilde{z}_m$;
    (S4.3.3) $v_m = \hat{v}_m/\|\hat{v}_m\|$ and $z_m = \hat{z}_m/\|\hat{z}_m\|$;
    (S4.3.4) $m = m + 1$;
  end
end

As long as singular $(A - \sigma^{(k)}E)$ and invariant subspaces are avoided, the dual RA implementation is guaranteed to yield the desired V and Z. There is still no guarantee that the resulting $(\hat{A} - \sigma^{(k)}\hat{E})$ are all nonsingular (see Section 3.3.3). However, the ability to construct V and Z without the chance of breakdowns is an important point. As long as the orthogonality is maintained in a stable fashion, the dual RA implementation generates V and Z in a completely stable fashion. This is the first Krylov-projection-based implementation for rational interpolation (or Padé approximation) possessing this stability property.

Recall that the construction of an orthogonal basis is not a trivial process. Except for the Householder and reorthogonalized Gram-Schmidt cases, a gradual loss of orthogonality is observed in the computed versions of V and Z. Yet exact orthogonality is not a condition on Theorem 3.1, and even significant losses of orthogonality do not impair the quality of the reduced-order model. Some sort of postprocessing is required, however, if V and Z become severely ill-conditioned. The appearance of dependent columns with the classical Gram-Schmidt approach (implemented in Algorithm4.3), is rare but possible. In such a situation, the postprocessing approach discussed at the end of Section 4.1.1 is a possible remedy.

In summary, the dual RA method tends to be a better behaved algorithm than the other RK variants seen in this chapter. Unfortunately, the dual RA implementation is not as fast as a rational Krylov version that is derived in the next section. This fact is not particularly surprising; a trade-off between algorithm speed and robustness is common in numerical linear algebra. One should understand this spectrum of possibilities so that the proper implementation may be used with a given application. The general RK method of Section 4.1 provides great flexibility in balancing computational effort and stability.

4.1.3 An initial rational Lanczos algorithm

Rational Lanczos is a version of the RK algorithm that maintains a biorthogonal V and W. As such, it is simple to develop an initial version of the rational Lanczos (RL) method, Algorithm 4.4, from the general RK algorithm. Because the biorthogonalized V and W are not duals to each other, the steps of the RL implementation are not particularly symmetric with respect to each other.

Rational Lanczos is a potentially fast implementation of Krylov-based model reduction. It shares its name with a similar algorithm that was derived in [82]. The beauty

Algorithm 4.4 Rational Krylov (Initial RL Version)

Initialize: $q_1 = b$, $w_1 = (\beta_1^w)^{-1}c$ and $j_k = 0$ for $k = 1$ to K

For $m = 1$ to M,

(S4.4.1) Input: σ_m, interpolation point for m^{th} iteration;

(S4.4.2) $\tilde{v}_m = (A - \sigma_m E)^{-1} q_{p_m+1}$ and $z_m = (A - \sigma_m E)^{-T} w_{p_m+1}$;

(S4.4.3) $\gamma_m^v v_m = \tilde{v}_m - V_{m-1} W_{m-1}^T \tilde{v}_m$;

(S4.4.4) $q_{m+1} = E v_m$ and $\tilde{w}_{m+1} = E^T z_m$;

(S4.4.5) $\beta_{m+1}^w w_{m+1} = \tilde{w}_{m+1} - W_m V_m^T \tilde{w}_{m+1}$;

(S4.4.6) For $k = 1$ to K, If $\sigma_m = \sigma^{(k)}$, then $j_k = j_k + 1$; end; end

end

---

of the RL algorithm lies in the speed with which $\hat{A}$ and $\hat{E}$ can be computed. There are, in fact, two paths to generating a valid reduced-order model with rational Lanczos given Algorithm 4.4. The approach in [82] emphasizes the expression in (S4.4.3) and permutes the order of the W columns to produce a model in the form of (2.7), up to a similarity transformation. The approach developed in this section starts with the expression in (S4.4.5) and leads to a model that takes the exact form of (2.7). Although these two paths differ significantly, because V and W are not duals to each other, both paths are based on Algorithm 4.4 and lead to similar final results. We do not spend time reviewing the older version. The following implementation is superior in both its ease of development and the form of the resulting reduced-order model. Although both implementations of this section and [82] rapidly lead to rational interpolation, the newly proposed implementations follow clearly from the projection framework utilized throughout this dissertation.

We begin by considering the form of $\hat{E} = Z^T E V$ given that V and Z are generated according to Algorithm 4.4. The value of the m^{th} column of $\hat{E}^T$ appears during the computation of w_{m+1} in (S4.4.5) of the m^{th} iteration,

$$\begin{aligned} \beta^w_{m+1} w_{m+1} &= \tilde{w}_{m+1} - W_m V_m^T \tilde{w}_{m+1} \\ &= E^T z_m - W_m (V_m^T E^T z_m). \end{aligned} \tag{4.14}$$

The second equality (4.14) follows from the definition of $\tilde{w}_{m+1}$ in (S4.4.4) of the m^{th} iteration. The m^{th} column of $\hat{E}^T$ arises directly from the rightmost quantity in (4.14). Multiplying (4.14) on the left by V^T and recalling the biorthogonality of the matrices V and W yields the results

$$v_l^T E^T z_m = \beta^w_{m+1} \quad \text{for} \quad l = m+1$$

and

$$v_l^T E^T z_m = 0 \quad \text{for} \quad l > m+1.$$

These results hold for any value of m. Hence, $\hat{E}^T$ is an upper-Hessenberg matrix. In fact, even more can be said. For the $l \leq m$ case, algorithm step (S4.4.2) of the m^{th} iteration leads to the relation

$$v_l^T E^T z_m = v_l^T E^T (A - \sigma_m E)^{-T} w_{p_m+1}. \tag{4.15}$$

For specific values of $l \leq m$, the quantity in (4.15) is zero as well. The matrix $\hat{E} = Z^T E V$ generated by the RL algorithm is extremely sparse.

Theorem 4.3 *For the columns of V and W generated by rational Lanczos, $v_l^T E^T z_m = 0$ for $l < p_m$.*

Proof: To determine the values of $l \leq m$ such that the quantity in (4.15) is zero, note that $V_{p_m}^T w_{p_m+1}$ equals 0 by the biorthogonality of V and W. The question as to when (4.15) is zero can therefore be rephrased as when does $(A - \sigma_m E)^{-1} E v_l$ lie in the column space of V_{p_m}? The answer to this question follows from the fact that p_m is the index of the next to last iteration using the interpolation point σ_m. Due to this fact and Theorem 4.1,

$$\begin{aligned} \text{colsp}\,\{V_{p_m}\} \;\subseteq\;& \mathcal{K}_{j_{\tilde{k}}-1}((A - \sigma^{(\tilde{k})} E)^{-1} E, (A - \sigma^{(\tilde{k})} E)^{-1} b) \\ & \bigcup_{k \neq \tilde{k}} \mathcal{K}_{j_k}((A - \sigma^k E)^{-1} E, (A - \sigma^k E)^{-1} b), \end{aligned} \tag{4.16}$$

where $\tilde{k}$ is the index such that $\sigma^{(\tilde{k})} = \sigma_m$ and the values j_k, $k = 1$ to K, are those set by (S4.4.6) in the m^{th} iteration. Because by definition, $\sigma^{(\tilde{k})}$ was used in the $(p_m)^{th}$ iteration, it is known that the relation

$$\begin{aligned} \text{colsp}\left\{V_{p_m-1}\right\} \quad \subseteq \quad & \mathcal{K}_{j_{\tilde{k}}-2}((A-\sigma^{(\tilde{k})}E)^{-1}E, (A-\sigma^{(\tilde{k})}E)^{-1}b) \\ & \bigcup_{k \neq \tilde{k}} \mathcal{K}_{j_k}((A-\sigma^{(k)}E)^{-1}E, (A-\sigma^{(k)}E)^{-1}b) \end{aligned} \tag{4.17}$$

holds. By Lemma 4.1 and expressions (4.16) and (4.17), it is found that

$$\text{colsp}\left\{(A-\sigma_m E)^{-1}EV_{p_m-1}\right\} \subseteq \text{colsp}\left\{V_{p_m}\right\}.$$

Hence, we obtain

$$(A-\sigma_m E)^{-1}Ev_l \in \text{colsp}\left\{V_{p_m}\right\} \quad \text{for} \quad l < p_m$$

and (4.15) is zero for $l < p_m$.

If the above development, and (4.17) in particular, is to make sense, $j_{\tilde{k}}$ must be greater than one. When $j_{\tilde{k}} < 2$, $v_l^T E^T z_m$ is generally nonzero for all $l \leq m+1$. If $j_{\tilde{k}}$ is greater than one, $v_l^T E^T z_m$ is nonzero for $p_m \leq l \leq m+1$. ■

The presence of nonzero $v_l^T E^T z_m$ for $m+1 \geq l \geq p_m$ is a generalization of the three-term recursion present in the standard Lanczos method. In the Lanczos method, K equals 1 and p_m equals $m-1$, so that nonzero terms exist for values of l between $m+1$ and $m-1$. The behavior of the RL algorithm is demonstrated in Example 4.3.

Example 4.3 *Consider executing the RL algorithm for $M = 11$ iterations that use $\sigma^{(1)}$ in the first four iterations, $\sigma^{(2)}$ in iterations five through eight, and $\sigma^{(1)}$ again in the last three iterations. The values of the parameters σ_m, p_m, j_1 and j_2 as m varies are presented in Table 4.2. The structure of the matrix $\hat{E}^T$ is shown in Figure 4.1.*

The structure of $\hat{E}$ is particularly elegant when the interpolation points are utilized in an alternating fashion. In this case, p_m equals $m-K$ for $m > K$. Thus, $\hat{E^T}$ has a lower bandwidth of 1 and an upper bandwidth of K.

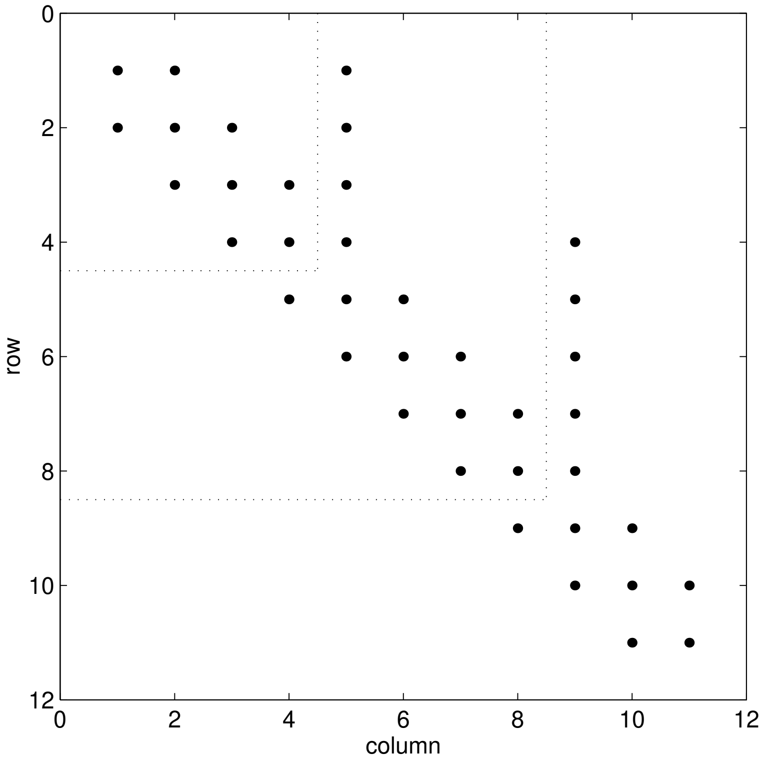

Figure 4.1: Sparsity Structure of $\hat{E}^T$ in Example 4.3

Table 4.2: Rational Lanczos Parameters in Example 4.3

m	1	2	3	4	5	6	7	8	9	10	11
σ_m	$\sigma^{(1)}$	$\sigma^{(1)}$	$\sigma^{(1)}$	$\sigma^{(1)}$	$\sigma^{(2)}$	$\sigma^{(2)}$	$\sigma^{(2)}$	$\sigma^{(2)}$	$\sigma^{(1)}$	$\sigma^{(1)}$	$\sigma^{(1)}$
p_m	0	1	2	3	0	5	6	7	4	9	10
j_1	1	2	3	4	4	4	4	4	5	6	7
j_2	0	0	0	0	1	2	3	4	4	4	4

Example 4.4 *Consider executing the RL algorithm again, as in Example 4.3, except with the interpolation points alternating in the order indicated by Table 4.3. Because seven iterations are still performed at $\sigma^{(1)}$ and four at $\sigma^{(2)}$, this new reduced-order model is equivalent (up to a similarity transform) with the results of Example 4.3. However, the use of alternating interpolation points leads to a different $\hat{E}^T$ in Figure 4.2. Note the banded structure of this $\hat{E}^T$.*

The structure of $\hat{A}$ is surprisingly sparse, as well. In fact, $\hat{A}$ follows in a very direct fashion from $\hat{E}$. To see this fact, note once more (S.4.4.2) and the biorthogonality of V and W. The m^{th} row of $\hat{A}$ is then

$$\begin{aligned} z_m^T AV &= z_m^T(A - \sigma_m E)V + \sigma_m z_m^T EV \\ &= w_{pm+1}^T V + \sigma_m z_m^T EV \\ &= i_{pm+1}^T + \sigma_m z_m^T EV. \end{aligned} \tag{4.18}$$

Thus, the m^{th} row of $\hat{A}$ is σ_m times the m^{th} row of $\hat{E}$, plus a standard unit vector. For a single interpolation point, $\hat{A}$ is $I + \sigma\hat{E}$. The matrix $\hat{A}$ follows in a trivial fashion from $\hat{E}$ in rational Lanczos. Yet we know that the $\hat{E}$ is simple to compute, as well.

Besides providing a pleasing structure to $\hat{E}$ and $\hat{A}$, the sparsity present in the RL algorithm can significantly reduce the memory and computational requirements of the algorithm. As seen in the Lanczos algorithm, banded matrices correspond to shortened recursions for the computation of the V and W vectors. For example, (S4.4.5) can be computed as

$$\beta_{m+1}^w w_{m+1} = \tilde{w}_{m+1} - \sum_{l=1}^{m} w_l(v_l^T E^T z_m). \tag{4.19}$$

Those components in the summation of (4.19) with $v_l^T E^T z_m = 0$ drop out trivially. For the special case, where the interpolation points are alternated in a regular fashion, e.g., Example 4.4, the scalars $v_l^T E^T z_m$ are zero for $l < m - K$ if $m > K$. Hence, (S4.4.5) becomes

$$\beta_{m+1}^w w_{m+1} = \tilde{w}_{m+1} - \sum_{l=\max(1,m-K)}^{m} w_l(v_l^T E^T z_m) \tag{4.20}$$

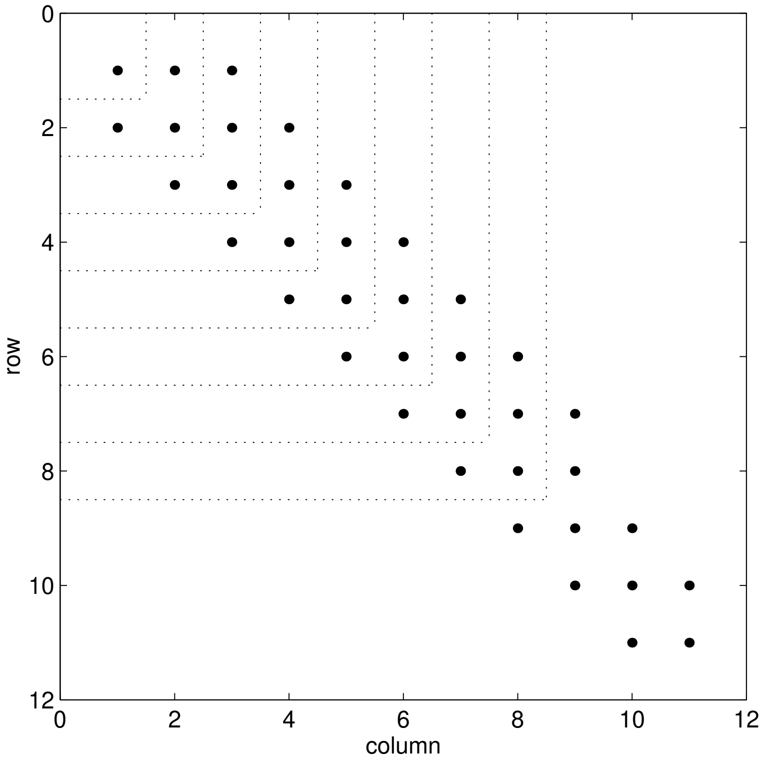

Figure 4.2: Sparsity Structure of $\hat{E}^T$ in Example 4.3

Table 4.3: Rational Lanczos Parameters in Example 4.4

m	1	2	3	4	5	6	7	8	9	10	11
σ_m	$\sigma^{(1)}$	$\sigma^{(2)}$	$\sigma^{(1)}$	$\sigma^{(2)}$	$\sigma^{(1)}$	$\sigma^{(2)}$	$\sigma^{(1)}$	$\sigma^{(2)}$	$\sigma^{(1)}$	$\sigma^{(1)}$	$\sigma^{(1)}$
p_m	0	0	1	2	3	4	5	6	7	9	10
j_1	1	1	2	2	3	3	4	4	5	6	7
j_2	0	1	1	2	2	3	3	4	4	4	4

for this special case. The computation of w_m is a length $K+2$ recursion which requires only a knowledge of the vectors v_{m-K} to v_m and w_{m-K} to w_m. Earlier vectors in V and W need not be stored in memory. If the interpolation points are regularly alternated and K is reasonably small, the RL algorithm is comparable from a memory and work standpoint with the standard Lanczos method. The use of regularly alternated interpolation points is important. Other interpolation point selection schemes, although not necessarily introducing a higher total number of nonzero elements into $\hat{A}$ and $\hat{E}$, lead to larger values of $m-p_m$ and thus, lengthened v_m and w_m recursions. A trade-off between the flexibility in interpolation point selection and the length of the iterative recursions exists.

4.1.4 A practical rational Lanczos algorithm

A version of the RL method that utilizes regularly alternated interpolation points is provided in Algorithm 4.5. There is a slight reordering of the steps in this version which makes for an easier implementation in practice. However, this reordering does not change the results. When discussing the RL algorithm in the remaining chapters, we are referring to Algorithm 4.5 unless otherwise noted.

The shortened recursion for w_m in (S4.5.3) follows from (4.20). The validity of truncating the v_m recursion in (S4.5.2) follows from the values of

$$\gamma_{m,l} = w_l^T(A-\sigma_m E)^{-1}Ev_{m-1}.$$

Dual to the proof of Theorem 4.3, expression (4.2) in Lemma 4.1 indicates that the vector $E^T(A-\sigma_m E)^{-T}w_l$ lies in the column space of W_{m-2} for $l < m-K-1$. Thus, the biorthogonality of V and W forces $\gamma_{m,l}$ to be zero for $l < m-K-1$.

The matrices $\hat{A}$ and $\hat{E}$ generated by Algorithm 4.5 follow readily from the above developed results. For example, comparing (4.20) and (S4.5.3) yields the descriptor

Algorithm 4.5 Rational Krylov (Banded RL Version)

Initialize: $\tilde{w}_1 = c$ and $\tilde{v}_1 = (A - \sigma^{(1)}E)^{-1}b$ and $m = 1$

For $j = 1$ to J,

  For $k = 1$ to K,

    (S4.5.1) if $m > 1$, $\tilde{v}_m = (A - \sigma^{(k)}E)^{-1}Ev_{m-1}$; end

    (S4.5.2) $\hat{v}_m = \tilde{v}_m - \sum_{l=\max(1,m-K-1)}^{m-1} v_l\gamma_{m,l}$ where $\gamma_{m,l} = w_l^T\tilde{v}_m$;

    (S4.5.3) $\hat{w}_m = \tilde{w}_m - \sum_{l=\max(1,m-K-1)}^{m-1} w_l\beta_{m,l}$ where $\beta_{m,l} = v_l^T\tilde{w}_m$;

    (S4.5.4) $v_m = \hat{v}_m/\gamma_{m,m}$ where $\gamma_{m,m} = \sqrt{|\hat{w}_m^T\hat{v}_m|}$;

    (S4.5.5) $w_m = \hat{w}_m/\beta_{m,m}$ where $\beta_{m,m} = \text{sign}(\hat{w}_m^T\hat{v}_m)\gamma_{m,m}$;

    (S4.5.6) $\tilde{w}_{m+1} = E^T(A - \sigma^{(k)}E)^{-T}w_m$;

    (S4.5.7) $m = m + 1$;

  end

end

---

matrix of the reduced-order model,

$$\hat{E} = \begin{bmatrix} \beta_{2,1} & \beta_{2,2} & & & & \\ \vdots & \beta_{3,2} & \ddots & & & \\ \beta_{K+2,1} & \vdots & \ddots & \beta_{JK-K,JK-K} & & \\ & \beta_{K+3,2} & & \beta_{JK-K+1,JK-K} & \ddots & \\ & & \ddots & \vdots & \ddots & \beta_{JK,JK} \\ & & & \beta_{JK+1,JK-K} & \cdots & \beta_{JK+1,JK} \end{bmatrix}. \quad (4.21)$$

Computing the last column of $\hat{E}$ requires the execution of (S4.5.3) in the $(M+1)^{st}$ iteration. This fact is consistent with the behavior of the standard Lanczos method. Given $\hat{E}$, the columns of $\hat{A}$ follow readily from (4.18).

Because c lies in the direction of w_1, the biorthogonality of V and W yields

$$\hat{c}^T = \begin{bmatrix} \beta_{1,1} & 0 & \dots & 0 \end{bmatrix}. \quad (4.22)$$

An expression for the reduced-order input vector results from the relation

$$\hat{b} = Z^Tb = Z^T(A - \sigma^{(1)}E)(A - \sigma^{(1)}E)^{-1}b = Z^T(A - \sigma^{(1)}E)v_1\gamma_{1,1}. \quad (4.23)$$

The vector (4.23) is simply the first column of $(\hat{A} - \sigma^{(1)}\hat{E})$ times a scalar. Due to (4.18) and (4.21), this vector is zero in all but its first K elements. Thus, $\hat{b}$ is sparse as well.

A rational interpolant can be achieved through short biorthogonalization recurrences. The efficiency of these short recurrences does not come without introducing some additional pitfalls, however. In the remainder of this section, we will explore some of the potential difficulties and the corresponding remedies that arise in a practical RL implementation. Analogies to each of these pitfalls and remedies exist in the standard Lanczos method.

The first harsh reality of the RL method is that the employed short recursions fail to keep V and W biorthogonal in finite precision. In fact, it is observed in practice that this biorthogonality is lost rapidly and to a significant extent. This issue might not be such a concern if the reduced-order model was explicitly constructed according to (2.7). However, the RL method constructs its reduced-order model based on a biorthogonality assumption. The validity of the banded structure of (4.21) is intertwined with the assumption of biorthogonality. The proposed remedy to similar problems in the literature is a simple one—do nothing. One ignores the loss of biorthogonality, continues to use short recursions, and blindly chooses the reduced-order model according to (4.21) through (4.23). Of course, the implemented reduced-order model is no longer an exact rational interpolant. Surprisingly enough though, the approximation generated with falsely assumed biorthogonality converges in practice, albeit slightly slower than the rational interpolant. For the symmetric $K = 1$ case in finite precision, the eventual convergence of the resulting approximation is assured [83]; but it is stressed that $M > N$ steps may be required to achieve the desired level of convergence in practice. Further comments on such behavior in the area of iterative solvers for linear systems of equations may be found in [84, 85]. The bottom line is that a loss of (bi)orthogonality tends to slow but not destroy the convergence of approximations that assume (bi)orthogonality and avoid the explicit use of (2.7). This behavior is still not perfectly understood, even for the nonsymmetric Lanczos method. It is certainly beyond the scope of this study to attempt to address the RL algorithm with further rigor.

Some intuition as to why the RL results converge at all follows from a local satisfaction of the Petrov-Galerkin constraint. It is shown in Section 5.2, without biorthogonality assumptions, that the output residual, $\mathbf{r}_c = c - (sE - A)^T Z\hat{\mathbf{x}}_c$, resulting from M rational Lanczos iterations is of the form

$$\mathbf{r}_c = \beta_{M+1,M+1} w_{M+1} \{ i_M^T (s\hat{E} - \hat{A})^{-T} \hat{c} \}.$$

Thus, if v_l is orthogonal to w_{M+1} for all l satisfying $L \leq l < M + 1$, which is guaranteed for $L = M - K$ by the implemented short recursions, then $v_l^T \mathbf{r}_c$ is zero for $L \leq l \leq M$. Although the RL algorithm may not enforce full biorthogonality in practice, it does guarantee that the current output residual is orthogonal to recent v directions. The Petrov-Galerkin constraints hold with respect to these recent v directions.

One further observation is important regarding the convergence of the RL algorithm in light of the biorthogonality loss. It has been repeatedly observed in practice that the eventual convergence of the RL results occurs only when p_m is replaced by the value $m-1$. This approach, introduced just prior to Section 4.1.1, contradicts the standard practice of choosing p_m to be the index of the next-to-last iteration employing σ_m. Replacing p_m with $m-1$ is actually implemented in the RL Algorithm 4.5. The superiority of the $m-1$ choice for methods that construct their approximations by assuming (bi)orthogonality was also observed in [81]. Related comments on the topic of the p_m choice may also be found in [81].

A second practical concern in the RL implementation is the so-called serious breakdown. A breakdown occurs in a Lanczos-type method when $\hat{w}_{m+1}^T \hat{v}_{m+1} = 0$. The assumptions of Theorem 4.1 are violated in this event, because either β_{m+1}^w or γ_{m+1}^v must take on the value zero. In this case, biorthogonality cannot be maintained and the algorithm cannot proceed. The theoretical background for such a breakdown is closely related to the theory of Section 3.3.3. In fact, the dot product $\hat{w}_{m+1}^T \hat{v}_{m+1}$ is proportional to the error in the $(2m+1)^{st}$ moment of the order m reduced-order model corresponding to V_m and W_m [82]. Recall though from Theorem 3.2, that a fortuitous matching of the $(2m+1)^{st}$ moment leads to a singularity in the order $m+1$ approximation. However, unlike the requirements of Section 3.3.3, which need only be concerned with singularities in the order M approximation, a singular $W_m^T V_m$ for any $m \leq M$ leads to a breakdown of rational Lanczos. Each subsequent step of rational Lanczos relies on the assumed biorthogonality of the existing projection matrices.

Solutions for sidestepping serious breakdowns include both the look-ahead approach and the interpolation point changes discussed in Section 3.3.3. These remedies are no longer simply postprocessing events; one may need to avoid breakdown at any iteration. The interested reader may find the intricate details of look-ahead Lanczos in [69]. Editing the interpolation order is not difficult to implement, but it does require the maintenance of longer biorthogonality recursions as $m - p_m$ grows. Either way, the avoidance of the breakdown results in fill-in outside of the band of elements in the rational Lanczos-generated $\hat{A}$ and $\hat{E}$. It is stressed again that the issue of serious breakdowns is only a concern for the RL version of the considered RK implementations.

One final pertinent issue in implementing rational Lanczos is the choice of the γ^v and β^w parameters. Care should be taken when scaling the vectors $\hat{v}_m$ and $\hat{w}_m$ to obtain the biorthogonal vectors v_m and w_m. The standard approach, the one taken in Algorithm 4.5, is to scale $\hat{v}_m$ and $\hat{w}_m$ such that $\beta_{m,m} = \pm\gamma_{m,m}$. A more stable approach occasionally seen in the Lanczos literature is to select

$$\gamma_{m,m} = \sqrt{\frac{|w_m^T v_m| \cdot \|v_m\|_2}{\|w_m\|_2}} \quad \text{and} \quad \beta_{m,m} = \operatorname{sign}(w_m^T v_m)\sqrt{\frac{|w_m^T v_m| \cdot \|w_m\|_2}{\|v_m\|_2}}$$

so that the norms of v_m and w_m are identical.

4.2 Comparisons

Given the variety of numerical approaches for implementing rational interpolation via projection, the obvious question becomes which method is the best? No simple answer exists. Problem dependent factors such as the eigenvalue spectrum, the sparsity pattern of the large-scale matrix pencil and the desired accuracy in the reduced-order model can favor different variations of the general RK approach. Handling these varying factors in a numerically efficient and robust manner requires flexibility.

The computational costs of the three methods of Sections 4.1.1, 4.1.2 and 4.1.4 follow from relatively simple analyses of Algorithms 4.2, 4.3 and 4.5. The results of these analyses on M iterations of each algorithm are summarized in Table 4.4. Column two indicates the number of floating point operations (flops) required to generate the two projection matrices V and W or Z. Column three lists the number of flops required to form the model once these projection matrices are formed. The data in both columns are only accurate to the order of the dominant terms. Table 4.2 utilizes the special notation $\mathcal{A}$, $\mathcal{E}$, $\mathcal{F}$ and $\mathcal{X}$ to represent the cost of specific matrix operations utilized in the RK method. The cost to acquire the triangular factors of $(A - \sigma^{(k)}E)$ is denoted by $\mathcal{F}$, $\mathcal{X}$ is the cost to solve a system of equations given these factors, $\mathcal{E}$ is the cost of multiplying a dense vector by E, and $\mathcal{A}$ is the cost of multiplying a dense vector by A. Determining the precise costs in floating point operations (flops) for these matrix operations requires a knowledge in practice of the sparsity patterns of A and E and of the specific techniques used to exploit the sparsity patterns. For example, most RC models of circuit interconnects involve an $\mathcal{F}$ on the order of $N^{1.4}$ and $\mathcal{X}$, $\mathcal{E}$, and $\mathcal{A}$ on the order of N.

Table 4.4: Computational Costs of RK Implementations

Method	Projection Matrix Generation	Model Generation
RP	$K\mathcal{F} + 2M(\mathcal{E} + \mathcal{X})$	$M(\mathcal{A} + \mathcal{E}) + 4NM^2$
Dual RA	$K\mathcal{F} + 2M(\mathcal{E} + \mathcal{X}) + 4NM^2$	$M(\mathcal{A} + \mathcal{E}) + 4NM^2$
RL	$K\mathcal{F} + 2M(\mathcal{E} + \mathcal{X}) + 4NMK$	$2M^2$

Each of the methods requires at least $K\mathcal{F} + 2M(\mathcal{E} + \mathcal{X})$ operations to form and use the factors of $(A - \sigma^{(k)}E)$. The remaining difference between the three algorithms is due to the amount of orthogonalization/biorthogonalization performed and the effort required to generate the reduced-order model. The RP algorithm requires no orthogonality recursions, the RL method requires length K recursions and the dual RA method requires length M recursions. If the fixed cost $K\mathcal{F} + 2M(\mathcal{E} + \mathcal{X})$ dominates, e.g., the matrix factorizations cost $\mathcal{F}$ is large due to a lack of easily exploitable sparsity, then these differences are negligible. If, on the other hand, the matrices are extremely sparse and M is large, the cost of the different implementations varies significantly. In this case, the RL algorithm is desirable, because its cost only grows linearly in Km. The RL algorithm employs short recursions that automatically lead to a reduced-order model. The dual RA method is far more expensive when the fixed costs are negligible, because both its cost to orthogonalize V and Z and its effort to compute the reduced-order model grow quadratically in m. The RP implementation requires linearly increasing work to compute V and Z, but quadratically increasing effort to form its reduced-order model. Because the cost of the RP method is on the order of the dual RA approach, and yet its robustness is less, the sequential RP implementation of Algorithm 4.2 is rarely recommended.

The memory requirements of the different variations are related to costs. A summary of the memory requirements for the three algorithms is in Table 4.5. The second column lists the memory needed in iteratively storing the columns of V, W and/or Z, while the third column lists any additional memory required to store the reduced-order model. Analogous to Table 4.4, the variable $\mathcal{F}$ now denotes the space to store a matrix factorization, while $\mathcal{A}$ and $\mathcal{E}$ denote the number of nonzero elements in A and E. Again, the overall memory requirements cannot be exactly specified, but depend on sparsity and the values of K, M and N. Assuming the memory required to store the factorizations is not dominant, the RL implementation requires a fixed amount of memory, while the memory requirements of the dual RA and RP implementation grow linearly with m.

Table 4.5: Memory Requirements of RK Implementations

Method	Projection Matrix Generation	Model Generation
RP	$K\mathcal{F} + \mathcal{E}$	$2M^2 + \mathcal{A} + 2MN$
Dual RA	$K\mathcal{F} + \mathcal{E} + 2MN$	$2M^2 + \mathcal{A}$
RL	$K\mathcal{F} + \mathcal{E} + 2KN$	$2M^2$

From a memory and cost standpoint, the RL method is preferred over the dual RA method. However, Section 3.3.2 suggests that numerical accuracy favors the methods in exactly the opposite order. The RL implementation gains its speed from less reliable short recursions. Although the rational Lanczos-generated model may eventually converge, this convergence may be delayed relative to the results of the dual rational Arnoldi implementation. This behavior is explored in Example 4.5.

Example 4.5 *To compare the behavior of the dual rational Arnoldi and the rational Lanczos implementations, we consider a generated problem intended to mimic the behavior of a packaging interconnect of a circuit. This generated problem consists of fifteen identical segments connected in series (see Figure 4.3 for the structure of a segment). Using Modified Nodal Analysis (MNA) [86], a set of equations of size $N = 47$ can be formulated to describe the interconnect. The frequency response of the interconnect between 10^8 and 10^{11} Hertz is shown in Figure 4.4. The input to this system is a voltage source placed at the left of the first segment; the output is the current through this source.*

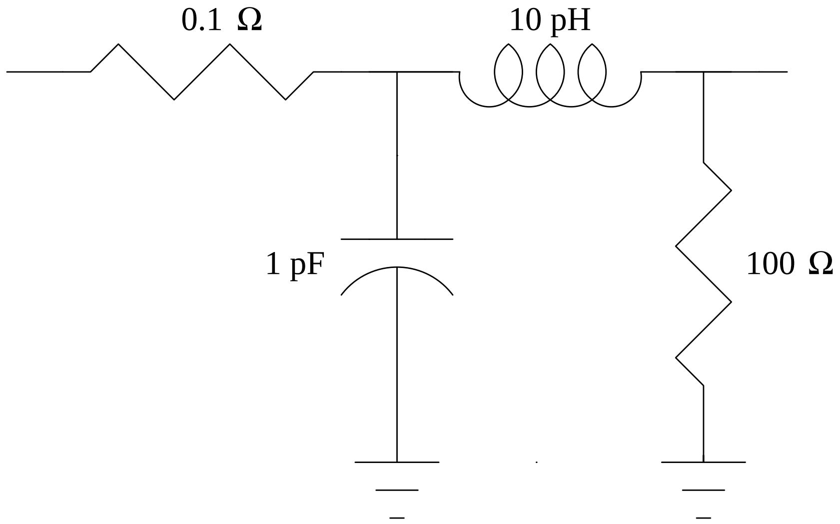

Figure 4.3: Interconnect Segment of Example 4.4

Forty iterations of both the dual RA and the rational Lanczos algorithms were executed in MATLAB; the utilized code is available in Appendix B. In both cases, these iterations alternated between the use of the interpolation points $\sigma^{(1)} = 2\pi 10^8$ and $\sigma^{(2)} = 2\pi 10^{11}$.

Figures 4.5 through 4.7 display the convergence, loss of biorthogonality/orthogonality, and computational costs for the two approaches as a function of the iteration m. *Figure 4.5 presents a computed estimate of the relative* $\mathcal{H}_2$ *error norm (weighted over* 10^8 *to* 10^{11} *Hz) between the original system and the respective reduced-order models. Although the identical selection of interpolation points in both cases suggests identical convergence in infinite precision, the convergence of the RL method (dotted line) is clearly slower. This behavior in finite precision agrees with practical observations and the limited theory in the literature (see the discussion in Section 4.1.4). The delay in convergence is correlated with a loss of biorthogonality in the RL case (see the dotted line in Figure 4.6 denoting* $\|I - W_m^T V_m\|_2$*). The orthogonality in the dual RA case is also eventually lost as well (see the dashed line denoting* $0.5(\|I - V_m^T V_m\|_2 + \|I - Z_m^T Z_m\|_2)$*). The dual RA implementation employs classical Gram-Schmidt orthogonalization. This approach is unstable, yet it is more robust than the shortened biorthogonalization recursion used in rational Lanczos. However, we stress that this difference in orthogonalization schemes does not itself determine the convergence difference. The difference arises from the fact that dual RA employs (2.7) to construct the reduced-order model, which does not depend on orthogonality, while the RL approach constructs a banded reduced-order pencil, which does depend on biorthogonality. This assumption of bandedness and the associated undesirable convergence delay in rational Lanczos is offset, though, by reduced computational effort. Figure 4.7 presents the cumulative number of floating point operations required by the two modeling implementations as a function of* m. *The work involved in the RL case grows linearly with* m *(dotted line), while the work in the dual RA case grows quadratically with* m *(dashed line). It is interesting to note that both of the approaches require about the same number of total operations to compute reduced-order models with errors in Figure 4.5 less than* 10^{-10}. *Although requiring more iterations, the RL approach need not involve more work. Such behavior tends to be problem dependent and contributes to the long-standing arguments over Lanczos-type versus Arnoldi-type approaches in the numerical linear algebra literature.*

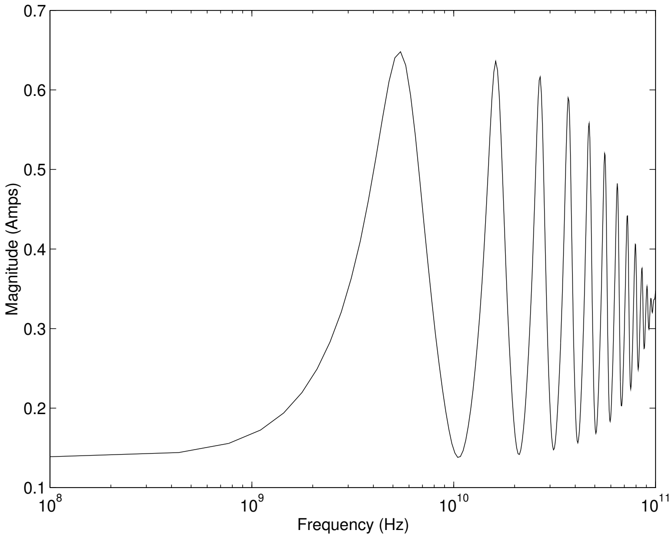

Figure 4.4: Frequency Response of Example 4.4

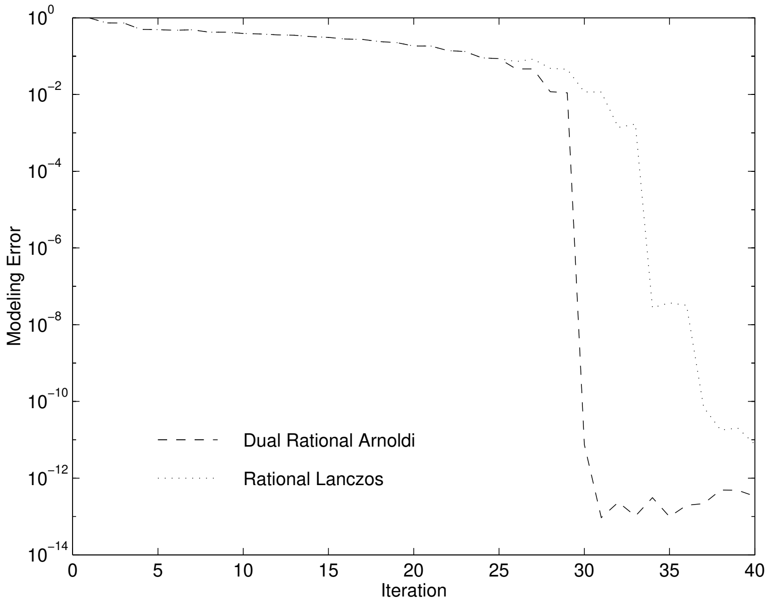

Figure 4.5: Convergence in Example 4.4

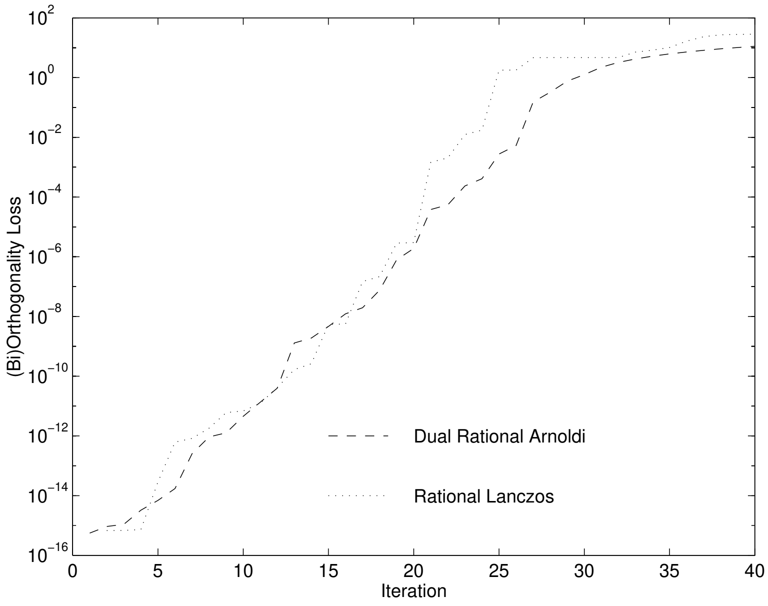

Figure 4.6: Loss of (Bi)Orthogonality in Example 4.4

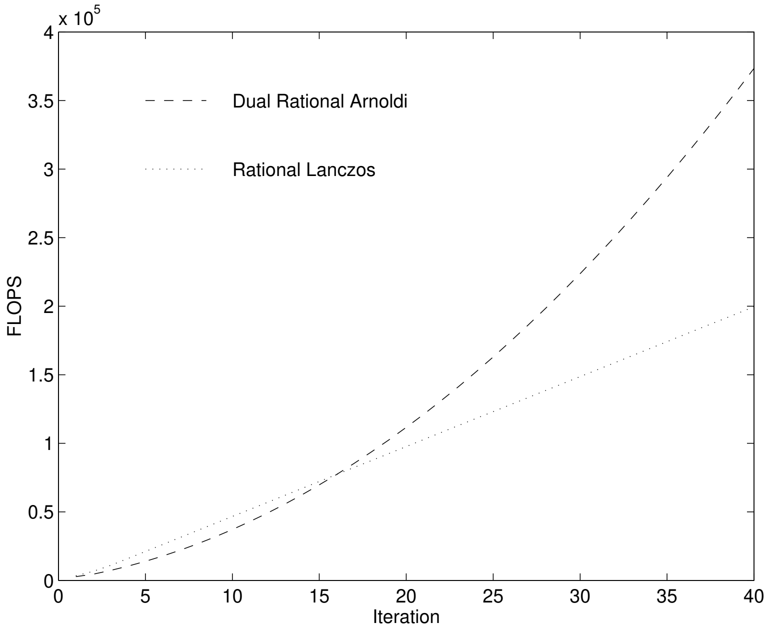

Figure 4.7: Computational Costs of Example 4.4

A few statements can be made to summarize the choice for the RK implementation. The RP algorithm is not appropriate for cases where J_k exceeds 10. The dual RA algorithm is most likely preferred when the complexity of factorizing $(A - \sigma^{(k)}E)$ dominates. Beyond these extremes, one must balance the convergence reliability of dual rational Arnoldi against the shortened recursions of rational Lanczos. Additionally, further research is merited to determine if any other desirable versions of the general RK algorithm exist.

CHAPTER 5

MODEL ERROR

A knowledge of the error between the original system and the computed rational interpolant is important for several reasons. It can be used to monitor the number of iterations required for convergence of the reduced-order model. In simulation, one needs to know that the response of the reduced-order model is sufficiently close to that of the original system. In control, one hopes to construct a controller from the reduced-order model that performs acceptably with the original system. In all applications, unnecessarily large models should be avoided due to computational cost. A measure of the error might also be feedback to adapt the modeling procedure itself. For example, one can attempt to select the interpolation points used in later steps to focus on the errors present after earlier iterations.

In this chapter, two approaches for estimating the error in the reduced-order model are developed. The merits of each are compared, particularly in the examples at the conclusion of this chapter. Unfortunately, none of these proposed techniques are guaranteed to be completely without inaccuracies. This reality follows naturally from the fact that a combined knowledge of both the reduced-order model and the modeling error implies a total knowledge of the original system. Because a complete analysis of the original system is to be avoided, one must resort to approximations to measure the gap between the original and reduced-order models.

5.1 Complementary Approximations

A simple approach for estimating the frequency-response error, $\epsilon(s) = \hat{\mathbf{h}}(s) - \mathbf{h}(s)$, between the original and reduced-order models is to compute the difference between two reduced-order models,

$$\hat{\epsilon}(s) = \hat{\mathbf{h}}(s) - \hat{\mathbf{h}}_{\perp}(s). \tag{5.1}$$

The transfer function of $\hat{\mathbf{h}}_{\perp}(s)$ corresponds to some second and completely different low-order approximation of the original system. Both of these approximations can be generated by any (and not necessarily the same) Krylov-based projection algorithm. Hence, (5.1) is a suitable and achievable error estimate for any of the previously discussed modeling techniques.

The two low-order approximations used in (5.1) should contrast in their approximations of the original system because this difference estimates the modeling error. That is, two points of view of the original system are sought, which are designed to be complementary. The use of drastically different viewpoints typically suggests that $\hat{\mathbf{h}}$(s) and $\hat{\mathbf{h}}_{\perp}(s)$ agree consistently only at those frequencies where both approximations are accurate. Where these two different viewpoints agree (where $\hat{\epsilon}(s)$ is small), one assumes that $\epsilon(s)$ is small. Where these two different viewpoints diverge, at least one of the two approximations must be inaccurate and $\epsilon(s)$ is assumed to be significant. Note that this last assumption errs on the conservative side, because the lack of a converged $\hat{\mathbf{h}}_{\perp}(s)$ at some frequency does not directly imply a large $\epsilon(s)$ at that frequency.

The generation of two distinct reduced-order models requires the construction of two different projection pairs of dimension M, the previously seen V, Z and the second pair $V_{\perp}, Z_{\perp}$ (note that the $\perp$ subscript denotes complimentary, but not necessarily orthogonal directions; colsp $\{V\}$ = colsp $\{V_{\perp}\}$ when $M = N$). The flexibility in forming these two pairs of projection matrices resides in the choice of interpolation points. Two different sets of interpolation points are sought, which lead to two distinct reduced-order models. In [87], the frequency responses of multiple Padé approximations, where each obviously utilizes a single, distinct interpolation point, are compared to estimate convergence. As a generalization for rational interpolation, we propose the use of two interlaced sets of interpolation points (interlaced moment-matching). An example of this point distribution is seen in Figure 5.1, where the black dots correspond to the first approximation and the white dots to the other. The use of interlaced interpolation points (versus distinct clusters of points) provides each of the two reduced-order models with a reasonable opportunity to converge across the entire frequency range. Recall that both viewpoints must converge before the estimated error diminishes. Keeping the individual interpolation points of the two models (i.e., $\sigma^{(k)}$ and $\sigma_{\perp}^{(k)}$) separated leads to approximations with the desired local, complementary viewpoints.

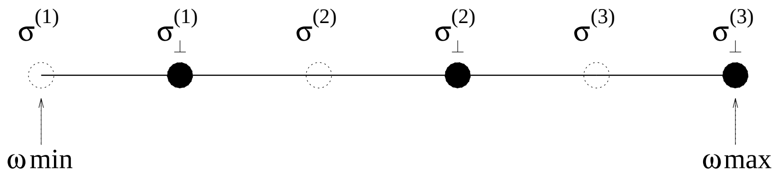

Figure 5.1: Interpolation Point Interlacing in Complementary $K = 3$ Models

Although the viewpoints of the two approximations in $\hat{\epsilon}(s)$ are designed to be complementary, it is possible that both may miss a given feature in the original dynamic model. In this situation, $\hat{\epsilon}$ does not indicate this missing feature and the error estimate is incorrect. Such difficulties tend to be associated with lightly damped poles that are not identified by either of the reduced-order models. This issue is taken up again in Section 5.3, although Section 5.3 shows $\hat{\epsilon}$ to be appropriate in most circumstances.

The cost of generating the error estimate involves the computation of a second reduced-order model and also the evaluation of (5.1). Generating an additional approximation simply doubles the cost for the appropriate algorithm in Table 4.4. Various possibilities and associated costs exist for evaluating $\hat{\epsilon}(s)$. Computing the $\mathcal{H}_{\infty}$ norm of (5.1) via conventional eigenvalue analyses of Hamiltonian matrices [88] costs a minimum of $30(2m)^3$ flops [3]. There is typically a total cost on the order of $100M^4$ flops when the error is estimated after each iteration of the algorithm, e.g., find $\|\hat{\epsilon}_1\|_{\infty}$, $\|\hat{\epsilon}_2\|_{\infty}$, $\ldots$, $\|\hat{\epsilon}_M\|_{\infty}$. This cost can be scaled back slightly by not evaluating the error at every iteration. An alternative approach to computing $\hat{\epsilon}$ is to simply evaluate it at multiple points. In practice, approximately $8m$ well-placed points in the fashion of the frequency-response algorithms of [89] give reasonable results for our purposes. By transforming the $\hat{A}$ and $\hat{E}$ matrices into upper-Hessenberg form ($8m^3$ flops [3]) prior to evaluation [9], the cost of computing $\hat{\epsilon}$ at the fixed points can be kept at roughly $16m^3$ flops per iteration. The total cost to estimate the error throughout the algorithm is thus on the order of M^4 flops. From Table 4.4, the cost to evaluate $\hat{\epsilon}$ does not typically exceed the cost of actually generating the reduced-order model by the dual rational Arnoldi approach. However, it is possible (given easily exploitable sparsity) that the $\hat{\epsilon}(s)$ evaluation may eclipse the cost of actually generating a reduced-order model via rational Lanczos. This model difference approach may not be appropriate for use with the RL implementation.

Assuming that the cost of generating a second model dominates the computation of $\hat{\epsilon}$, one may still wonder if this apparent doubling of required effort is worthwhile. There are possible savings gained through an error estimate, which may more than offset the doubling of computation. With a sense of the modeling error, one can adapt the utilization of interpolation points according to the dynamics absent in the model (see Section 6.4). Adaptive interpolation point placement selection can lead to acceptable models for smaller values of M and thus save on work and storage. An error estimate also suggests a stopping criterion, e.g., the model size is assumed sufficient when the error drops below a certain level. A good stopping criterion allows one to avoid excessive and wasteful iterations. Although the use of an error estimate may increase the work per iteration, a knowledge of $\hat{\epsilon}$ can reduce the total number of iterations required. For implementations where effort grows quadratically with m, e.g., dual rational Arnoldi, a reduction in total iterations is especially significant.

The error estimate can oftentimes be improved through a simple modification of (5.1). Recall that we are ultimately interested in the acceptable, low-order model represented by $\hat{\mathbf{h}}(s)$. The second low-order function $\hat{\mathbf{h}}_\perp(s)$ serves only to estimate the true frequency response $\mathbf{h}(s)$. A better approximation for $\mathbf{h}(s)$ can be found by combining the information in the two sets of projection matrices V, Z and $V_\perp, Z_\perp$ to obtain a $2M^{th}$ order model, $\hat{\mathbf{h}}_\cup(s)$. The updated error estimate then becomes

$$\hat{\epsilon}^+(s) = \hat{\mathbf{h}}(s) - \hat{\mathbf{h}}_\cup(s). \tag{5.2}$$

Similar to (5.1), one hopes that this new error estimate is large wherever $\hat{\mathbf{h}}(s)$ has not converged due to the presence of the $V_\perp, Z_\perp$ directions in the other reduced-order model.

This new estimate is superior to (5.1) from the perspective that the new error approximation can be expected to drop to zero whenever $\hat{\mathbf{h}}(s)$ converges. Unlike $\hat{\mathbf{h}}_\perp(s)$, $\hat{\mathbf{h}}_\cup(s)$ includes the original V and Z directions and, thus, tends to at least converge wherever $\hat{\mathbf{h}}$ does. From a cost standpoint, generating $\hat{\epsilon}^+$ may require 2^2 times as much effort as $\hat{\epsilon}$. The projection matrices $V_\perp$, $Z_\perp$ must be (bi)orthogonalized against V, Z in generating $\hat{\mathbf{h}}_\cup$. Moreover, the cost of evaluating $\hat{\epsilon}^+$ is larger than before. The advantages of $\hat{\epsilon}^+$ must be significant if (5.2) is to be preferred over (5.1).

5.2 Residual Expressions

An alternative approach to quantifying the modeling error is through the previously discussed residual expressions,

$$\mathbf{r}_b(s) = b - (sE - A)V\hat{\mathbf{x}}_b \quad \text{and} \quad \mathbf{r}_c(s) = c - (sE - A)^T Z\hat{\mathbf{x}}_c.$$

Residual expressions are a significant tool for quantifying the error in iterative linear system solving. It is known that very simple relations for the residuals arise in many Arnoldi and Lanczos contexts. Given the role of the dual systems (1.2) in the model reduction, it is not surprising then that the residuals are pertinent to the modeling error as well. Residuals were utilized in [59] for the partial realization problem. We formalize a new, fundamental relationship between the residuals and the modeling error in the following result.

Theorem 5.1 *The difference between the frequency responses of the original and reduced-order systems is* $\mathbf{r}_c^T(A - sE)^{-1}\mathbf{r}_b$.

Proof: Starting from the frequency-response definitions in (2.5) and (2.8), the modeling error is

$$
\begin{aligned}
\epsilon(s) &= c^T(sE-A)^{-1}b - \hat{c}^T(s\hat{E}-\hat{A})^{-1}\hat{b} \\
&= c^T(sE-A)^{-1}b - c^T V\hat{\mathbf{x}}_b \\
&= c^T(sE-A)^{-1}\{b-(sE-A)V\hat{\mathbf{x}}_b\} \\
&= c^T(sE-A)^{-1}\mathbf{r}_b.
\end{aligned} \tag{5.3}
$$

By the Petrov-Galerkin conditions in Section 2.3.1, $Z^T\mathbf{r}_b = 0$, so that (5.3) can be further expanded as

$$
\begin{aligned}
\epsilon(s) &= \{c^T - \mathbf{x}_c^T Z^T(sE-A)\}(sE-A)^{-1}\mathbf{r}_b \\
&= \mathbf{r}_c^T(sE-A)^{-1}\mathbf{r}_b,
\end{aligned} \tag{5.4}
$$

the desired result. ∎

Evaluating the error expression (5.4) in its entirety remains a difficult task. However, a sufficiently small $\mathbf{r}_b$ or $\mathbf{r}_c$ at some s_0 typically implies a small error at that frequency by itself. The only exception to this behavior occurs when s_0 is near an eigenvalue of (A, E) so that elements of $(A - s_0E)^{-1}$ grow large. Analogous to Section 5.1, large errors in $\hat{\mathbf{h}}(s)$ due to the presence of weak poles along the imaginary axis may not be adequately reflected in the residual.

It is stressed that monitoring $\mathbf{r}_b$ and/or $\mathbf{r}_c$ does not directly lead to an estimate for the modeling error. Acquiring the modeling error requires an inverse of $(sE - A)$ which is not possessed. Rather, one must concentrate on the trends in the residual behavior as s and m vary. Attempting to gauge these trends demands the evaluation of $\mathbf{r}_b$ and/or $\mathbf{r}_c$ at numerous values of s. Fortuitously, the residual expressions of many of the implementations in Chapter 4 simplify through the following result.

Lemma 5.1 *The matrices of the general RK algorithm satisfy*

$$(A - sE)V_m = Q_{m+1} \left\{ \begin{bmatrix} \uparrow & & & \uparrow \\ & & & \\ i_{p_1+1} & \cdots & \cdots & i_{p_m+1} \\ & & & \\ \downarrow & & & \downarrow \\ 0 & \cdots & \cdots & 0 \end{bmatrix} \begin{bmatrix} \gamma_1^v & \bar{v}_2 & \cdots & \uparrow \\ & \gamma_2^v & & \bar{v}_m \\ & & \ddots & \downarrow \\ & & & \gamma_m^v \end{bmatrix}^{-1} \right. \tag{5.5}$$

$$\left. + \begin{bmatrix} \bar{q}_2 & \cdots & \uparrow \\ \gamma_2^q & & \bar{q}_{m+1} \\ & \ddots & \downarrow \\ & & \gamma_{m+1}^q \end{bmatrix} \begin{bmatrix} \sigma_1 - s & & \\ & \ddots & \\ & & \sigma_m - s \end{bmatrix} \right\}$$

and

$$(A - sE)^T Z_m = W_{m+1} \left\{ \begin{bmatrix} \uparrow & & & \uparrow \\ & & & \\ i_{p_1+1} & \cdots & \cdots & i_{p_m+1} \\ & & & \\ \downarrow & & & \downarrow \\ 0 & \cdots & \cdots & 0 \end{bmatrix} \begin{bmatrix} \beta_1^z & \bar{z}_2 & \cdots & \uparrow \\ & \beta_2^z & & \bar{z}_m \\ & & \ddots & \downarrow \\ & & & \beta_m^z \end{bmatrix}^{-1} \right. \tag{5.6}$$

$$\left. + \begin{bmatrix} \bar{w}_2 & \cdots & \uparrow \\ \beta_2^w & & \bar{w}_{m+1} \\ & \ddots & \downarrow \\ & & \beta_{m+1}^w \end{bmatrix} \begin{bmatrix} \sigma_1 - s & & \\ & \ddots & \\ & & \sigma_m - s \end{bmatrix} \right\}$$

for $m \geq 1$.

Lemma 5.1 writes the $(sE - A)V$ and $(sE - A)^T Z$ matrices in the residual expression as the product of a fixed matrix of size $N \times (m+1)$ and a low-order frequency-dependent matrix. This ability is extremely beneficial in the rational Lanczos versions because $\bar{v}_m = \bar{z}_m = 0$ for all m. Recalling the definition of $\hat{A}$ and $\hat{E}$ in (4.18) and (4.21) for rational Lanczos, (5.6) reduces to

$$
\begin{aligned}
(A - sE)^T Z_m &= W_m \left(I_m + \hat{E}_m \begin{bmatrix} (\sigma_1 - s) & & \\ & \ddots & \\ & & (\sigma_m - s) \end{bmatrix} \right) + w_{m+1} i_m^T \beta_{m,m} (\sigma_m - s) \\
&= W_m (\hat{A} - s\hat{E})^T + w_{m+1} i_m^T \beta_{m,m} (\sigma_m - s). \qquad (5.7)
\end{aligned}
$$

The RL output residual vector after M iterations is therefore

$$
\begin{aligned}
\mathbf{r}_c(s) &= c - W(s\hat{E} - \hat{A})^T (s\hat{E} - \hat{A})^{-T} \hat{c} + \beta_{M,M}(\sigma_M - s) w_{M+1} i_M^T \hat{\mathbf{x}}_c \\
&= W i_1 \beta_{1,1} - W\hat{c} + \beta_{M,M}(\sigma_M - s) w_{M+1} i_M^T (s\hat{E} - \hat{A})^{-T} \hat{c} \\
&= w_{M+1} i_M^T (s\hat{E} - \hat{A})^{-T} \hat{c} \beta_{M,M} (\sigma_M - s) \qquad (5.8)
\end{aligned}
$$

due to the definition of $\hat{c}$ in (4.22). Through (5.8), the norm of the output residual can be computed as the norm of w_{M+1} times a frequency-dependent function composed of low-order matrices. The norm of w_{M+1} need only be computed once regardless of the number of times $\mathbf{r}_c(s)$ is evaluated. Because i_M is only nonzero in its last element, the function $i_M^T(s\hat{E} - \hat{A})^{-T}\hat{c}$ can be rapidly evaluated at an arbitrary frequency s_0. One need only (a) find a left transformation to place the upper-Hessenberg matrix $(s_0\hat{E} - \hat{A})^T$ into upper-triangular form and (b) perform a single step of back-substitution to find the last element in $\hat{\mathbf{x}}_c(s_0)$. These upper-triangular transformations in the first step are easily updated with $4m$ flops from one iteration to the next, because $\hat{A}_{m-1}^T$ and $\hat{E}_{m-1}^T$ are the leading minors of the upper-Hessenberg matrices $\hat{A}_m^T$ and $\hat{E}_m^T$. Assuming the residual is evaluated at $8m$ points, the cost of iteratively updating the norm of $\mathbf{r}_c$ is approximately $2N + 32m^2$ flops per iteration. The overall cost is roughly $2MN + 5M^3$ flops, a value that compares favorably to the expense of computing the reduced-order model with rational Lanczos.

For the other versions of the RK algorithm, expressions (5.5) and (5.6) must be used to generate the residuals. Updating the residual norms from one iteration to the next involves $O(mN)$ operations to iteratively update $Q_m^T Q_m$ or $W_m^T W_m$ and an additional $O(m^2)$ operations per frequency point to solve a low-order, upper-Hessenberg system of equations. In the general case, the residual evaluation across M iterations involves $O(M^2N + M^4)$ operations, a cost that is comparable to the treatment of $\hat{\epsilon}$ in the previous section.

5.3 Comparisons

The computation of the modeling error involves a common theme: balancing computational expense against the quality of the results. The two methods of this chapter provide error estimates at costs that are no larger than the modeling expense itself. The residual tends to be a slightly cheaper estimation, particularly in the case of the efficient rational Lanczos algorithm. To evaluate its performance versus that of the complementary model comparison (5.1), we analyze several examples. Example 5.1 considers the effectiveness of the error estimates in predicting the worst-case error in the reduced-order model .

Example 5.1 *In this example, we consider a model of tokamak plasma dynamics which was originally presented in [56]. The frequency response of the original model is shown in Figure 5.2. This response is relatively simple, due to the presence of all of the eigenvalues of (A, E) on the negative real axis. Frequency responses of this type are quite common in many applications, however.*

The dual RA algorithm can be applied to this problem for $M = 10$ iterations with $\sigma = 1$. The actual error in the frequency response $\hat{\mathbf{h}}(s)$ is indicated in the second column of Table 5.1, as m varies from 1 to 10. This difference is the maximum relative difference between $\mathbf{h}$ and $\hat{\mathbf{h}}$ over the frequencies in the range $1 \leq \omega \leq 1000$. Estimates based on complementary approximations (5.1) are presented in the third column for various m. This third column compares two reduced-order models centered respectively at $\sigma = 1$ and $\sigma_\perp = 100$. The use of (5.1) provides a reasonable error estimate in this example. The maximum product of the relative residuals is indicated in the fourth column of Table 5.1. Both $\mathbf{r}_b$ and $\mathbf{r}_c$ should be included in the residual error estimate, where possible, to take the two-sided properties of Theorem 5.1 into account. The fourth column provides an excellent estimate of the worst-case frequency-response error, as well. Either of the *approaches in Sections 5.1 or 5.2 provides an acceptable measure of the modeling error for this problem.*

Beyond a worst-case bound, the user may be interested in an error estimate across the frequency range. Such an estimate may be useful, for example, to modify the selection of future interpolation points.

Example 5.2 *This problem arises from a partial element equivalent circuit (PEEC) model of a patch antenna structure (see [90] and experimentation in [46]). Containing* 2100 *capacitances,* 172 *inductances and* 6990 *mutual inductances, the circuit can be realized as a system of dimension* 480*. The magnitude of the frequency response of this circuit is shown in Figure 5.3. Note the presence of multiple sharp peaks resulting from lightly damped poles along the imaginary axis.*

The rational Lanczos algorithm was applied to the PEEC model for $M = 100$ *iterations. The generated approximation alternated* $J_1 = 50$ *iterations at* $\sigma^{(1)} = 1$ *GHz and* $J_2 = 50$ *iterations at* $\sigma^{(2)} = 4$ *GHz. Figures 5.4 through 5.7 present the true modeling error (dashed line) and the* $\hat{\epsilon}$ *estimate of (5.1) for* $m = 25, 50, 75$ *and* 100*. In acquiring* $\hat{\epsilon}$*, the first reduced-order model was compared to a second, complementary model generated about a single interpolation point at* $\sigma_{\perp} = 2.5$ *GHz. Although occasionally missing one of the many peaks,e.g.,* 3.5 *GHz in Figure 5.5, or occasionally overestimating a peak, e.g.,* 0.5 *GHz in Figure 5.6,* $\hat{\epsilon}$ *provides a surprisingly accurate measurement of the true error* ϵ *across the entire frequency range.*

Figures 5.8 through 5.11 compare the true error ϵ *(dashed line) to the output residual* $\mathbf{r}_c(s)$ *(dotted line) for* $m = 25, 50, 75$ *and* 100*. Recall in rational Lanczos that this residual value can be readily evaluated according to (5.8). For ease of analysis, the results of the residual computation are scaled (by the same amount at all frequencies), so that the means of the magnitudes of* ϵ *and* $\mathbf{r}_c$ *are identical in the figures. In practice, one must concentrate on the relative changes in* $\mathbf{r}_c$ *as the values of the variables* m *and* s *vary. Clearly, the residual results are not as precise as those in Figures 5.4 through 5.7. Keep in mind that the residual estimate may be generated with* $O(M)$ *less effort than (5.1).*

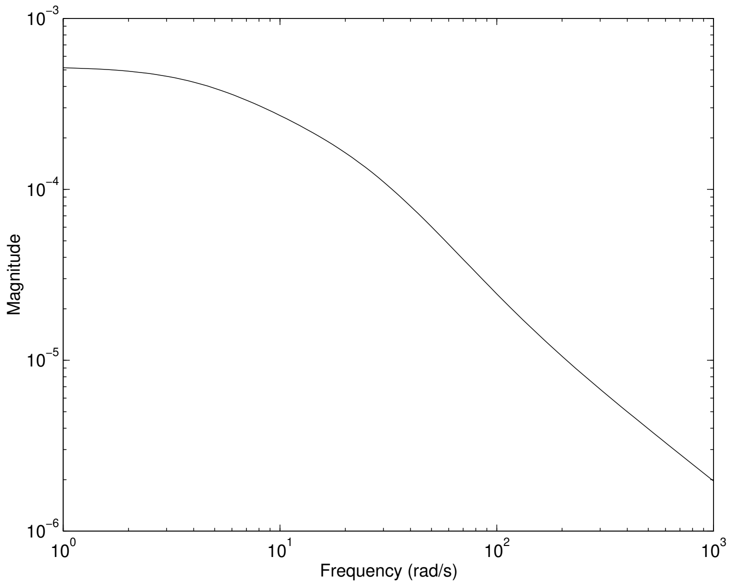

Figure 5.2: Frequency Response of Example 5.1

Table 5.1: Modeling Error Estimates of Example 5.1

m	$\max_\omega \frac{\lvert \mathbf{h}(\imath\omega)-\hat{\mathbf{h}}_m(\imath\omega)\rvert}{\lvert \mathbf{h}(\imath\omega)\rvert}$	$\max_\omega \frac{\lvert \hat{\mathbf{h}}_m(\imath\omega)-\hat{\mathbf{h}}_{\perp m}(\imath\omega)\rvert}{\lvert \hat{\mathbf{h}}_{\perp m}(\imath\omega)\rvert}$	$\max_\omega \left\{ \frac{\lvert \mathbf{r}_{c_m}(\imath\omega)\rvert}{\lVert c\rVert_2} \cdot \frac{\lvert \mathbf{r}_{b_m}(\imath\omega)\rvert}{\lVert b\rVert_2} \right\}$
1	4.6807e+00	1.3377e+00	1.5269e+00
2	1.3172e+00	5.6735e-01	3.9489e-01
3	2.2291e+00	6.9063e-01	2.7383e+00
4	1.0583e-01	1.2019e-01	2.3849e-02
5	2.4734e-02	2.5753e-02	6.6484e-03
6	2.8501e-03	2.8525e-03	3.6360e-03
7	3.0310e-03	3.2918e-03	3.4935e-03
8	1.6833e-03	1.9949e-03	1.1889e-03
9	5.3874e-03	7.5989e-03	1.2661e-02
10	2.6107e-04	1.0067e-03	4.3114e-04

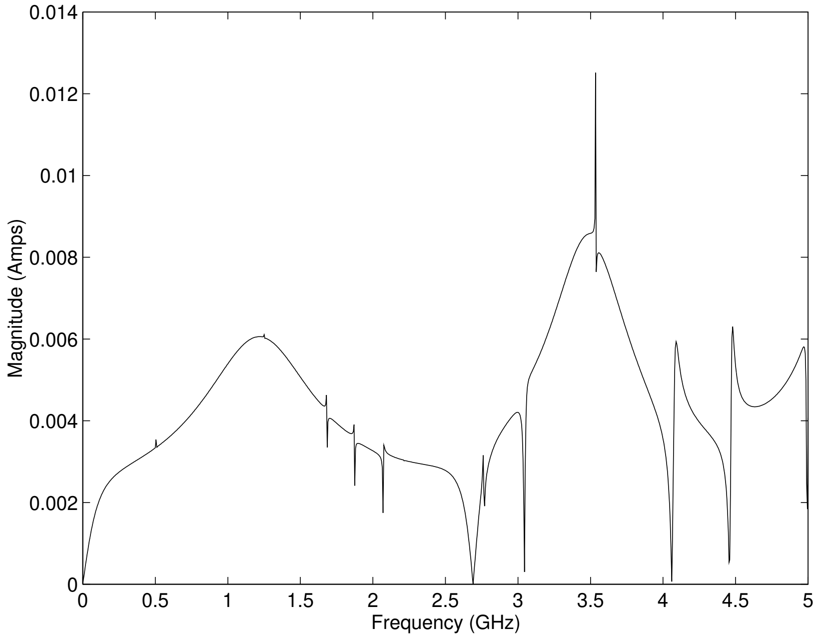

Figure 5.3: Frequency Response of Example 5.2

The residual plots are able to capture the general shape of the error curve and point out the locations of the spikes in the frequency response.

As the Examples 5.1 and 5.2 suggest, the above proposed error-estimation techniques are effective in many situations. However, the error estimates based on model differences or residuals do not always adequately treat sharp spikes in the frequency response. The assumption that a small $\hat{\epsilon}$ or residual guarantees a small error is not necessarily valid at these sharp peaks (not valid in the neighborhoods of poles along the imaginary axis). This difficulty is demonstrated in Example 5.3. The point of this example is not to completely invalidate the proposed error estimation methods. Rather, it is to demonstrate that care must be used when treating systems with lightly damped poles.

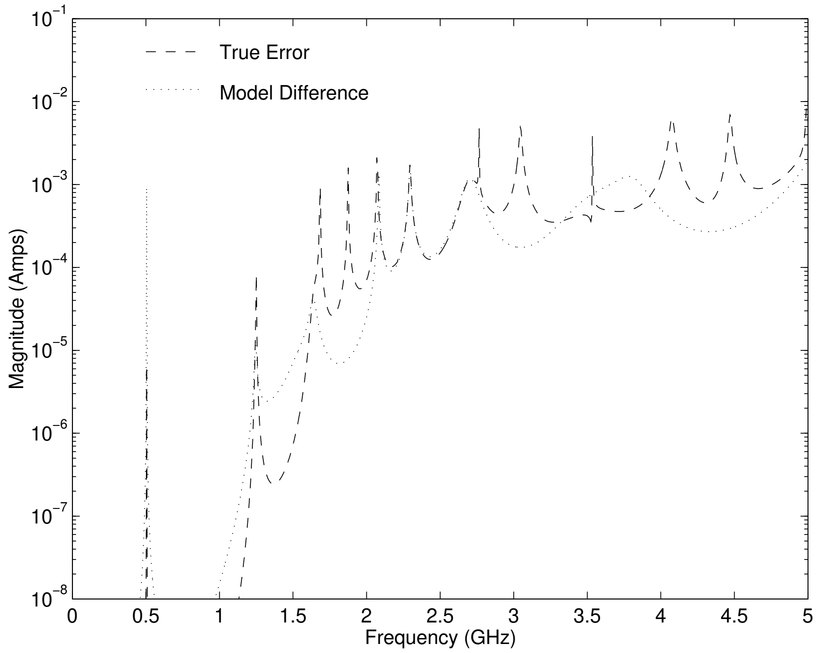

Figure 5.4: Error Estimate $\hat{\epsilon}_{25}$ in Example 5.2

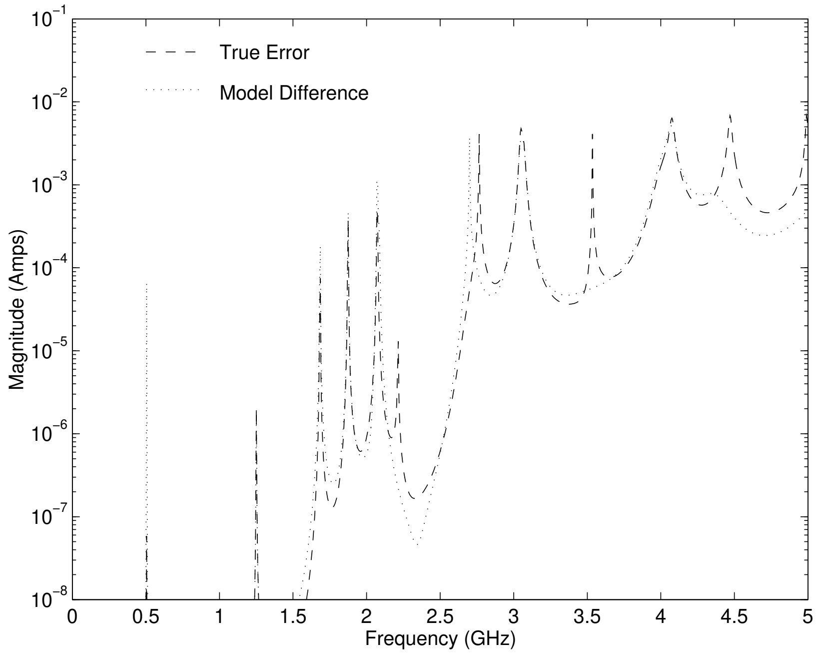

Figure 5.5: Error Estimate $\hat{\epsilon}_{50}$ in Example 5.2

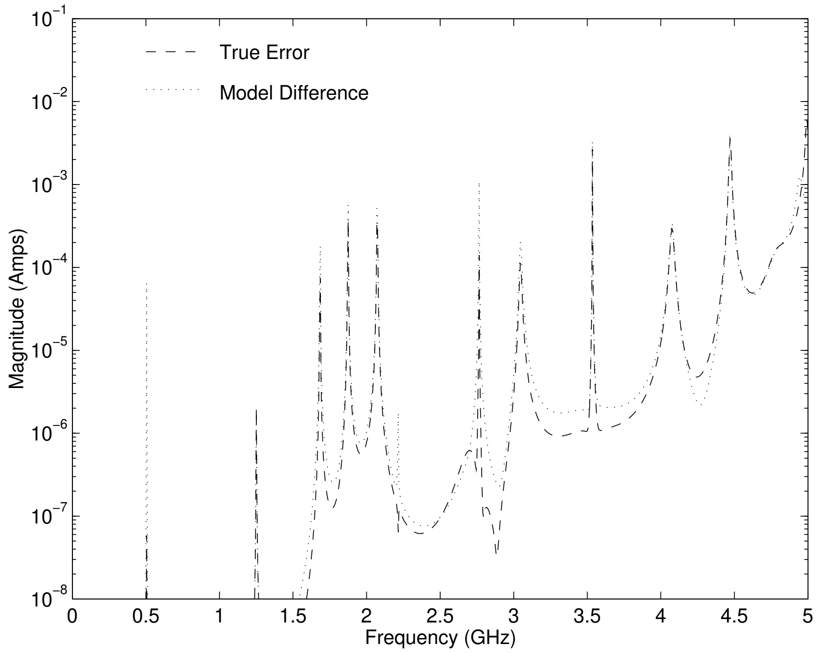

Figure 5.6: Error Estimate $\hat{\epsilon}_{75}$ in Example 5.2

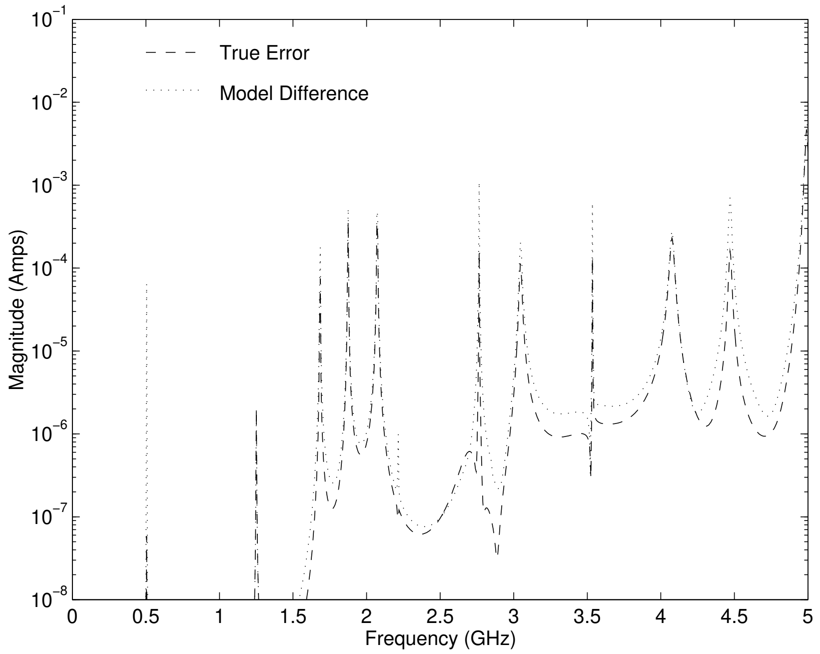

Figure 5.7: Error Estimate $\hat{\epsilon}_{100}$ in Example 5.2

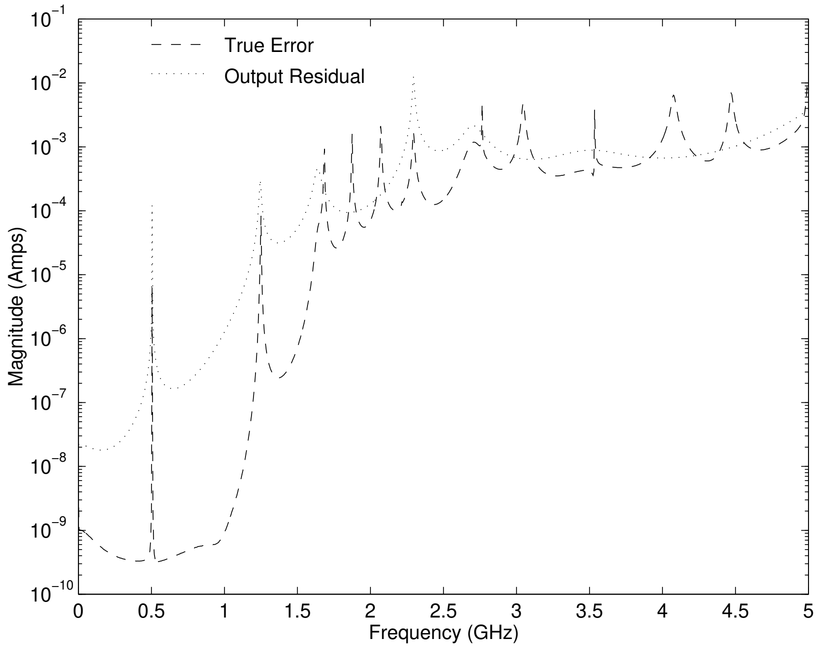

Figure 5.8: Error Estimate $\mathbf{r}_{c_{25}}$ in Example 5.2

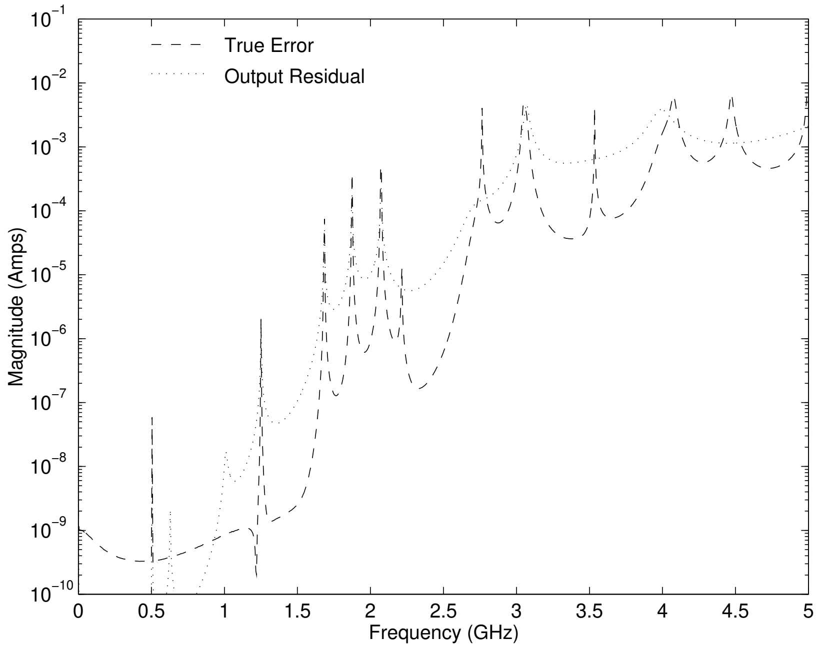

Figure 5.9: Error Estimate $\mathbf{r}_{c_{50}}$ in Example 5.2

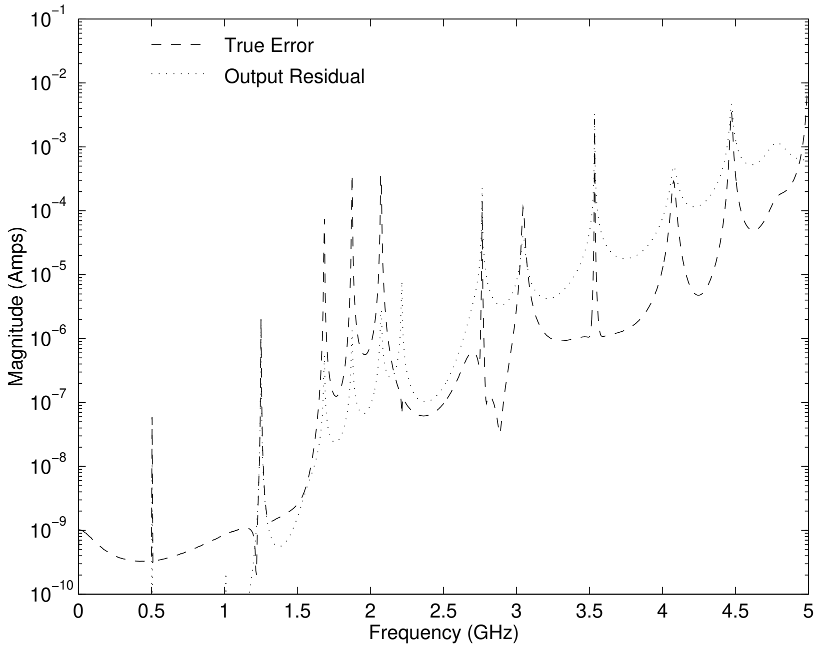

Figure 5.10: Error Estimate $\mathbf{r}_{c_{75}}$ in Example 5.2

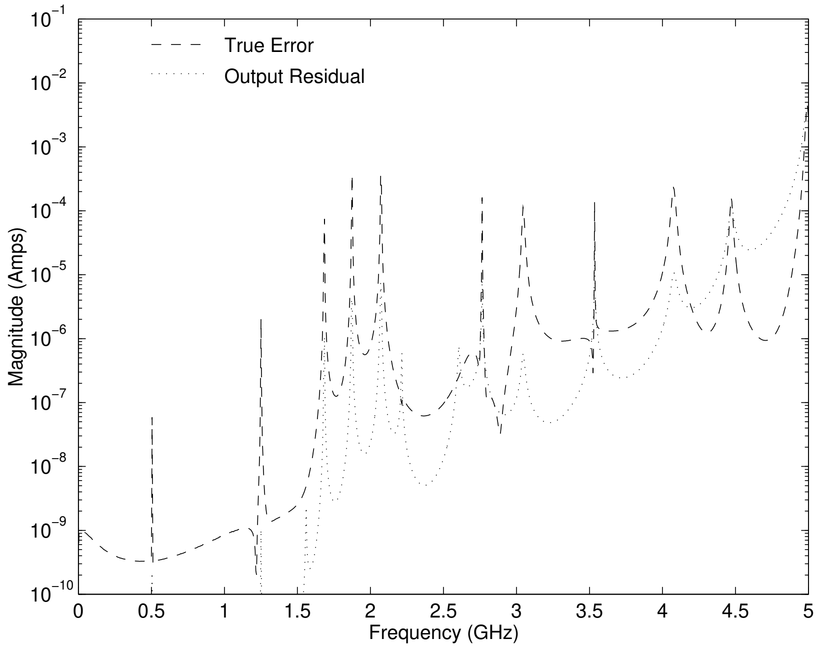

Figure 5.11: Error Estimate $\mathbf{r}_{c_{100}}$ in Example 5.2

Example 5.3 *Consider the simple seventh-order system defined by*

$$A = \begin{bmatrix} -.001 & -5 & & & & & \\ 5 & -.001 & & & & & \\ & & -1 & -5 & & & \\ & & 5 & -1 & & & \\ & & & & -7 & & \\ & & & & & -5 & \\ & & & & & & -3 \end{bmatrix}, \quad b = c = \begin{bmatrix} \sqrt{.001} \\ \sqrt{.001} \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

and $E = I$. The frequency response of this system is shown in Figure 5.12. A fourth-order model using the interpolation point $\sigma = 0.5$ has the true error shown (dashed line) in Figures 5.13 and 5.14. Note that this reduced-order model fails to capture the peak in the original frequency response at 5 rad/s.

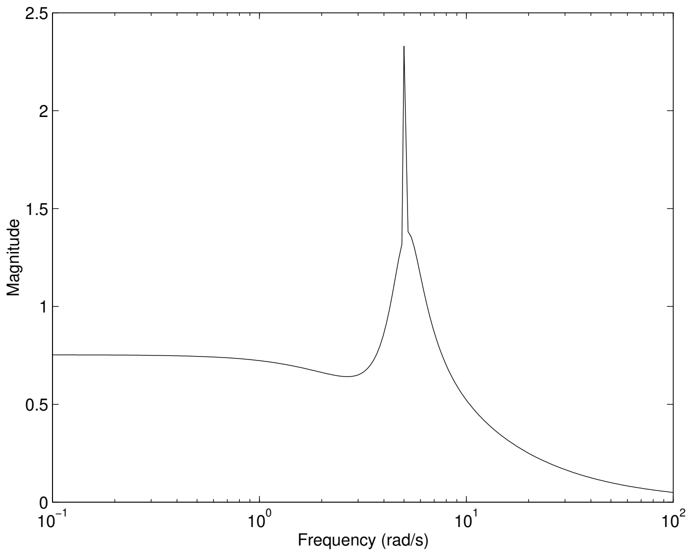

Figure 5.12: Frequency Response of Example 5.3

A second model, and associated $\hat{\mathbf{h}}_{\perp}(s)$, was constructed using the interpolation point $\sigma_{\perp} = 10$ with $M = 4$. The error estimate, based on comparing $\hat{\mathbf{h}}(s)$ and $\hat{\mathbf{h}}_{\perp}(s)$ according to (5.1), is shown as a dotted line in Figure 5.13. Unfortunately, this error estimate $\hat{\epsilon}$ falsely suggests convergence (it estimates two digits of accuracy everywhere) and misses the peak at 5 rad/s. Note also that the estimate based on (5.1) is too conservative at *low frequencies; its values there are based on the convergence of* $\hat{\mathbf{h}}_{\perp}(s)$ *rather than the behavior of* $\hat{\mathbf{h}}(s)$*.*

An error estimate based on the residuals is displayed as a dotted line in Figure 5.14. The dotted curve indicates (without scaling) the products of the relative output and input residuals,

$$\frac{\mathbf{r}_b \mathbf{r}_c}{\|b\|_2 \|c\|_2}$$

at each frequency. As before, this error estimate fails to indicate the absent peak at 5 *rad/s. The residual does provide a better estimate of the true errors at low frequencies though.*

Based on the above discussions and examples, the qualities of the proposed error-estimation techniques can be summarized. The $\hat{\epsilon}$ estimate of (5.1):

1. Provides a direct approximation to ϵ

2. Is accurate at frequencies corresponding to the complementary interpolation points $\sigma_{\perp}^{(k)}$

3. Can overestimate the error at frequencies corresponding to the primary interpolation points $\sigma^{(k)}$

4. May underestimate errors corresponding to sharp frequency peaks

A residual based error estimate:

1. Must be scaled to ϵ, particularly if only one of the residuals is computed

2. Is accurate at frequencies corresponding to the interpolation points of the reduced-order model

3. May fail to indicate errors corresponding to sharp frequency peaks

4. Typically requires less work, particularly for the $\mathbf{r}_c$ function in rational Lanczos

Some of these features are complementary, suggesting that a robust implementation might incorporate both approaches when possible.

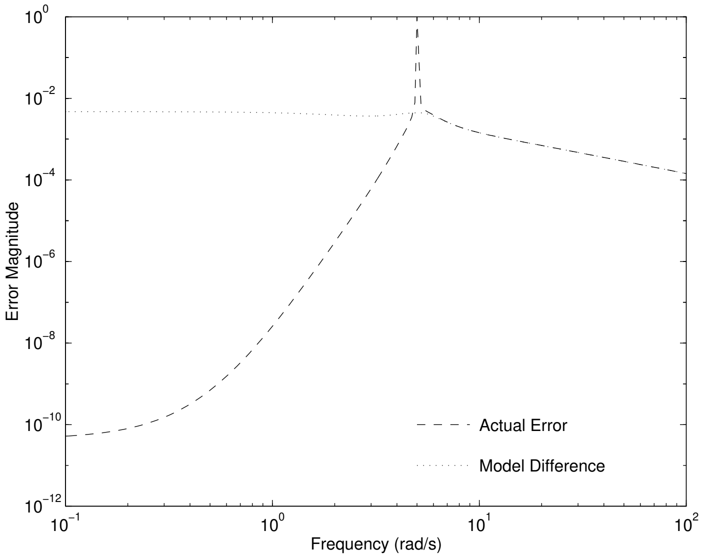

Figure 5.13: Error Estimate $\hat{\epsilon}_4$ in Example 5.3

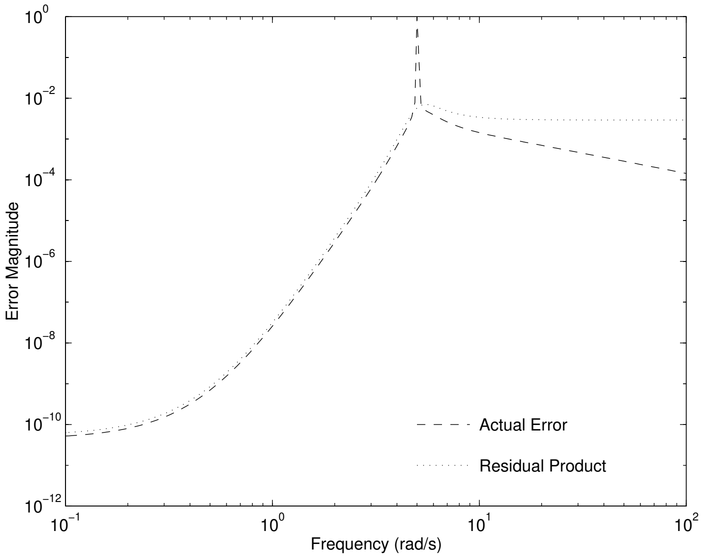

Figure 5.14: Error Estimate $\mathbf{r}_{b_4} \cdot \mathbf{r}_{c_4}$ in Example 5.3

CHAPTER 6

MODEL INTERPOLATION POINTS

The Krylov-based projection methods of Chapter 4 interpolate the value and consecutive derivatives of the frequency response of the original system at one or more points. Yet by itself, the knowledge that one is interpolating the frequency response reveals little concerning the quality of the resulting reduced-order model. For example, one can extract any combination of the poles (eigenvalues) of the original system via appropriate interpolation choices [41]. The precise location of the interpolation points and the amount of data matched at each interpolation point are the central factors in determining the accuracy and dimension of the reduced-order model.

The placement and selection of interpolation points are studied in this chapter. Connections are made between the locations of interpolation points and the convergence behavior of the model. In particular, effort is concentrated on the popular choices of purely real or imaginary interpolation points. This analysis provides insights into the relations between the model's convergence, the placement of the interpolation points, and the dynamics of the system. However, these insights are not sufficient by themselves, because the dynamics of the original system are rarely known a priori. The latter portion of this chapter focuses on the implementation of point placement and selection in practice, a nontrivial problem given that the dynamic behavior of the original system is rarely known prior to the model's convergence.

6.1 Analysis Tools

To implement rational interpolation, both the location of the interpolation points and the number of moments matched about each of the interpolation points must be specified.

Together these decisions determine both the size and accuracy of the reduced-order model. Understanding the impact of these choices on the resulting reduced-order is not alway straightforward, however. We begin by relating the choices for interpolation points to the convergence of the eigenvalues of (A, E) and to the residuals $\mathbf{r}_b$, $\mathbf{r}_c$. Past results such as Theorem 5.1, in turn, connect these quantities to the convergence of the reduced-order model.

One should understand that the convergence analyses performed in this chapter tend to consist of general trends rather than precise mathematical derivations. Unfortunately, the current understanding of the convergence of Krylov projection methods is limited, particularly when nonsymmetric matrices or multiple Krylov subspaces are involved. However, the presented level of rigor is sufficient for providing an intuition for interpolation point placement and selection. The insights of this section serve as important backdrops for the more practical implementation decisions in Sections 6.2 and 6.3.

We also initially assume that the number of interpolation points is small, so that the convergence in a given frequency range depends primarily on a single pair of dual Krylov subspaces with a corresponding interpolation point in that region. Associating a given frequency range with a single interpolation point, σ, simplifies the analysis and is reasonable in many applications. Additionally, this assumption errs on the conservative side because considering all the interpolation points at once (considering the union of information in the projector) typically only improves the convergence at a given frequency.

The role of the eigenvalues of (A, E) in the behavior of a system is grounded in the work of Bode [26]. Relations exist between the poles of the original system and the peaks in the system's frequency response. For the considered Krylov projection methods, the convergence of the eigenvalues of $(\hat{A}, \hat{E})$ to those of (A, E) is governed by the spectrum of the matrix $(A - \sigma E)^{-1}E$ in the neighborhood of $s = \sigma$. It is simple to show that if λ and $\mathbf{x}$ are an eigenvalue and eigenvector of (A, E), then $1/(\lambda - \sigma)$ and $\mathbf{x}$ are an eigenvalue and eigenvector of $(A - \sigma E)^{-1}E$. Moreover, the tendency of an eigenvalue λ to appear as an eigenvalue of the pencil $(\hat{A}, \hat{E})$ depends upon the extent that $1/(\lambda - \sigma)$ is:

1. *Positioned* on the outer edge of the spectrum of $(A-\sigma E)^{-1}E$. In particular, the eigenvalues of the original system that are closest to σ are mapped to the outside of the spectrum of $(A-\sigma E)^{-1}E$.

2. *Separated* from the other eigenvalues of $(A-\sigma E)^{-1}E$. We say $1/(\lambda-\sigma)$ is well separated, if the distance between $1/(\lambda-\sigma)$ and its closest neighbor is on the order of $|1/(\lambda-\sigma)|$.

3. *Strengthened* due to the presence of large eigenvector components in the vectors b and/or c. The residue, ρ, corresponding to a λ is a measure of this strength. The residue arises in the partial fraction expansion (2.5).

To motivate these observations, we note that the construction of each utilized Krylov subspace involves the multiplication of vectors by $(A-\sigma E)^{-1}E$. If a vector g is expanded in terms of the eigenvectors, $\mathbf{x}_n$, of (A, E) and multiplied by $(A-\sigma E)^{-1}E$, the result is (assuming distinct eigenvalues for simplicity),

$$(A-\sigma E)^{-1}Eg = (A-\sigma E)^{-1}E\sum_{n=1}^{N}\alpha_n\mathbf{x}_n = \sum_{n=1}^{N}\frac{\alpha_n}{\lambda_n-\sigma}\mathbf{x}_n.$$

Thus, those eigenvectors λ_n that are strong (α_n is large) and/or near σ (positioned on the outside of the spectrum of $(A-\sigma E)^{-1}E$) are emphasized by multiplication with $(A-\sigma E)^{-1}E$. In either case, the scaling $\alpha_n/(\lambda_n-\sigma)$ grows large. Although they may be emphasized, eigenvectors corresponding to eigenvalues in a cluster (not well separated) are all emphasized to the same extent, making it difficult to discern individual directions in the cluster. Up to a point, this analysis is similar in spirit to that for a power method [3]. We stress though that care must be taken in limiting such a comparison, the Krylov projection involves entire subspaces rather than simply single directions.

Additional relations between the interpolation points and the convergence of the reduced-order model follow from a point of view based on approximately solving the dual systems of linear equations (1.2). The error in the frequency response of the reduced-order model is $\mathbf{r}_c^T(sE-A)^{-1}\mathbf{r}_b$. As explained in Section 3.2, the speed at which these residuals are driven to zero depends on the choice of DS preconditioners used in the Krylov subspaces. The exact DS preconditioner, $P_k = (A - \sigma^{(k)}E)^{-1}$, is in turn determined by the choice of the interpolation point. Therefore, it is certainly of interest to know how the properties of a DS preconditioner relate to the use of a given interpolation point. Unlike an analysis of eigenvalue convergence, the residuals associated with the approximate solutions to (2.9) are defined and pertinent for all values of s.

The matrix $(A - \sigma E)^{-1}(A - sE)$ is at the center of the linear system solver point of view considered in Section 3.2. Recall that $P = (A - \sigma E)^{-1}$ plays the role of an exact preconditioner in a Krylov-based solver for linear systems of equations involving $A - \sigma E$. One possible path to discuss DS preconditioning and the associated matrix $(A - \sigma E)^{-1}(A - sE)$ is through the concept of clustering. A DS preconditioner may be evaluated according to the number of distinct eigenvalue clusters appearing in the spectrum of $P(A - sE)$. A small number of tightly-packed clusters is preferred in the spectrum of $P(A - sE)$ for values of s in the frequency range of interest. Two eigenvalues $\tilde{\lambda}_n$ and $\tilde{\lambda}_{n+1}$ of $(A - \sigma E)^{-1}(A - sE)$ at some $s = \eta$ are said to be clustered if their relative difference,

$$\frac{|\tilde{\lambda}_n - \tilde{\lambda}_{n+1}|}{\min(|\tilde{\lambda}_n|, |\tilde{\lambda}_{n+1}|)}, \tag{6.1}$$

is sufficiently small. A cluster at one is particularly desirable.

To motivate this concept of clustering, consider a matrix G that equals $P(A - \eta E)$ or any other generic matrix. Further assume that this matrix has γ clusters. Solving the system of equations $Gx_g = g$ via a projection technique leads to a solution that lies in the Krylov space $\mathcal{K}_J(G, g)$. If the clusters in G are tight, i.e., the eigenvalues in a cluster lie atop one another, then the dimension of $\mathcal{K}_J(G, g)$ does not exceed γ. For example, the dimension of $\mathcal{K}_J(G, g)$ is only one if G is a scaled identity matrix (the eigenvalues of G in this case are all identical) and x_g lies in $\mathcal{K}_1(G, g)$. Thus, the number of steps required by an iterative solver to find x_g does not exceed $\gamma - 1$. If the eigenvalues of G all exist at three points, for example, an exact solution arises after only two iterations. Clustering reduces the number of directions considered by an iterative method in finding the solution. Of course, as the clusters become less tight (the clustered elements are no longer exactly on top of each other), one can only say that the rank of an increasingly perturbed version of $\mathcal{K}_J(G, g)$ equals γ. As a very rough rule-of-thumb, one can expect the relative error in the solution after $\gamma - 1$ iterations to be proportional to the relative distance (6.1) between clustered eigenvalues.

The impact of the choice of σ on the spectrums of the two matrices $(A - \sigma E)^{-1}E$ and $(A - \sigma E)^{-1}(A - sE)$ is key in the following analyses of interpolation points. Not surprisingly, these two matrices are in fact related up to a simple scaling and shift by the expression (3.5).

6.2 Point Placement

Based on the desired features in the spectrums of $(A-\sigma E)^{-1}E$ or $(A-\sigma E)^{-1}(A-sE)$, various strategies for locating the interpolation points may be evaluated. We begin with the simpler single point approaches.

6.2.1 Imaginary interpolation points

The use of an imaginary interpolation point is perhaps the most logical of starting points, because one is interested in minimizing the frequency-response error $\mathbf{h}(s) - \hat{\mathbf{h}}(s)$ along the imaginary axis ($s = \imath\omega$). To aid in the analysis of this choice, an example mapping from λ to $1/(\lambda - \sigma)$ is displayed in Figure 6.1 for an imaginary interpolation point. Recall that $1/(\lambda - \sigma)$ is an eigenvalue of $(A - \sigma E)^{-1}E$ if λ is an eigenvalue of (A, E). As evident from either the expression $1/(\lambda - \sigma)$ or Figure 6.1, an imaginary interpolation point maps those eigenvalues of (A, E) near σ to well-separated positions on the outer edge of the spectrum of $(A - \sigma E)^{-1}E$. From the discussion in Section 6.1, one should therefore expect those poles of the original system nearest σ to rapidly appear in $(\hat{A}, \hat{E})$. Practical experience confirms this observation. The poles near σ tend to appear, regardless of their strength. The use of an imaginary interpolation point is a powerful tool for finding all of the information in the neighborhood near σ.

The advantage of an imaginary interpolation point locally is unfortunately its downfall globally. Although eigenvalue strength and separation come into play away from σ, one

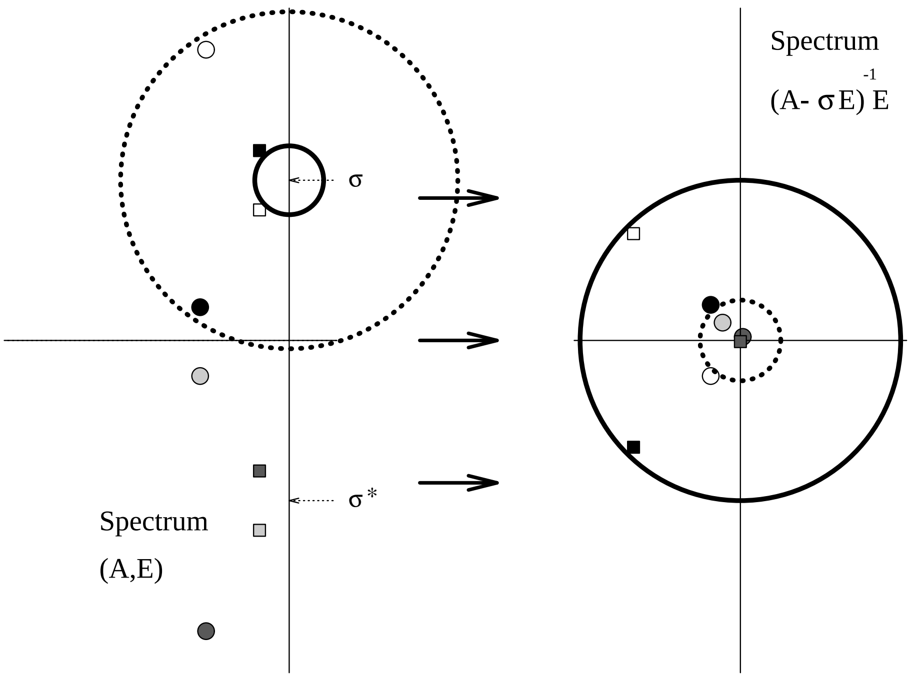

Figure 6.1: Eigenvalue Mapping for an Imaginary Interpolation Point

can roughly expect the convergence tendency of an eigenvalue of (A, E) to be inversely proportional to its distance from σ. Thus, a desired eigenvalue, λ_d, of (A, E) that is near the imaginary axis but away from σ must wait on the convergence of all the eigenvalues closer to σ (including eigenvalues that are possibly a distance $\approx |\lambda_d|$ into the left-half plane). Yet unless they are very strong, eigenvalues far into the left-half plane do not impact $\mathbf{h}(s)$. Unless very weak, an eigenvalue near the imaginary axis leads to a peak in the frequency response [26]. The convergence of λ_d may be forced to wait on an undetermined number of eigenvalues that are far into the left-half plane and nonessential to the model. In this situation, stagnation is observed in the reduction of the modeling error until these non-contributing eigenvalues are all identified. This stagnation can be further amplified if λ_d lies in a cluster of eigenvalues, or if λ_d is weak. In slowing convergence, this stagnation leads to a model of unnecessarily large size.

From a DS preconditioner point of view, the previous discussion remains pertinent. Due to Lemma 2.1, the eigenvalues of $(A-\sigma E)^{-1}(A-sE)$ are simply those of the matrix $(A-\sigma E)^{-1}E$ shifted by one and scaled by $(s-\sigma)$. The deviation of the eigenvalues of the DS preconditioned matrix from a cluster at one is directly proportional to the distance of s from σ. As s deviates from σ, numerous eigenvalues escape the cluster and the effectiveness of the DS preconditioner diminishes. The convergence of $V\hat{x}_b$ and $Z\hat{x}_c$ to the true solutions of (1.2) at values of s away from σ slows, because the iterative solver must capture the scattered eigenvalues that are outside of the cluster. The order in which these scattered eigenvalues are found depends on the same issues touched on in the previous paragraph.

To summarize, imaginary interpolation points always lead to excellent results locally, but can result in extremely slow convergence at all frequencies away from σ. An example of this behavior is provided in Section 6.4.

Before leaving the topic of imaginary points, the reader should note that those poles near the complex conjugate of the interpolation point, σ^*, are not typically near σ and are therefore mapped by $(A - \sigma E)^{-1}E$ towards the cluster at the origin. If the convergence of the poles of the original system is to occur in complex conjugate pairs, interpolation at both σ and σ^* is required. The results of interpolating at σ^* are simply the dual to those for σ.

Some comments on the implementational aspects of complex arithmetic are also in order. Although one allows complex $\sigma^{(k)}$, the avoidance of complex V and Z is desirable. Beyond potential cost savings by keeping V and Z purely real, a real reduced-order model is preferred for its consistency with the real description of the original system. An approach used to retain real operations with complex $\sigma^{(k)}$ was developed in [91] for the rational Arnoldi algorithm. The thrust of this development is that one should always treat the complex points $\sigma^{(k)}$ and $\sigma^{(k)^*}$ pairwise. Whenever an iteration with $\sigma^{(k)}$ is executed, perform a simultaneous iteration with its conjugate $\sigma^{(k)^*}$. Because of the relation

$$(A - \sigma^{(k)^*}E)^{-1}v = \{(A - \sigma^{(k)}E)^{-1}v\}^*, \tag{6.2}$$

a knowledge of one direction automatically implies the knowledge of its conjugate. The execution of these two simultaneous, conjugate iterations need only introduce two real directions real$\{(A-\sigma^{(k)}E)^{-1}v\}$ and imag$\{(A-\sigma^{(k)}E)^{-1}v\}$ into V rather than two complex ones. The use of (6.2) halves the extra effort (typically an increase by a factor of four) involved in working with the complex matrix $(A - \sigma E)$. For further details, the reader is referred to [91].

6.2.2 Real interpolation points

All of the interpolation points in the prior examples of this dissertation are real. The utility of a positive real interpolation point again follows from the mapping of the generalized eigenvalues of (A, E) to the eigenvalues of the matrix $(A - \sigma E)^{-1}E$. Such a mapping is displayed in Figure 6.2 and is determined by the following result.

Lemma 6.1 *If the initial system (2.1) is stable and σ is a positive, real number, then the eigenvalues of $(A - \sigma E)^{-1}E$ are contained in a circle of radius $\frac{1}{2\sigma}$ that is centered at $\frac{-1}{2\sigma}$.*

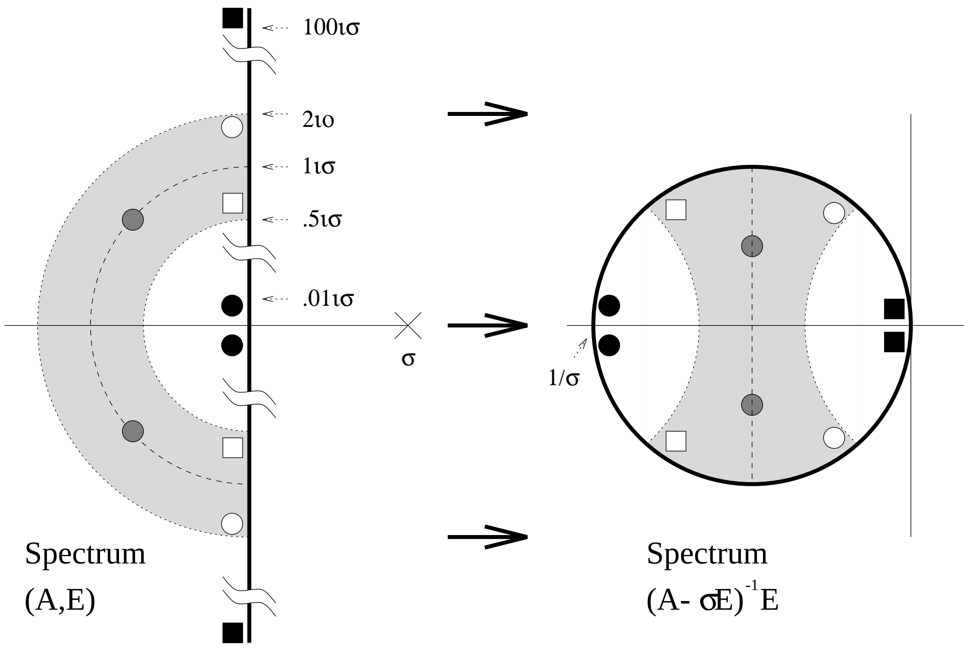

Figure 6.2: Eigenvalue Mapping for a Real Interpolation Point

According to Lemma 6.1, a value λ in the left half of the complex plane is mapped to $1/(\lambda - \sigma)$, a position inside a circle. The imaginary axis, the edge of the left-half plane, is compressed onto the edge of this circle. A simple analysis of the mapping $\lambda \rightarrow 1/(\lambda - \sigma)$ reveals that those poles of the initial system with magnitudes much less than σ are all mapped to a cluster at $\frac{-1}{\sigma}$. Those poles with magnitudes much greater than σ are all mapped to a cluster at 0. Only those poles with magnitudes on the order of σ can avoid being squeezed into a cluster at 0 or $\frac{-1}{\sigma}$. Poles that are not well separated in the spectrum of $(A - \sigma E)^{-1}E$ tend to be approximated as a single compressed pole in the reduced-order model. Hence, one can only expect poles with $|\lambda_n| \approx \sigma$ to be distinguishable in the reduced-order model. In particular, one should expect the strong poles with magnitudes on the order of σ to appear in $(\hat{A}, \hat{E})$. In Figure 6.2, these poles lie in the shaded regions.

Real positive interpolation points provide a distant view of the dynamics of a stable initial system. The position of a mapped eigenvalue $1/(\lambda - \sigma)$ and, thus, the overall convergence of the approximate eigenvalue is typically less sensitive to the value of Real(λ) when σ is real. Practical experience verifies that a pole away from the imaginary axis still appears in the model if a real interpolation point is used and if that pole is well separated and/or strong.

Example 6.1 *Consider an eigenvalue $\lambda = -.01 + \iota$ and a perturbed version of this eigenvalue, $\lambda_\delta = -.1 + \iota$. The change in the resulting mapped eigenvalue relative to the perturbation of the pole,*

$$\frac{|(\lambda - \sigma)^{-1} - (\lambda_\delta - \sigma)^{-1}|}{|\lambda - \lambda_\delta|},$$

is less than 0.5, if $\sigma = 1$, but is 1000 if $\sigma = \iota$. The difference between the mapped versions of λ and λ_δ is significant for the imaginary interpolation point, but is negligible for the real interpolation point.

Besides eigenvalue convergence, we are also interested in the approximate system solutions to (1.2) (and in turn the DS preconditioned matrix, $(A - \sigma E)^{-1}(A - sE)$) as s varies. A distance exists between a real σ and the system poles, so that small relative perturbations in s barely impact the results with the DS preconditioned matrix. The following result helps to formalize this statement.

Lemma 6.2 *If λ is an eigenvalue of (A, E), then $\tilde{\lambda} = \frac{\lambda - s}{\lambda - \sigma}$ is an eigenvalue of the matrix $(A - \sigma E)^{-1}(A - sE)$.*

Assuming a stable initial system and a positive real σ, the difference $|\lambda - \sigma|$ is at least as large as σ. Based on this observation and Lemma 6.2, the eigenvalues of

$$(A - \sigma E)^{-1}(A - sE) \quad \text{and} \quad (A - \sigma E)^{-1}(A - (s + \delta)E)$$

vary by at most δ/σ, when $s \approx \sigma\iota$. Compare this fact to the imaginary interpolation case where the change in the spectrum of the DS preconditioned matrix was directly proportional to the perturbation, δ, on the interpolation point. The DS preconditioner with a real interpolation point has the opportunity to be suitable over wider regions of s.

The potential for a real interpolation point to be effective over wide regions of s and actually being effective over broad regions are two different things. In the examples of Section 6.4, a real interpolation point leads to convergence in a wide neighborhood about $\sigma\iota$ except about a finite number of weak eigenvalues of (A, E) along the imaginary axis. For insight into this observation, we return to Figure 6.2. Because the spectrum of the DS preconditioned matrix is simply a shifted and scaled version of Figure 6.2, it is clear that a real interpolation point can leave a large number of well-separated eigenvalues in the spectrum of $(A - \sigma E)^{-1}(A - sE)$ (corresponding to all eigenvalues of (A, E) with magnitudes near σ). Our previous discussions indicate that the strong, scattered eigenvalues are the ones tending to appear in the model. The eigenvectors corresponding to these strong eigenvalues play an important part in the solutions to (1.2), because they are ones that by definition dominate b and c. Also, because the significance of these directions arises from their role in the fixed vectors b and c, their importance depends less on variations in frequency. The only points where these directions can be overshadowed are those where weaker poles crop up near the imaginary axis. The eigenvector corresponding to a weak, predominantly imaginary pole, λ_w, dominates the solution of $(A - sE)\mathbf{x}_b = b$ as s nears λ_w. Yet, because of a Krylov projection's emphasis on strong poles, this weak direction can be absent in the reduced-order model and the converge of $V\hat{\mathbf{x}}_b$ and $Z\hat{x}_c$ at $s = \iota|\lambda_w|$ delayed. In summary, a real interpolation point tends to yield a broader, but courser convergence to the true frequency response.

6.2.3 Multiple interpolation points

Now that we better understand the strengths and weaknesses of a single interpolation point, it is appropriate to turn to the more general rational interpolation problem. Our goal remains the same: a rapid and efficient convergence to an accurate reduced-order model. Combinations of interpolation points: real, imaginary and complex can be used to achieve this goal. We tend to concentrate on purely real or imaginary points to keep the development manageable.

The first proposed placement strategy, and the one favored in the previous examples of this dissertation, is to logarithmically space interpolation points between ω_{min} and ω_{max} on the real axis. A large spacing between these real points is appropriate due to the broad convergence regions of real interpolation points discussed in Section 6.2.2. The use of one interpolation point per every other order of frequency magnitude is recommended. For a large class of problems, this strategy involves only a single interpolation point. In general, only a very small number of interpolation points are required by this technique. This strategy is thus appropriate when the cost of factorizing $(A - \sigma^{(k)}E)$ is large. The use of real interpolation points is also preferred for the rational Lanczos algorithm, as it allows for a small bandwidth in $\hat{A}$ and $\hat{E}$. The only drawback of real interpolation points are the previously mentioned difficulties with lightly damped poles (sharp peaks in the frequency response). Numerous iterations may be required, if such points are to be found.

Spaced interpolation points along the imaginary axis are a second possibility. Rapid convergence in the frequency response can be expected with this strategy about the K neighborhoods centered at $s = \iota\sigma^{(k)}$. Either linear or logarithmic spacing can be utilized to adjust this behavior to the frequency range of the problem. An accurate, low-order model can be expected given a sufficient number of imaginary interpolation points, but the value of this number is rarely known a priori. Underestimating the number of required imaginary interpolation points can lead to stagnated convergence away from the $\sigma^{(k)}$. Frequency-response peaks due to weak eigenvalues along the imaginary axis may be untouched, if these peaks are located between and away from the preset imaginary interpolation points. That is, problems may arise if the local convergence regions about each $\sigma^{(k)}$ do not readily overlap. Avoiding such problems requires either an adaptive introduction of extra interpolation points during the modeling process or a fine grid of closely spaced points. Either strategy, and especially the latter, is undoubtedly costly as many factorizations of $(A - \sigma^{(k)}E)$ are involved. Possible algorithmic approaches for overcoming these expenses are proposed in Chapters 7 and 8. Further details on adaptive point placement and selection are provided in Section 6.3.

Of course, a combination of real and imaginary interpolation points is possible. The distanced real points produce the general features of the frequency response. Imaginary interpolation points may then be introduced as needed to capture the exact behavior in user-specified frequency ranges. The imaginary interpolation points can be used to weigh desirable application-specific features which are known to the user.

Even though it is not implemented in the following, there is another approach for selecting multiple interpolation points which is of interest. This approach is due to [92] and actually leads to an optimal reduced-order model in a certain sense. It is shown in [92] that the $\mathcal{L}_2$ norm of the inverse Laplace transform of $\mathbf{h}(s) - \hat{\mathbf{h}}(s)$ is minimized if $\hat{\mathbf{h}}(s)$ matches the values of $\mathbf{h}(s)$ at $\sigma_m = -\hat{\lambda}_m$, $m = 1, 2, \ldots, M$, where the $\hat{\lambda}_m$ are the poles of the reduced-order model. Unfortunately, these values $-\hat{\lambda}_m$ where interpolation is proposed are not known a priori. An algorithm is proposed in [93] for iteratively locating these points, but both the convergence and efficiency of this procedure are questionable for large-scale problems. Regardless of the feasibility of the implementation, this approach does suggest that a combination of real and imaginary interpolation points can be preferred over a single interpolation point.

6.3 Point Selection

Once the values of the interpolation points are set, their ordering and use (the values for J_k) need to be specified. Schemes based on simply alternating the interpolation points at each iteration were already seen in Chapter 4. Alternating points was particularly important in the rational Lanczos algorithm, because it promoted a small bandwidth in the reduced-order matrices. Yet even with such a simple scheme, one may not wish to match the same number of moments about every interpolation point. For example, the error over a certain frequency range might be observed to drop much faster than over another. Frequently, such information only arises as a picture of the system develops during the modeling process. Thus, even in the simplest schemes, an adaptive control of the interpolation point selection may be worthwhile.

It is logical to base adaptive interpolation decisions on one of the error estimates developed in Chapter 5. The interpolation points of new iterations can be specified as m grows with the goal of reducing the remaining error in the approximation. Various degrees of effort can placed towards controlling the future modeling steps via some error estimate $\hat{\epsilon}$.

6.3.1 Adaptive termination

In this simplest of approaches, one utilizes $\hat{\epsilon}$ to specify only the number of moments ($2J_k$) matched about each interpolation point. The placement and ordering of the interpolation points, e.g., alternated in consecutive blocks, etc., are specified prior to execution. A given $\sigma^{(k)}$ is then utilized in the prescribed ordering until $\hat{\epsilon}$ drops below an acceptable level across the frequency ranges that correspond to $\sigma^{(k)}$. If imaginary points are used, for example, one ceases to use $\sigma^{(k)}$ when the modeling error is small for all frequencies between it and its nearest neighbors. This approach is even pertinent for $K = 1$; the corresponding frequency range simply runs from ω_{min} to ω_{max} in this case.

6.3.2 Adaptive selection

This approach utilizes the error estimate to determine both the ordering of the $\sigma^{(k)}$ and the values of J_k for some predetermined set of interpolation points. One simply chooses the interpolation point for the $(m+1)^{st}$ iteration that is closest to the frequency where $\hat{\epsilon}_m$ is largest. Intuitively, such an approach is pleasing as one strives to reduce the maximum error at each step.

6.3.3 Adaptive placement

Taking the previous approach one step farther, one can adaptively place a completely new interpolation point in the m^{th} iteration at the frequency where $\hat{\epsilon}_m$ is largest. The set of interpolation points is not fixed a priori, but grows during the process. To limit this growth, one may choose to perform multiple iterations before changing to a new point. If m is an iteration where a new interpolation point is chosen, a good rule of thumb is to persist with this chosen interpolation point until the frequency-response change in some future iteration, e.g., $\hat{\mathbf{h}}_{m+f}(s) - \hat{\mathbf{h}}_{m+f-1}(s)$, drops to less than 10% of $\hat{\mathbf{h}}_m(s) - \hat{\mathbf{h}}_{m-1}(s)$. That is, remain with a given interpolation point until signs of stagnation arise. In this manner, one attempts to obtain full benefits from an interpolation point before suffering the costs involved with moving to another one. Note that adaptive placement is only appropriate for imaginary interpolation points; real ones possess a broader range suited for the previously mentioned adaptive selection.

Example 6.2 *Consider applying the three adaptive schemes to find an approximation between* 1 *and* 10 *rad/s. In adaptive termination (Section 6.3.1), one might start by using* $\sigma^{(1)} = 1$ *in odd iterations and* $\sigma^{(2)} = 10$ *in even iterations. One stops utilizing* $\sigma^{(1)}$ *as an interpolation point, as soon as* $\hat{\epsilon}$ *becomes small over* 1 *to* 5 *rad/s. Likewise,* $\sigma^{(2)}$ *is no longer used after* $\hat{\epsilon}$ *becomes small over* 5 *to* 10 *rad/s. In adaptive selection (Section 6.3.2), the interpolation point utilized in the* m^{th} *iteration* (σ_m) *depends on* $\hat{\epsilon}_{m-1}$*. If* $\hat{\epsilon}_{m-1}$ *is largest over* 1 *to* 5 *rad/s, then* $\sigma_m = 1$*; otherwise,* σ_m *is set to be* 10*. In adaptive* *placement (Section 6.3.3), σ_m is chosen to be $\sqrt{-1}$ times the frequency in the range $1 \leq \omega \leq 10$, where $\hat{\epsilon}_{m-1}(\iota\omega)$ is largest.*

Although the use of adaptation increases in going from the point selection approaches of Sections 6.3.1 to 6.3.3, the performance need not increase accordingly. For both practical and theoretical reasons, the adaptive selection scheme may not be preferred over the simpler combination of regularly alternating interpolation points and adaptive termination. Assuming exact DS preconditioners and infinite precision, only the values of J_k, and not the interpolation point ordering, determine the reduced-order model. In theory, the approaches in Sections 6.3.1 and 6.3.2 should yield similar results. Furthermore, the error estimate $\hat{\epsilon}$ used by adaptive selection or placement is only an approximation. Regularly alternating the interpolation points as in Section 6.3.1 insures that information is matched across the entire frequency range.

From a cost standpoint, care should also be taken before abandoning the simpler adaptive termination scheme. Interpolation point-ordering based on $\hat{\epsilon}$ (Section 6.3.2) destroys the banded structure in the rational Lanczos method. The adaptive placement of points (Section 6.3.3) may lead to a large number of imaginary interpolation points. Although many points may speed convergence, a large K is most likely impractical for the algorithms of Chapter 4 (the algorithms seen up to this point).

6.4 Comparisons

To compare various interpolation point placement and selection techniques, one can generate reduced-order models according to the following five strategies:

Strategy 1. A single real interpolation point at ω_{max}

Strategy 2. Two imaginary interpolation points at $\pm\omega_{min}$

Strategy 3. A real interpolation point at every other order of magnitude

Strategy 4. Five conjugate pairs of linearly spaced imaginary interpolation points

Strategy 5. Adaptively placed imaginary interpolation points according to $\max(\epsilon)$

Strategy 1, a real point at ω_{max}, is endorsed in [48] while Strategy 2, an imaginary point at ω_{min}, is quite common throughout the literature. Strategies 3 and 4 were discussed in Section 6.2.3. Strategy 5 is an adaptive strategy that places a new interpolation point (after every ten iterations in Examples 6.3 and 6.4) wherever the current modeling error is greatest. This strategy is not implementable in practical situations, because only an error estimate $\hat{\epsilon}$ is commonly available. Strategy 5 serves as a rough bound on the best achievable error and is, thus, useful for comparisons. The expected behavior of each strategy follows from Section 6.2. Recall that a real interpolation point, $\sigma^{(k)}$, tends to induce a course convergence over the approximate range $0.1\sigma^{(k)} < \omega < 10\sigma^{(k)}$, while an imaginary interpolation point, $\sigma^{(k)}$, tends to induce precise convergence in the neighborhood of $s = \iota\sigma^{(k)}$.

Example 6.3 *We return to the compact disc player considered in Example 2.1. In this example, the CD subsystem to be approximated relates the position of the player's focusing lens to inputs at the radial arm actuator. The frequency response of this subsystem is shown in Figure 6.3.*

One hundred RK iterations were executed with each of the five interpolation point strategies listed at the beginning of this section. The specific interpolation points corresponding to each of these strategies are listed in Table 6.1. The model reduction was implemented with a reorthogonalized version of the dual rational Arnoldi to insure that the quality of the approximation depended solely on interpolation point placement and selection. The relative errors in the reduced-order model as both the interpolation point strategy and the number of iterations varies are presented in Table 6.2. The error is measured by a computed $\mathcal{H}_2$ norm, which is uniformly weighted over the frequency range $\omega_{min} = 0.1$ to $\omega_{max} = 10^5$ rad/s.

With the exception of the single real interpolation point case, the strategies of Table 6.2 perform comparably well. The difficulties with a single interpolation point follow from

Table 6.1: Interpolation Point Strategies of Example 6.3

Strategy	Interpolation Point(s)
1	10^5
2	$\pm\iota$
3	1, 100, 10^4
4	$\pm\iota$, $\pm 10\iota$, $\pm 100\iota$, $\pm 10^3\iota$, $\pm 10^4\iota$
5	$\pm\iota$, $\pm 100\iota$, $\pm 3000\iota$, $\pm 600\iota$, $\pm 10^4\iota$, $\pm 4200\iota$, $\pm 1.8 \cdot 10^4\iota$, $\pm 1.6 \cdot 10^4\iota$, $\pm 3.9 \cdot 10^4$, $\pm 300\iota$

Table 6.2: Convergence with the Interpolation Point Strategies of Example 6.3

m	Strategy 1	Strategy 2	Strategy 3	Strategy 4	Strategy 5
10	1.0000e+00	2.6246e+00	9.7267e-01	1.1844e+00	2.7315e+00
20	1.0002e+00	2.3167e-01	1.3378e-01	4.0014e-01	4.4137e-02
30	1.0001e+00	6.0595e-03	6.5276e-02	1.6369e-01	9.9232e-03
40	1.0002e+00	1.1386e-03	1.5089e-03	1.8898e-03	6.1080e-04
50	1.0002e+00	5.3858e-04	1.7842e-03	1.4359e-03	2.0964e-04
60	1.0001e+00	7.5327e-04	4.2890e-04	2.9126e-04	3.3147e-04
70	1.0002e+00	3.8352e-04	3.0646e-04	1.2832e-04	8.5592e-05
80	1.0036e+00	8.3960e-05	1.0041e-04	8.2559e-05	1.9209e-05
90	1.0052e+00	1.0465e-05	7.5123e-06	1.4374e-05	7.9303e-06
100	1.0395e+00	4.6947e-06	1.9490e-05	1.1960e-05	4.5992e-06

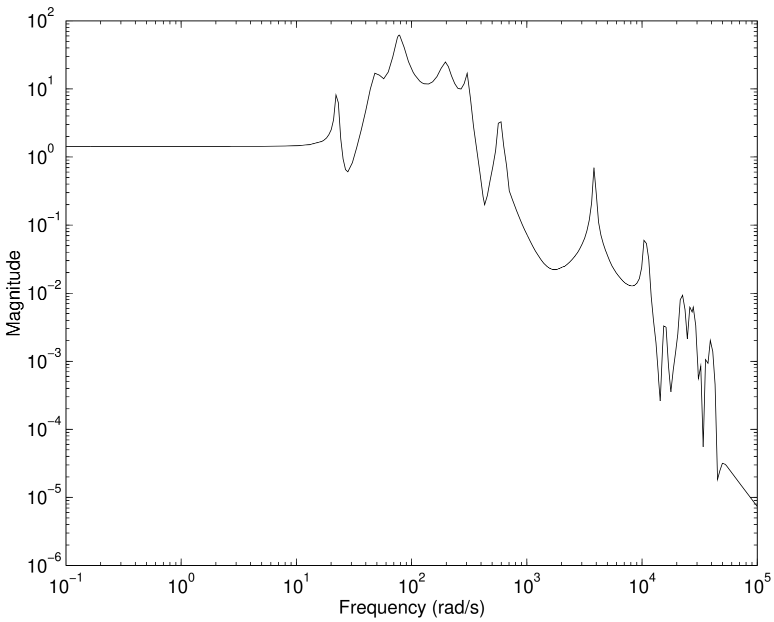

Figure 6.3: Frequency Response of Example 6.3

the portion of the spectrum of (A, E) in Figure 6.4. As one moves up and way from the origin in this figure, the vertical spacing between the eigenvalues grows proportionally with ω, while the horizontal spacing only gradually increases. The vertical spacing for the eigenvalues across the top of the figure is on the order of 10^4, while it drops to order 10^1 as one approaches the origin.

To a real interpolation point at 10^5, only the vertical spacings on the order of 10^4 are significant. The eigenvalues across the top of the figure appear well separated, while those eigenvalues with imaginary parts $< 10^4$ appear as a large cluster at the origin. Unfortunately for Strategy 1, the individual eigenvalues in this cluster dominate the frequency response. Strategy 1 stagnates, because it emphasizes the eigenvalues across the top of Figure 6.4. Strategy 2, on the other hand, focuses on the eigenvalues nearest to the origin. The eigenvalues along the imaginary axis do not appear as a cluster to this interpolation point. Strategy 2 is able to continue converging at higher frequencies as m grows, because the eigenvalues lie on a single path along the imaginary axis. Think of circles of increasingly larger radii that are centered at the origin. As these circles

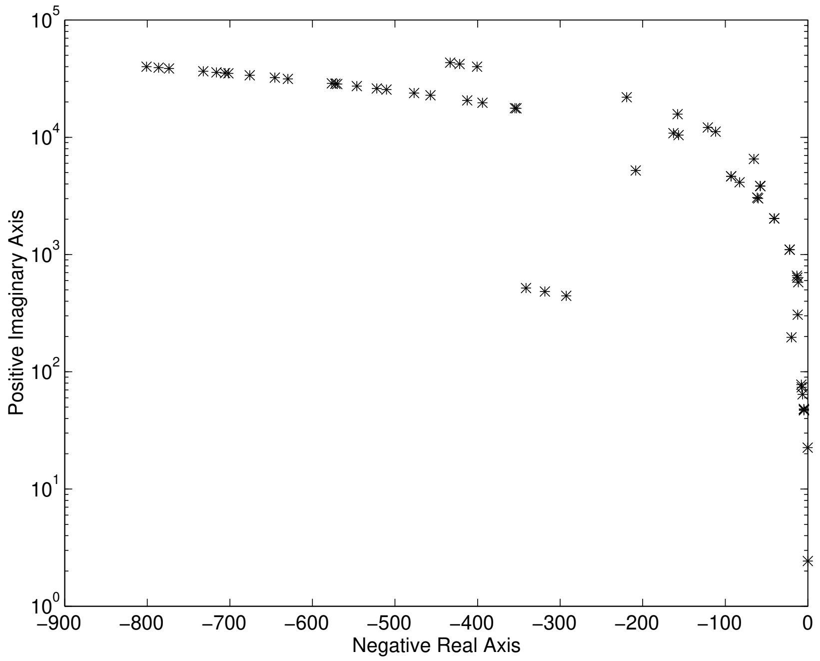

Figure 6.4: Eigenvalue Spectrum of Example 6.3

grow in size, the only additional eigenvalues that they enclose lie at higher frequencies along the imaginary axis. There are no eigenvalues with large real and small imaginary components that would induce stagnation in Strategy 2. Observe the empty space in the lower-left corner of Figure 6.4. Strategies 3 through 5 all utilize at least one interpolation point with a magnitude on the order of one as well. Thus, these latter strategies share the desirable convergence properties of Strategy 2.

The CD player possesses an eigenvalue spectrum that favors low-magnitude interpolation points. However, as Example 6.4 shows, interpolation points near the origin can cause difficulties in other problems.

Example 6.4 *In this example, we once more consider the PEEC problem introduced in Example 5.2. Like Example 6.3, the five strategies at the beginning of this section were each utilized for one-hundred dual rational Arnoldi iterations. The specific interpolation points used in each strategy are listed in Table 6.3. The convergence results with each of these strategies are presented in Table 6.4. Because the frequency range of interest only* *spans an order of magnitude in Figure 5.3, the same single real interpolation point is appropriate for Strategies 1 and 3.*

Unlike Example 6.3, Strategy 2, the single imaginary interpolation point, suffers in this example. To understand this occurrence, consider the portion of the PEEC spectrum of (A, E) that appears in Figure 6.5. Note that the axes in Figure 6.5 are scaled to Hz for easy comparison with the frequency response in Figure 5.3. A cluster of eigenvalues near the origin leads to the stagnation observed in the second column of Table 6.4. Using imaginary interpolation points at $\pm 0.5\iota$, Strategy 2 emphasizes this cluster at the origin instead of the line of eigenvalues running up along the imaginary axis. However, this line of eigenvalues dominates the frequency response; note the correspondence between their locations and the spikes in Figure 5.3. Although weak at times, these eigenvalues are easily captured by the other strategies, particularly numbers four and five. The more complicated Strategies 4 and 5 do show speed ups over a single real point, yet Strategy 1 is not unreasonable given its economical implementation. In fact, Strategy 5 could only be carried out for seventy iterations in this problem due to the memory limitations of the machine executing the algorithm.

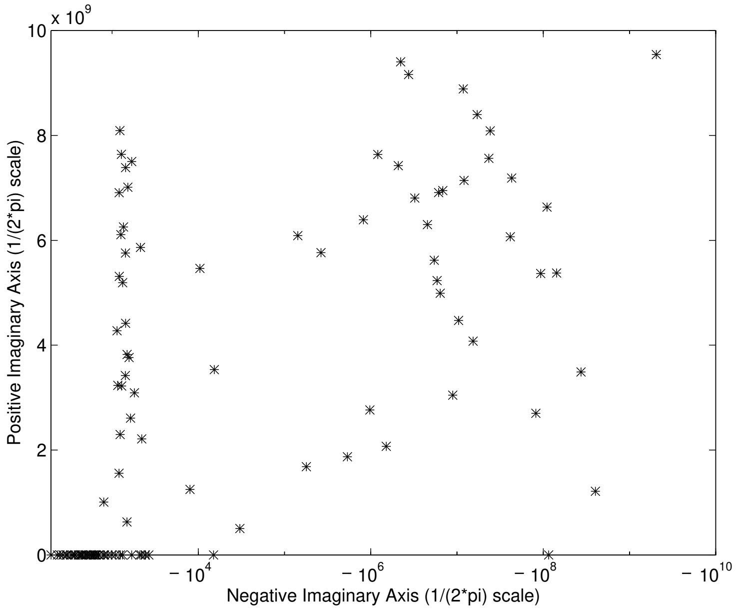

Figure 6.5: Eigenvalue Spectrum of Example 6.4

Table 6.3: Interpolation Point Strategies of Example 6.4

Strategy	Interpolation Point(s)
1	5
2	$\pm 0.5\iota$
3	5
4	$\pm\iota$, $\pm 2\iota$, $\pm 3\iota$, $\pm 4\iota$, $\pm 5\iota$
5	$\pm\iota$, $\pm 3.5\iota$, $\pm 5\iota$, $\pm 3\iota$, $\pm 2\iota$, $\pm 0.5\iota$, $\pm 4.5\iota$, $\pm 2.75\iota$

Table 6.4: Convergence with the Interpolation Point Strategies of Example 6.4

m	Strategy 1	Strategy 2	Strategy 3	Strategy 4	Strategy 5
10	6.0169e-01	6.4804e-01	6.0169e-01	5.7333e-01	6.6406e-01
20	2.2386e-01	8.4414e-01	2.2386e-01	1.4775e-01	2.2477e-01
30	2.3394e-01	2.2597e-01	2.3394e-01	1.3035e-01	1.3782e-01
40	2.5878e-01	2.7452e-01	2.5878e-01	1.4116e-02	2.7480e-02
50	2.6881e-01	2.0472e-01	2.6881e-01	1.6305e-04	7.9555e-04
60	1.0115e-01	1.1185e-01	1.0115e-01	2.7311e-06	2.3797e-05
70	4.4557e-02	1.4684e-01	4.4557e-02	1.7026e-07	8.2285e-08
80	9.9652e-03	1.0727e-01	9.9652e-03	4.7323e-08	
90	3.1338e-04	6.0746e-02	3.1338e-04	1.1786e-07	
100	3.6179e-06	6.8173e-02	3.6179e-06	3.6125e-08	

A single interpolation point is suitable for many applications. The actual location of that optimal point varies though from problem to problem. The placement of a single point may be further clouded by the fact that little is frequently known of a system's response prior to the model reduction. On the other hand, Examples 4.3 and 4.4 suggest that a few systematically placed interpolation points insure fast convergence in a variety of situations. Strategy 3, for example, provides a robust and competitive convergence (relative to any single-point strategy) in all attempted problems.

As for the adaptive strategies of Section 6.3, the results are mixed. Convergence improvements by a factor of two to three can at times be observed with adaptive placement and selection. Yet, these convergence improvements tend to be accompanied by significant increases in cost. The accuracy of the adaptations also tends to be limited by the quality of the error estimates, $\hat{\epsilon}$. Further research is needed in this area. Perhaps the true value of adaptation may only be realized in methods that avoid the sequential implementation of exact DS preconditioners. Alternatives to the exact case are studied in both Chapters 7 and 8.

CHAPTER 7

PARALLEL RATIONAL INTERPOLATION

A significant amount of high-level parallelism appears to exist in rational interpolation. Multiple interpolation points and their corresponding Krylov subspaces are involved. Treating these points concurrently might significantly reduce the time required to construct the reduced-order model. Although possible complications arise in the (bi)orthogonalization and generation of the approximation, an interesting version of the RP algorithm is devised which avoids these difficulties and provides impressive preliminary experimental results.

7.1 Overview

One way to enhance the performance of Krylov-based model reduction is to execute the algorithm on multiple processors. One breaks the algorithm into portions that can be treated in a parallel fashion. Some interaction is typically required between these sub-portions, although communications should be kept to a minimum.

There are at least two types of exploitable parallelism within model reduction via projection. The first is the parallelism that exists to various degrees in the basic matrix operations: matrix-vector products, matrix factorizations, etc. The second type of parallelism arises from the fact that the column spaces of the projection matrices V and Z are composed of multiple Krylov subspaces, which are only differentiated by the choice of $\sigma^{(k)}$. One hopes to concurrently construct the subspaces making up the unions in V and Z. Accordingly, the interpolation points are scattered across the processors. In theory, this second type of parallelism could be combined with that in the basic matrix operations. A strategy with two or more levels of parallelism results. One most likely needs to utilize both of these levels to meet the memory requirements imposed by larger problems. In this section, only the structure of the V and Z column spaces is exploited. The reader is referred to [94] for parallel versions of the more basic matrix operations.

Corresponding to the assignment of distinct interpolation point(s) to each processor, several assumptions are required. First, one must assume that sufficient memory exists to store the sparse factorizations of $(A - \sigma^{(k)}E)$ at all the interpolation points. In a distributed network of processors [94], one must therefore assume that the memory local to each processor can store the sparse factorizations corresponding to its assigned interpolation point(s). Second, and for reasons of scalability [94], one must allow the number of interpolation points K to meet or exceed the number of moments, $2J$, matched about each interpolation point. The interpolation points can be treated in parallel. Matching higher moments (large J) about a given point is a sequential procedure, i.e., the only parallelism available is in the matrix operations. This $K \geq J$ assumption tends to invalidate a parallel version of the RL algorithm of Section 4.1.3. A large K in rational Lanczos leads to a large bandwidth in $\hat{A}$ and $\hat{E}$ (long recurrences) and, hence, eliminates the RL method's edge over the dual rational Arnoldi algorithm. The assumption of a large K also runs contrary to the tendencies throughout the rest of this dissertation. In this proposed parallel approach, many interpolation points are utilized, which each capture the dynamic behavior in their locality. Hence, a highly parallel implementation appears to be suited to the use of imaginary rather than real interpolation points (recall Section 6.2). Given moderate parallelism (only a few processors), combinations of real and imaginary shifts are most likely appropriate (we do not consider this case further in this section).

The bases for the individual subspaces in the unions, i.e., bases for

$$\mathcal{K}_J((A - \sigma^{(k)}E)^{-1}E, (A - \sigma^{(k)}E)^{-1}b) \tag{7.1}$$

and

$$\mathcal{K}_J((A - \sigma^{(k)}E)^{-T}E, (A - \sigma^{(k)}E)^{-T}c) \tag{7.2}$$

can be constructed concurrently. Consider, for now, the case where one interpolation point is assigned per processor. Matrices $V_{J,k}$ and $Z_{J,k}$ can be constructed on the k^{th} processor that yield bases for the subspaces (7.1) and (7.2) evaluated at $\sigma^{(k)}$. Upon completion, the overall projection matrices

$$V = \begin{bmatrix} V_{J,1} & \ldots V_{J,K} \end{bmatrix} \quad \text{and} \quad Z = \begin{bmatrix} Z_{J,1} & \ldots Z_{J,K} \end{bmatrix}$$

are stored across the K processors. Unfortunately, this simple approach is incomplete. The reduced-order model in (2.7) must still be computed from V and Z, requiring the interaction of all K processors. Related, but more limiting, one may place orthogonality conditions on V and Z. Orthogonalization requires each processor to access the entire V and Z matrices. However, broadcasting the dense columns of V and Z to all processors is a significant communication bottleneck.

7.2 A parallel dual rational Arnoldi algorithm

Several parallel variants of a one-sided rational Arnoldi method are developed in [81] for the eigenvalue problem. These techniques are trivially extended to the dual case, and we do not cover the details here. To summarize, the factorizations of $(A - \sigma^{(k)}E)$ at multiple points are scattered among processors. In each step of the algorithm, K new directions in both the column spaces of V and Z are concurrently constructed. To achieve orthogonal bases, one or more of the processors needs to access all of the existing directions in V and Z. But as mentioned above, the amount of communication involved in this orthogonalization is undesirable.

7.3 A parallel rational power algorithm

To avoid orthogonalization and the associated processor interactions, one can turn to the rational power Krylov method, which was proposed in Section 4.1.1. The RP method does not use orthogonalization, but requires numerous interpolation points. Both of these qualities are well-suited to a parallel implementation. In particular, V can be computed without processor interactions, if orthogonalization is avoided.

However, significant processor interaction may still be required to form $\hat{E} = Z^T EV$ and $\hat{A} = Z^T AV$. Consider, for example, that $\hat{a}_m = Z^T Av_m$ must be formed for $m = 1$ to M; yet the columns of Z are scattered across the processors.

Matters would be significantly simplified if every processor automatically knew the entire matrix Z at the start of execution. The matrices $\hat{A}$ and $\hat{E}$ would then be generated in parallel. Each processor would compute the columns $Z^T Av_m$ and $Z^T Ev_m$ of $\hat{A}$ and $\hat{E}$ that correspond to the columns of V that they possessed. No communication would be required to form V, $\hat{A}$ or $\hat{E}$. An automatic knowledge of Z (without communications) is possible if the form of Z is prespecified in a manner known to all of the processors. If Z is prespecified to be a matrix of random elements, for example, each processor can directly access Z by simply knowing the seed value of the random number generator. Similarly, Z contains the first M columns of an identity matrix, each processor could trivially generate Z for itself. Of course, such versions of Z fail to satisfy the desired rational interpolation form in (3.2). Yet moment-matching is two-sided; a V that satisfies (3.1) leads to a reduced-order model which still matches M (rather than $2M$) moments. We can specify Z for ease of computation and V for approximation accuracy. One forfeits some moments by prespecifying the form of Z, but the loss of model-reduction quality in practice does not seem to be as harmful as one might fear. Even when Z is arbitrary, V matches moments and insures that the reduced-order model converges in at most N steps. As Examples 7.1 through 7.3 demonstrate, the convergence based on the one-sided use of V tends to be reasonably competitive with the previous two-sided algorithms. Moreover, the one-sided approach is highly parallel.

Besides avoiding the communication of Z, a parallel RP algorithm must carefully assign interpolation points to the processors. Interpolation point placement is crucial for load-balancing, i.e., making sure that the amount of work performed by each processor is fairly identical. If the interpolation points on two different processors are too close, then these processors compute (nearly) duplicate directions and work is wasted. One of the processors could have been more efficiently utilized elsewhere. On the other hand, if an insufficient number of processors are assigned to a frequency range where large modeling error exists, then this error only gradually declines, many iterations are required on a single processor, and the advantages of parallelism are lost. Beyond load-balancing, the placement of multiple interpolation points is fundamental to the RP method. Recall from Section 4.1.1 that the RP algorithm relies on frequent changes of the interpolation points (small J_k) to introduce sufficient information into V. Luckily for the RP approach, the use of many well-placed interpolation points is consistent with a load-balanced parallel implementation.

To effectively address point placement, we suggest the combination of several techniques proposed in the previous three chapters:

1. *An Error Estimate via Complementary Models (Section 5.1).* Determining appropriate locations for the interpolation points requires a sense of the dynamics of the original system. As the model reduction proceeds, the error estimate (5.1) can be utilized to obtain a sense of important behavior, which is absent from the reduced-order model. The comparison of complementary models is particularly appropriate for parallel algorithms; one can construct the two different viewpoints $\hat{\mathbf{h}}(s)$ and $\hat{\mathbf{h}}_{\perp}(s)$ concurrently.

2. *Adaptive Point Placement (Section 6.3).* It is difficult, if not impossible, to quantify the convergence behavior of the reduced-order model a priori. Therefore, to insure that each processor is contributing new information at every step, old interpolation points must be discarded where convergence has occurred, while new interpolation points must be introduced where the modeling error remains large. These decisions are made according to the error estimate.

3. *Dependency Postprocessing (Section 4.1.1).* Even with adaptive point placement based on error estimates, the RP algorithm may still occasionally compute (nearly) redundant information. Postprocessing via a singular value decomposition is therefore employed to remove the any redundancies in the projection matrices.

Algorithm 7.1 combines these techniques. This algorithm (a) computes directions in V at K points, (b) updates the set of interpolation points according to an updated error estimate, and (c) returns to the first step if needed. The computation of K new directions, (S7.1.1) through (S7.1.4), may be executed in parallel. These steps include the matrix factorizations in (S7.1.2) and the reduced-order model updates in (S7.1.4). Note that two complementary reduced-order models are actually computed in (S7.1.4).

This parallel RP algorithm completely replaces its set of interpolation points at the beginning of every iteration of the outer l loop (although in practice one might want to use a given interpolation point for a few iterations to reduce the required work). As such, only one moment (the value of the frequency response) is matched about any given interpolation point. At the end of each outer iteration, an error estimate is computed based on the comparison of two distinct reduced-order models. Mainly, the difference $\hat{\mathbf{h}}(s)-\hat{\mathbf{h}}_{\perp}(s)$ is computed at many points over the frequency range ω_{min} to ω_{max}. A scaled version of this difference is then used as a (discrete) probability distribution function for the random placement of the next set of K interpolation points. The probability that a new interpolation point is located at some frequency is proportional to the degree of modeling error estimated at that frequency.

Those operations outside of the k dependent loops require either communications between the processors, (S7.1.5) and (S7.1.8), or sequential operations, (S7.1.6). The communications require messages of length $O(M)$, in practice, while the sequential (S7.1.6) involves $O(M^3)$ operations. All communications and sequential operations involving order-N quantities are avoided. The communications and sequential operations occur L times, a value that is expected to be small.

The Algorithm 7.1 is designed for clarity and several details are intentionally overlooked. The amount of effort associated with $\hat{A}$ and $\hat{E}$ is not quite as much as suggested by (S7.1.4) and (S7.1.5). One does not need to compute/send the entire matrices $\hat{A}$ and $\hat{E}$ in each outer l iteration. The leading minors of the $\hat{A}$ and $\hat{E}$ matrices remain the same from one iteration to the next. Only the K bottom rows and K rightmost columns need

Algorithm 7.1 Rational Krylov (Parallel RP Version)

Initialize: $\hat{\epsilon} = 1$ for $\omega_{min} < w < \omega_{max}$;

For $l = 1$ to L,

  For $k = 1$ to $2K$,

    (S7.1.1) choose $\sigma^{(k)}$ randomly between ω_{min} and ω_{max} according to $\hat{\epsilon}$ distribution;

    (S7.1.2) $\hat{v}_{l,k} = (A - \sigma^{(k)}E)^{-1}b$;

    (S7.1.3) $v_{l,k} = \hat{v}_{l,k} / \; \|\hat{v}_{l,k}\|_2$;

    (S7.1.4) If $k \le K$,

      For $j = 1$ to l,

        $\hat{a}_{(j-1)K+k} = Z_{lK}^T A v_{j,k}$ and $\hat{e}_{(j-1)K+k} = Z_{lK}^T E v_{j,k}$;

      end

    else,

      For $j = 1$ to l,

        $\hat{a}_{\perp_{(j-2)K+k}} = Z_{\perp_{lK}}^T A v_{j,k-K}$ and $\hat{e}_{\perp_{(j-2)K+k}} = Z_{\perp_{lK}}^T E v_{j,k-K}$;

      end

    end

  end

  (S7.1.5) send $\hat{A}$, $\hat{E}$ and $\hat{A}_\perp$, $\hat{E}_\perp$ to all processors;

  (S7.1.6) postprocess approximations for rank-deficiencies

  For $k = 1$ to $2K$,

    (S7.1.7) compute $\hat{\epsilon} = \hat{\mathbf{h}} - \hat{\mathbf{h}}_\perp$ over $(\omega_{min} + \frac{k-1}{2K}\omega_{max})$ to $(\omega_{min} + \frac{k}{2K}\omega_{max})$;

  end

  (S7.1.8) send $\hat{\epsilon}$ to all processors;

end

---

to be treated in a given l outer iteration. The output and input vectors, V^Tc and Z^Tb, can be handled similarly. The effort involved in these vectors is minor, when compared to that in forming and communicating $\hat{A}$ and $\hat{E}$.

There are several other details that can be implemented in the parallel RP algorithm. First, one might edit $\hat{\epsilon}$ in (S7.1.1) after each interpolation point is chosen. As a new interpolation point is placed, set $\hat{\epsilon}$ to zero in the near proximity of this point. With this small trick, (nearly) duplicate interpolation points can be avoided. Implementing this optional $\hat{\epsilon}$ modification does require (S7.1.1) to be executed sequentially (this is not a real concern, because this step is so simple). Second, the outer l iteration would most likely be terminated and new interpolation points would cease to be chosen when $\hat{\epsilon}$ becomes sufficiently small everywhere. Third, the treatment of complex quantities in Algorithm 7.1 is rather nebulous. In practice, V is augmented with both the directions real$\{(A - \sigma^{(k)}E)^{-1}b\}$ and imag$\{(A - \sigma^{(k)}E)^{-1}b\}$ so as to incorporate both $\sigma^{(k)}$ and its complex conjugate into the reduced-order model.

The reliability of the proposed parallel methods could be questioned at this point. After all, terms such as random and nonorthogonal do not exactly inspire confidence in the approach. A rigorous understanding of the method's convergence, in practice, is not yet claimed. We rely instead on the generality of Theorem 3.1 and impressive initial experimental results. Several tests of a sequential (quasi-parallel) version of the parallel RP algorithm are presented in Examples 7.1 through 7.3. The MATLAB code used in these tests may be found in Appendix B. We stress that the reduced-order models generated with this quasi-parallel algorithm should exactly equal those formed with a true 16-processor machine. At this point, our goal is obviously not an indepth analysis of the approach's efficiency; although an execution time is desired that is in the neighborhood of L sparse linear system solves. Rather, we simply emphasize the potential of the approach.

In Examples 7.1 through 7.3, a few details should not be overlooked. First, the random generator seed was set to zero at the beginning of each test. Second, the plots associated with each case utilize a dotted line to denote the true frequency response, a dashed line to denote the frequency response of the reduced-order model, and a solid line to denote the error estimate $\hat{\epsilon}$. Third, it is expected that the reduced-order models will be augmented by an additional 16 states (corresponding to 8 interpolation points and their complex conjugates) after each outer MATLAB iteration. The actual model size, denoted by m, does not grow quite as quickly, however, because dependent directions (corresponding to singular values with relatives sizes under 10^{-8}) are discarded by the SVD postprocessing of Section 4.1.1.

Example 7.1 *In this problem, we reconsider the packaging interconnect of Example 4.5. A quasi-parallel RP algorithm was executed for only $L = 2$ outer iterations. The results after each iteration are shown in Figures 7.1 and 7.2. Both the quality of the reduced-order models and the error estimates are high. The accuracy of this 25^{th} order model is is competitive with the results of the sequential dual rational Arnoldi algorithm in Figure 4.5. Although the dual RA approach matches twice as many moments, dual RA does not aggressively utilize multiple interpolation points.*

Example 7.2 *In this problem, we reconsider the CD player subsystem of Example 6.3. The results after each of the first $L = 4$ iterations are displayed in Figures 7.3 through 7.6. The model accuracy is again comparable to that of the dual RA Algorithm 6.2. The error at high frequencies is due primarily to the measure ($\mathcal{H}_\infty$ norm) of the modeling error and not to a shortcoming of the parallel RP method.*

Example 7.3 *In this problem, we reconsider the PEEC model of Example 5.2. The results after each of the first $L = 4$ iterations are displayed in Figures 7.7 through 7.10. The reduced-order model accuracy with the parallel RP algorithm is actually superior to the previously seen dual RA results. The large number of spikes in this problem's frequency response are more rapidly found (in terms of model size m) with a large number of interpolation points. It is interesting to note that the number of discarded projection directions becomes large as the degree of convergence becomes significant (the model size between $l = 3$ and $l = 4$ changes only by two). The difference between the original and* *reduced-order models is no longer large enough to readily support multiple new directions (additional interpolation points) beyond $l = 3$.*

A parallel model-reduction algorithm involving one random projection matrix Z and one nonorthogonal projection matrix V was proposed. Using a random Z reduces the number of moments that can be matched by the reduced-order model. Yet a random Z allows for the parallel construction of the reduced-order model with minimal communications. Also, as long as V satisfies (3.1), M moments are still matched. Error estimates and adaptive point placement are aimed at acquiring an efficient, balanced convergence. By avoiding Gram-Schmidt recurrences during the construction of this V, parallelism is further increased. Although the lack of orthogonality in V can lead to dependences, these problems can apparently be overcome by the use of numerous interpolation points and SVD postprocessing of the reduced-order model. Although further study of this SVD postprocessing is required (see Chapter 10), the results of Examples 7.1 through 7.3 are noteworthy.

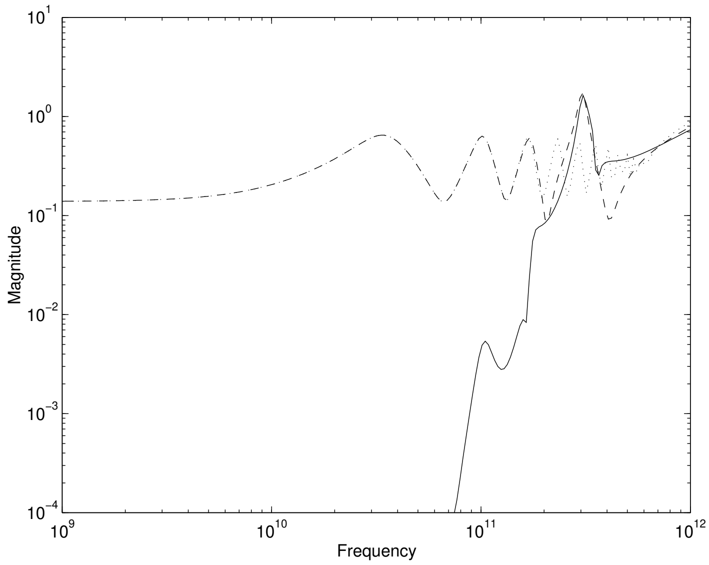

Figure 7.1: Approximate Frequency Response of Example 7.1 when $m = 12$

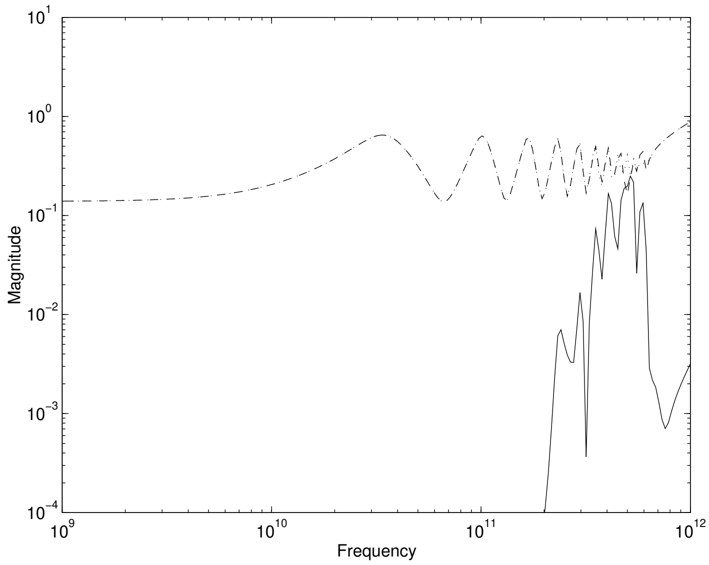

Figure 7.2: Approximate Frequency Response of Example 7.1 when $m = 25$

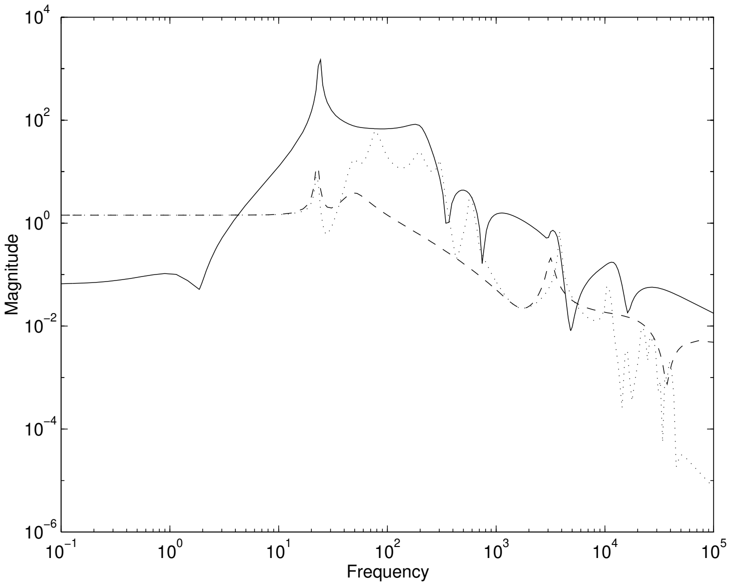

Figure 7.3: Approximate Frequency Response of Example 7.2 when $m = 14$

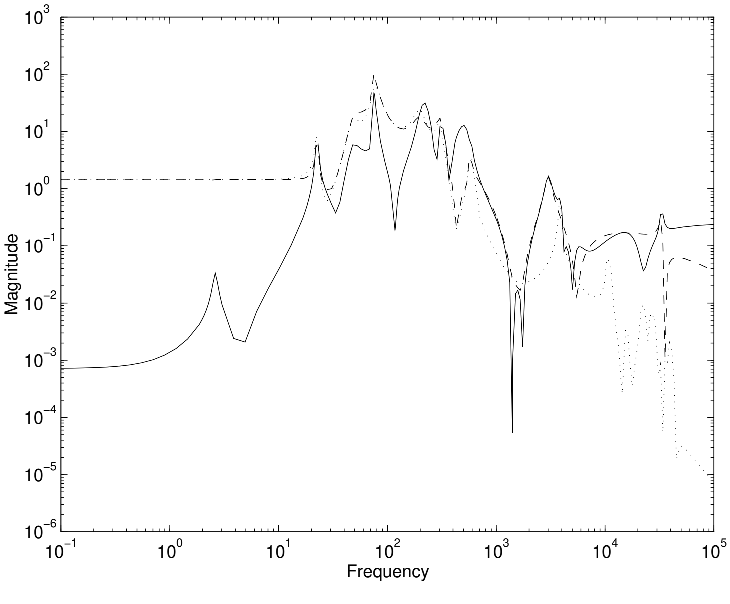

Figure 7.4: Approximate Frequency Response of Example 7.2 when $m = 27$

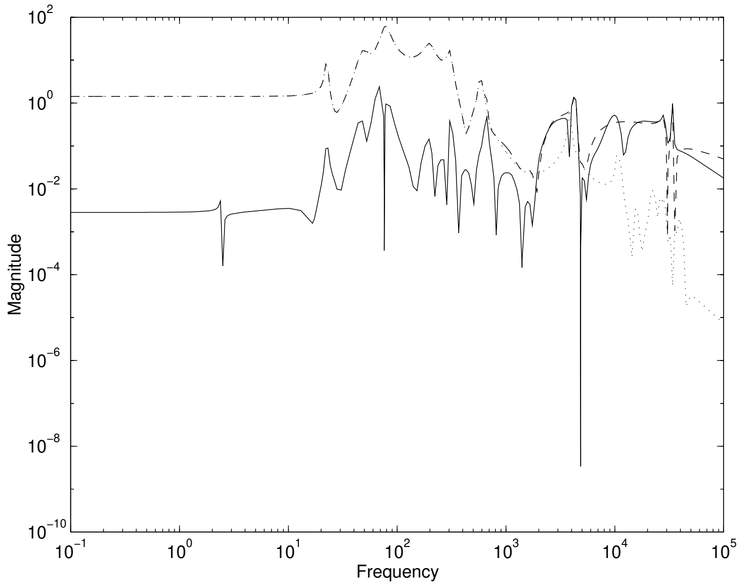

Figure 7.5: Approximate Frequency Response of Example 7.2 when $m = 40$

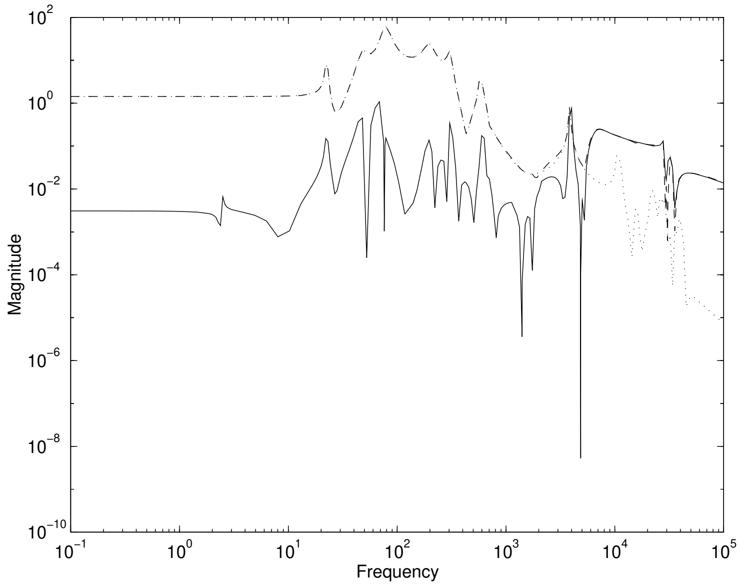

Figure 7.6: Approximate Frequency Response of Example 7.2 when $m = 48$

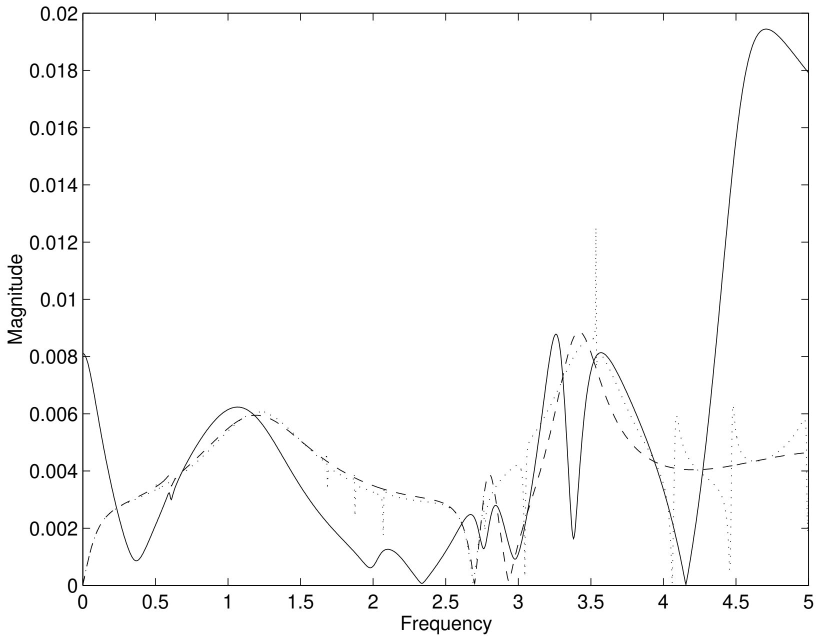

Figure 7.7: Approximate Frequency Response of Example 7.3 when $m = 15$

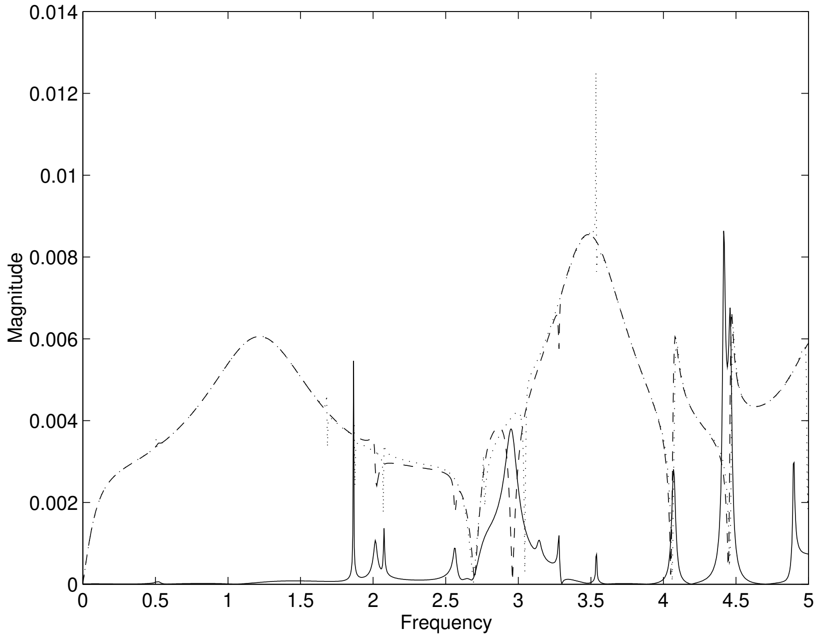

Figure 7.8: Approximate Frequency Response of Example 7.3 when $m = 31$

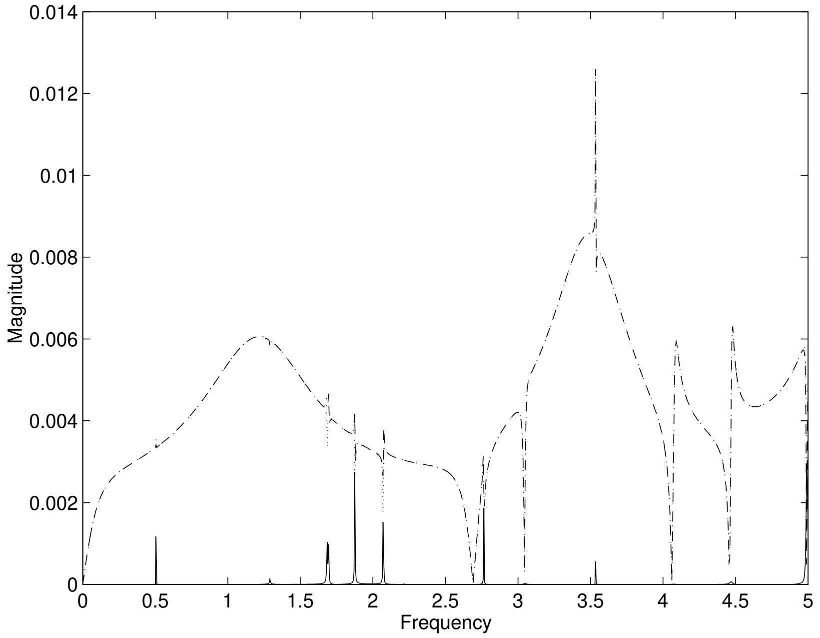

Figure 7.9: Approximate Frequency Response of Example 7.3 when $m = 42$

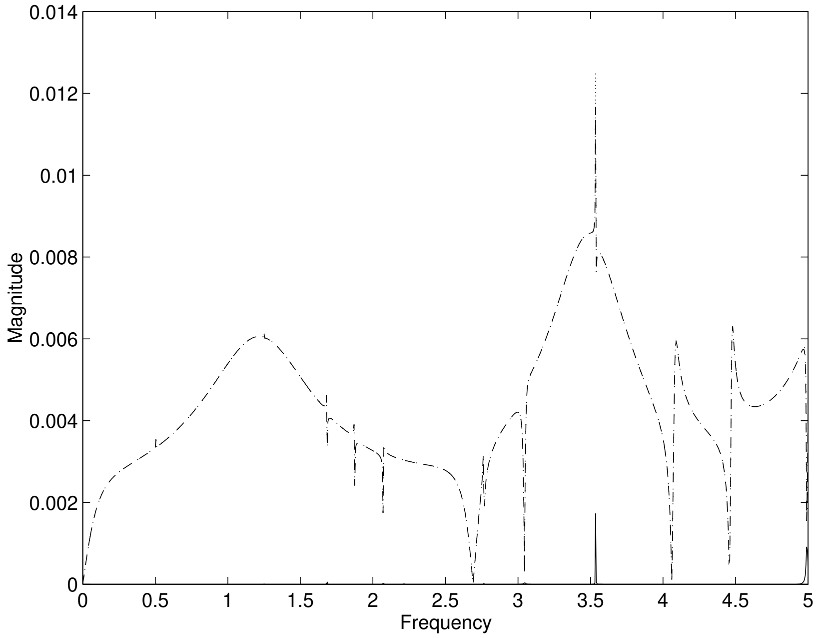

Figure 7.10: Approximate Frequency Response of Example 7.3 when $m = 44$

CHAPTER 8

APPROXIMATE RATIONAL INTERPOLATION

The power of the model-reduction techniques of the previous sections arises primarily from the construction of bases for the exact DS-preconditioned Krylov subspaces in (3.1) and (3.2). However, it is doubtful that these bases can be generated or even stored as the size and/or complexity of the original system grows. For example, direct sparse factorizations of a matrix pencil arising from a three-dimensional discretization are frequently impractical. This chapter considers approaches for reducing the computational effort required in rational interpolation. Both explicit and implicit approximations to $(A - \sigma E)^{-1}$ are incorporated into the projection subspaces. Some of the exact relations of Chapters 3 and 4 are lost, but reasonable results are still possible.

The following efforts represent some of the first attempts to relax the requirements on DS preconditioners in Krylov algorithms in order to reduce work. This chapter is a starting point rather than a final solution for reducing the computational effort in model reduction.

8.1 Conversions to Approximate Solves

The construction of an exact rational interpolant requires the exact DS preconditioners present in the subspaces of (3.1) and (3.2). By Lemmas 2.1 and 2.2, these subspaces take the form

$$\operatorname{colsp}\{V\} = \bigcup_{k=1}^{K} \mathcal{K}_{J_k}((A - \sigma^{(k)}E)^{-1}(A - sE), (A - \sigma^{(k)}E)^{-1}b) \tag{8.1}$$

$$\operatorname{colsp}\{Z\} = \bigcup_{k=1}^{K} \mathcal{K}_{J_k}((A - \sigma^{(k)}E)^{-T}(A - sE)^{T}, (A - \sigma^{(k)}E)^{-T}c), \tag{8.2}$$

where $(A - \sigma^{(k)}E)^{-1}$ is an exact DS preconditioner and the choice of s is superfluous. Emphasizing the iterative solution of the dual systems of equations (1.2), recall that Section 3.2 suggests the need to find only acceptable approximate solutions $V\hat{\mathbf{x}}_b$ and $Z\hat{\mathbf{z}}_c$ across the frequency range of interest. In adopting this view and abandoning strict moment-matching conditions, inexact DS preconditioners become appropriate. We replace $(A - \sigma^{(k)}E)^{-1}$ in (8.1) and (8.2) with approximations.

An adaptation of the RK algorithm utilizing approximations to $(A - \sigma^{(k)}E)^{-1}$ is presented in Algorithm 8.1. There are only two differences between Algorithms 4.1 and 8.1. The matrix $(A - \sigma_m E)^{-1}$ was replaced with $\Phi_m \approx (A - \sigma_m E)^{-1}$ in (S8.1.2) and E was replaced with $(A - \zeta_m E)$ in (S8.1.4). The scalar ζ_m corresponds to fixing a value for s in (8.1) and (8.2). It was noted in Section 3.2, that the choice of s in (8.1) and (8.2) is not relevant for the exact DS preconditioner case due to the shift invariance property of Krylov subspaces. Thus, in the previously assumed exact cases, it made computational sense to simply replace $(A - sE)$ with E (think of allowing s to grow larger).

Algorithm 8.1 Rational Krylov (Approximate General Version)

Initialize: $q_1 = (\gamma_1^q)^{-1}b$ and $w_1 = (\beta_1^w)^{-1}c^T$;

For $m = 1$ to M,

(S8.1.1) Input: σ_m, the interpolation point for m^{th} iteration;

(S8.1.2) $\tilde{v}_m = \Phi_m q_{p_m+1}$ and $\tilde{z}_m = \Phi_m^T w_{p_m+1}$;

(S8.1.3) $\gamma_m^v v_m = \tilde{v}_m - V_{m-1}\bar{v}_m$ and $\beta_m^z z_m = \tilde{z}_m - Z_{m-1}\bar{z}_m$;

(S8.1.4) $\tilde{q}_{m+1} = (A - \zeta_m E)v_m$ and $\tilde{w}_{m+1} = (A - \zeta_m E)^T z_m$;

(S8.1.5) $\gamma_{m+1}^q q_{m+1} = \tilde{q}_{m+1} - Q_m\bar{q}_{m+1}$ and $\beta_{m+1}^w w_{m+1} = \tilde{w}_{m+1} - W_m\bar{w}_{m+1}$;

end

The generated V and Z of the approximate RK algorithm no longer form bases for unions of Krylov subspaces (Lemmas 2.1 and 4.1 do not carry over to the inexact DS preconditioning case), nor do these projection matrices lead to rational interpolants. However, the reduced-order model formed from (2.7) with approximately DS-preconditioned

V and Z does satisfy the Petrov-Galerkin constraints (meeting the Petrov-Galerkin constraints is not restricted to moment-matching) and Theorem 5.1 still holds. As long as reasonable approximations $V\hat{\mathbf{x}}_b$ and $Z\hat{\mathbf{x}}_c$ to $\mathbf{x}_b$ and $\mathbf{x}_c$ are acquired, a good reduced-order model is achievable. It is interesting to note that even the exactly DS-preconditioned V and Z do not necessarily lead to optimal approximations to $\mathbf{x}_b$ and $\mathbf{x}_c$ at all s. Rational interpolation leads to exact DS preconditioners, which are only optimally suited for a few discrete points, i.e., the interpolation points. For frequencies away from the interpolation points, it is uncertain as to whether $(A - \sigma^{(k)}E)^{-1}$ is necessarily a better DS preconditioner than some $P_k \approx (A - \sigma^{(k)}E)^{-1}$.

Two new parameters, Φ_m and ζ_m, appear in Algorithm 8.1 and must be specified. We denote Φ_m as the DS preconditioning operator of the m^{th} iteration. The operator Φ_m approximates the action of $(A - \sigma_m E)^{-1}$ on a vector. If Φ_m is a matrix, which was always assumed in the past, then it is a member of the set $\{P_1, P_2, \ldots P_K\}$, where P_k is an approximation to $(A - \sigma^{(k)}E)^{-1}$. This choice is consistent with the previous notation regarding a DS preconditioner. Numerous possibilities exist for finding a fixed DS preconditioner P_k that approximates $(A - \sigma^{(k)}E)^{-1}$ [8]. Typically, one constructs a sparse matrix, which can be utilized to approximate the action of $(A - \sigma^{(k)}E)^{-1}$. In the incomplete LU approach, for example, an approximate, sparse LU factorization of $(A - \sigma^{(k)}E)$ is computed. Elements appearing during the factorization that correspond to a certain level of fill or possess sizes that are under a certain tolerance are dropped [95]. A second technique is the approximate inverse approach. One constructs a P_k with some sparsity pattern, so that $P_k(A-\sigma^{(k)}E)-I$ is minimized with respect to some norm, e.g., the Frobenius norm [96].

Alternatively, and more generally, one can think of Φ_m as an operation that takes in the vectors q_{p_m+1}, w_{p_m+1} and outputs the vectors $\tilde{v}_m$, $\tilde{z}_m$. Hence, Φ_m can represent an iterative system solver that computes approximate solutions to the equations*

$$(A - \sigma_m E)\underline{\tilde{v}}_m = q_{p_m+1} \quad \text{and} \quad (A - \sigma_m E)^T \underline{\tilde{z}}_m = w_{p_m+1}. \tag{8.3}$$

*Note that the exact (ideal) solutions in (8.3) are underlined. The vector that is actually computed (the approximation) is indicated in standard fashion without underlining.

When an iterative solver is utilized, Φ_m represents a nonlinear operation which is no longer associated with a fixed matrix P_k. The use of iterative solvers to implicitly perform DS preconditioning is common in the linear solver literature [70, 71, 97]. Methods of this type are known as inner-outer iterations. The outer iteration constructs a search subspace for the solution of the original problem. In our case, the outer iteration constructs the projection matrices V and Z. During each outer step, an entire loop of inner iterations is executed (not necessarily for the same number of steps each time) to generate the DS preconditioner for the current outer step. In our case, the inner iteration consists of iteratively constructing approximate solutions in (S8.1.2) to (8.3). The operator Φ_m is implicitly defined by the inner iteration. Many different iterations are possible when constructing the approximations $\tilde{v}_m$ and $\tilde{z}_m$ to $(A-\sigma_m E)^{-1}q_{p_m+1}$ and $(A-\sigma_m E)^{-T}w_{p_m+1}$. These inner iterative solvers are generally Krylov methods themselves. The approximate solutions $\tilde{v}_m$ and $\tilde{z}_m$ are typically chosen from the inner Krylov subspaces, according to Petrov-Galerkin or minimal residual constraints. Examples of iterative approaches satisfying these two respective constraints are the biconjugate gradient (BiCG) and quasi-minimal residual (QMR) methods [6, 8]. These particular methods are mentioned because they are two-sided; each can simultaneously generate approximation solutions to dual systems of equations involving $(A - \sigma^{(k)}E)$ and $(A - \sigma^{(k)}E)^T$. A noteworthy one-sided approach is the generalized minimal residual (GMRES) method [8, 84].

The other new parameter of interest in the approximate RK algorithm is ζ_m. If (and only if) Φ_m is an inexact DS preconditioner, then the specific choice for ζ_m in (S8.1.4) contributes to the specification of the V and Z column spaces. In the exact case, $(A - \zeta_m E)$ was replaced with E for convenience. We denote this exact case as $\zeta_m = \infty$, although one would certainly not compute $(A - \zeta_m E)$ as such. In the inexact case, it is possible to tune ζ_m for improved results.

One possible choice for ζ_m is the interpolation point σ_m. The motivation for this selection is that the matching of moments at σ_m is closely connected to solving the dual equations (1.2) at $s = \sigma_m$. The traditional subspaces involved in solving the system of equations (1.2) with s fixed at σ_m is (2.17) with $\zeta_m = \sigma_m$. A second motivation for this

ζ_m choice follows from the mapping of the eigenvalues of (A, E) to the eigenvalues of $\Phi_m(A - \zeta_m E)$. A mapping for the case when Φ_m is exact and $\zeta_m = \infty$ was presented in Figure 6.1. A mapping is sought that makes the desired eigenvalues stand out in the spectrum of the DS-preconditioned matrix. In Figure 6.1, the properties of the exact DS preconditioner drove the desired poles to the outer edge of the spectrum. An example of a mapping for the $\zeta_m = \sigma_m$ case is displayed in Figure 8.1. This mapping has been studied to a great extent in the eigenvalue literature for a technique known as Davidson's method [19, 98]. It transforms any eigenvalue of (A, E) near σ to a position in the spectrum of $\Phi_m(A - \sigma_m E)$ that is close to the origin. This behavior follows from the fact that if λ is an eigenvalue of (A, E), then zero is an eigenvalue of $\Phi_m(A - \lambda E)$ for any matrix Φ_m. On the other hand, because Φ_m is an approximation to $(A - \sigma_m E)$, it is hoped that the other eigenvalues of $\Phi_m(A - \sigma_m E)$ are close to 1. If this mapping occurs, the desired eigenvalues of (A, E) near σ are mapped towards the origin and are well separated from a cluster at 1. Unfortunately, this argument and the choice $\zeta_m = \sigma_m$ breaks down if Φ_m becomes too good of an approximation to $(A - \sigma_m E)^{-1}$. In this case, $\Phi_m(A - \sigma_m E)$ becomes the identity and all eigenvalues are mapped on top of each other at 1. Furthermore, multiplication of old directions in V and Z by the identity contributes no new information in subsequently computed directions.

As later experiments in Examples 8.3 and 8.4 indicate, the choice of ζ can lead to significant, but oftentimes unpredictable differences in the convergence of the reduced-order model. In practice, ζ_m can be tuned between ∞ and σ_m by using available information on the DS preconditioner quality and/or the convergence behavior of previous solves. Alternatively, more sophisticated approaches for implementing the approximate solvers are discussed in Section 8.3, which de-emphasizes ζ_m.

8.2 Approximate Solve Algorithms

Although the choices for Φ_m and ζ_m are not trivial, they are straightforward given the above discussions to adapt the algorithms of Chapter 4 for approximate system solves.

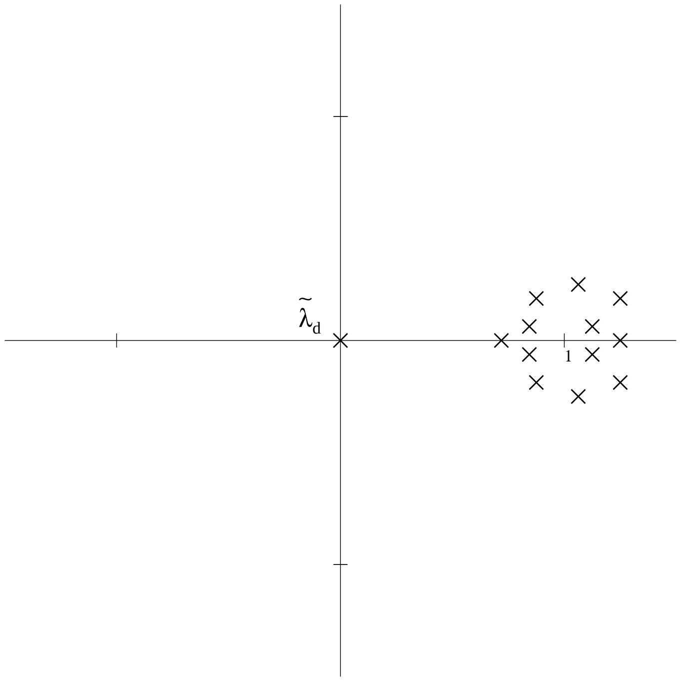

Figure 8.1: An Eigenvalue Mapping for $\zeta_m = \sigma_m$

All that is required is to repeat the modifications seen in the approximate RK algorithm. An approximate dual RA algorithm is presented in Algorithm 8.2.

As long as orthogonality is (nearly) maintained in V and Z, convergence in at most N steps is still guaranteed in the approximate dual RA version. Of course, effective DS preconditioners should be utilized to obtain an effective model of size $M \ll N$.

8.2.1 Approximate rational Lanczos

More than other RK variants seen in Chapter 4, the rational Lanczos algorithm relies on the properties of the exact DS preconditioner. This reliance made it an efficient algorithm in Section 4.1.4, but it also makes the approach a questionable one for inexact DS preconditioning. The RL algorithm relied on exact DS preconditioning in Theorem 4.3 to achieve banded $\hat{A}$ and $\hat{E}$ matrices. If approximations to $(A - \sigma^{(k)}E)^{-1}$ are utilized in (S4.5.6) of the RL algorithm, the reduced-order systems matrices become dense. Either these now nonzero off-diagonal terms must be computed or an error is incurred. Computing all of the elements in $\hat{A}$ or $\hat{E}$ corresponds to full-length biorthogonality recursions

Algorithm 8.2 Rational Krylov (Approximate Dual RA Version)

Initialize: $m = 0$

For $j = 1$ to J,

  For $k = 1$ to K,

    (S8.2.1) If $j = 1$,

        $\tilde{v}_m = \Phi_m b$ and $\tilde{z}_m = \Phi_m^T c$;

      else

        $\tilde{v}_m = \Phi_m (A - \zeta_m E) v_{m-K}$ and $\tilde{z}_m = \Phi_m^T (A - \zeta_m E)^T z_{m-K}$;

      end

    (S8.2.2) $\hat{v}_m = \tilde{v}_m - V_{m-1} V_{m-1}^T \tilde{v}_m$ and $\hat{z}_m = \tilde{z}_m - Z_{m-1} Z_{m-1}^T \tilde{z}_m$;

    (S8.2.3) $v_m = \hat{v}_m / \|\hat{v}_m\|$ and $z_m = \hat{z}_m / \|\hat{z}_m\|$;

    (S8.2.4) $m = m + 1$;

  end

end

(rather than length $K+2$ recursions). Thus, this approach for dealing with approximate DS preconditioning increases the cost of an approximate RL algorithm to levels comparable with the dual RA approach. However, the approximate dual RA algorithm is numerically more reliable; an $O(M^2 N)$ version of the RL algorithm is of little value.

The only other apparent option to dealing with an approximate RL algorithm is to simply ignore the error between Φ_m and $(A - \sigma_m E)^{-1}$. In this case, simply edit (S4.5.6) of the RL algorithm and nothing else. The approximate RL method is presented in Algorithm 8.3. Banded $\hat{A}$ and $\hat{E}$ are still assumed in this algorithm (they are still formed by using only the β terms in the length $K+2$ recursions). The error in this theoretically unsupported approach is characterized by Theorem 8.1.

Theorem 8.1 *The output residual expression for the approximate RL algorithm is*

$$\mathbf{r}_c(s) = \beta_{M+1,M+1} w_{M+1} i_m^T (s\hat{E} - \hat{A})^{-T} \hat{c}(\sigma_M - s) - \tilde{R}(s\hat{E} - \hat{A})^{-T} \hat{c}, \tag{8.4}$$

Algorithm 8.3 Rational Krylov (Approximate Banded RL Version)

Initialize: $\tilde{w}_1 = c$ and $\tilde{v}_1 = \Phi_1 b$ and $m = 1$

For $j = 1$ to J,

  For $k = 1$ to K,

    (S8.3.1) if $m > 1$, $\tilde{v}_m = \Phi_m E v_{m-1}$; end

    (S8.3.2) $\hat{v}_m = \tilde{v}_m - \sum_{l=\max(1,m-K-1)}^{m-1} v_l \gamma_{m,l}$ where $\gamma_{m,l} = w_l^T \tilde{v}_m$;

    (S8.3.3) $\hat{w}_m = \tilde{w}_m - \sum_{l=\max(1,m-K-1)}^{m-1} w_l \beta_{m,l}$ where $\beta_{m,l} = v_l^T \tilde{w}_m$;

    (S8.3.4) $v_m = \hat{v}_m / \gamma_{m,m}$ where $\gamma_{m,m} = \sqrt{|\hat{w}_m^T \hat{v}_m|}$;

    (S8.3.5) $w_m = \hat{w}_m / \beta_{m,m}$ where $\beta_{m,m} = \text{sign}(\hat{w}_m^T \hat{v}_m)\gamma_{m,m}$;

    (S8.3.6) $\tilde{w}_{m+1} = E^T \Phi_m^T w_m$;

    (S8.3.7) $m = m + 1$;

  end

end

---

where the m^{th} column of $\tilde{R}$ is

$$\tilde{r}_m = w_m - (A - \sigma_m E)^T (\Phi_m^T w_m), \tag{8.5}$$

the residual associated with the approximate system solve in (S8.3.6) of the m^{th} iteration.

Proof: Due to residual definition (8.5), the approximate system solution in the m^{th} iteration can be written as

$$z_m \equiv \Phi_m^T w_m = (A - \sigma_m E)^{-T} (w_m - \tilde{r}_m).$$

Multiplying this expression on the left by $(A - \sigma_m E)^T$, i.e.,

$$(A - \sigma_m E)^T z_m = (w_m - \tilde{r}_m),$$

and a shift by $sE^T z_m$ yields the equivalent expressions,

$$\begin{aligned} (A - sE)^T z_m &= (w_m - \tilde{r}_m) + (\sigma_m - s) E^T z_m \\ &= (w_m - \tilde{r}_m) + (\sigma_m - s) W_{m+1} \begin{bmatrix} \bar{w}_{m+1} \\ \beta_{m+1}^w \end{bmatrix}. \end{aligned} \tag{8.6}$$

The expression in (8.6) involves the relations in (S8.3.3) and (S8.3.6). Placing together the expressions (8.6) for all M iterations produces the matrix equality

$$(A - sE)^T Z = -\tilde{R} + W + W_{M+1} \begin{bmatrix} \bar{w}_2 & \ddots & & \\ \beta_{2,2} & & \bar{w}_{M+1} & \\ & \ddots & & \\ & & \beta_{M+1,M+1} & \end{bmatrix} \begin{bmatrix} (\sigma_1 - s) & & \\ & \ddots & \\ & & (\sigma_m - s) \end{bmatrix},$$

which due to the definitions of $\hat{E}$ in (4.21) and $\hat{A}$ in (4.18), is the expression

$$(A - sE)^T Z = W(\hat{A} - s\hat{E}) + (\sigma_m - s)\beta_{M+1,M+1} w_{M+1} i_M^T - \tilde{R}. \tag{8.7}$$

The equality (8.7) can now be substituted into (2.13) to acquire the output residual expression,

$$\begin{aligned} \mathbf{r}_c(s) &= c + \{W(\hat{A} - s\hat{E}) + (\sigma_m - s)\beta_{M+1,M+1} w_{M+1} i_M^T - \tilde{R}\}\hat{\mathbf{x}}_c \\ &= -\tilde{R}\hat{\mathbf{x}}_c - W\hat{c} + c + \beta_{M+1,M+1} w_{M+1} i_M^T (s\hat{E} - \hat{A})^{-T}\hat{c}(\sigma_m - s). \end{aligned} \tag{8.8}$$

The desired expression (8.4) follows trivially given (4.22), the definition of $\hat{c}$ in the RL algorithm. ■

In the exact case, the output residual resulting from the RL algorithm is a scaled version of w_{M+1}. This vector w_{M+1} stays biorthogonal with at least the recent directions of V. Moreover, the scaling of w_{m+1} drops to zero in regions around the interpolation point. When approximations to $(A - \sigma^{(k)}E)^{-1}$ are employed, the output residual is corrupted by the error in Φ_m. The residual $\tilde{r}_m$, associated with the approximation for the vector $(A - \sigma^{(k)}E)^{-T} w_m$, appears in $\mathbf{r}_c$. Several things should be noted concerning this corruption:

1. The corruption of $\mathbf{r}_c(s)$ is proportional to the error in Φ_m.

2. Unlike the exact case, the residual $\mathbf{r}_c$ in approximate RL is not generally forced to zero in the regions about the interpolation point.

3. Errors in the computation of $(A - \sigma_m E)^{-1}$ in the m^{th} iteration continue to appear in the reduced-order models of later iterations (the entire matrix $\tilde{R}$ appears in (8.4)).

4. The total corruption behaves as the sum (rather than product) of previous errors in the computation of $(A - \sigma_m E)^{-1}$, because the matrix $\tilde{R}$ appears in a matrix-vector product in (8.4). This fact is good news for exactly DS-preconditioned versions of the RL algorithm that are implemented in finite precision. Machine-level precision errors in the linear system solvers are not blown up in later steps.

In summary, significant errors between $(A - \sigma^{(k)} E)^{-1}$ and Φ_m do not appear to be acceptable in any iteration of the approximate rational Lanczos algorithm. Limited numerical experience supports this result.

8.2.2 Approximate rational power methods

As with the dual RA algorithm, an appropriate approximate version of the rational power algorithm follows readily with the modifications seen in Algorithm 8.1 for the approximate RK algorithm. There appears to be little value in such an approach, however, because the the advantages of the dual RA approach over a two-sided RP method remain.

With the collapse of the elegant rational Lanczos theory in the approximate case, one loses a low-memory (short recursion) projection technique. Although the dual RA approach is reliable, storing the $O(MN)$ elements of V and Z may exceed available memory. This fact is especially true when approximate solves are utilized; one expects the model dimension, M, to grow slightly in compensation for the solve inaccuracies. An approach requiring only $O(N)$ storage is achievable through a one-sided RP method. Again, we turn to the idea of a prespecified Z matrix as in Section 7.3. If Z is completely known a prior, the columns of V (and associated columns of $\hat{A}$ and $\hat{E}$) can be computed without a full knowledge of one another, because the RP approach does not incorporate orthogonalization. In the m^{th} step of the sequential, approximate RP algorithm, one computes $\tilde{v}_m = \Phi_m E v_{m-1}$ (or $\tilde{v}_m = \Phi_m b$ if σ_m is a new interpolation point). After scaling $\tilde{v}_m$, one then computes the entire m^{th} column of $\hat{A}_m$ and $\hat{E}_m$, i.e., $Z^T A v_m$ and $Z^T E v_m$. Once these low-order columns are acquired, $\tilde{v}_{m+1}$ and $\tilde{z}_{m+1}$ are computed and the vectors v_m and z_m are discarded from memory. At most, two length N vectors are required for storage in this iteration.

The prespecified Z matrix must meet several conditions if this approach is to be successful. First, a matrix $Z_{\tilde{M}}$ of size $N \times \tilde{M}$ must be known ahead of time, where the size of $\tilde{M}$ is at least as great as M. In the proposed low-memory RP approach (unlike the parallel version in Section 7.3), the m^{th} columns of $\hat{A}_m$ and $\hat{E}_m$ must be computed in their entirety during the m^{th} iteration. Computing the entire first column of $\hat{A}$ in the first iteration therefore requires the knowledge of M immediately, but we would prefer not to specify the final value for the reduced-order model size M a priori (hopefully, M is adapted to the complexity of the dynamic system). To avoid this a priori specification, choose an initial $\tilde{M}$ larger than the estimate for M and work with $Z_{\tilde{M}}$ throughout the algorithm. To be conservative, make a guess for M prior to the model reduction and choose $\tilde{M}$ to be several times that. Of course, if $\tilde{M}$ is significantly larger than the eventual true value of M, then work is wasted. At each iteration, one computes $Z_{\tilde{M}}^T A v_m$ and $Z_{\tilde{M}}^T E v_m$ when, in fact, only $Z_M^T A v_m$ and $Z_M^T E v_m$ are eventually needed (the final reduced-order pencil is cut from size $\tilde{M} \times M$ to $M \times M$). For this reason, the product of $Z_{\tilde{M}}^T$ with a vector must be a relatively cheap operation, so that wasted work is not too expensive. Additionally, it is hoped that $Z_{\tilde{M}}$ can be compactly stored, because it (but not V) must be available in its entirety.

As long as V satisfies (3.1), any value of Z leads to a reduced-order model which matches M moments. In Section 7.3, it was suggested to simply choose a random Z. A random $Z_{\tilde{M}}$ can be compactly stored, but a matrix-vector product involving $Z_{\tilde{M}}^T$ is an $O(\tilde{M}N)$ operation. For this reason, consider going to a $Z_{\tilde{M}}$ that is composed of random integers or is sparse. Choosing $Z_{\tilde{M}}$ to be the first $\tilde{M}$ columns of the identity matrix $I_{\tilde{M}}$ leads to extremely low storage and computations. However, sparsity patterns better suited to the structure of the problem may improve convergence. An appropriate choice for $Z_{\tilde{M}}$ is a topic for additional research.

An algorithm implementing a one-sided, approximate rational power method is presented as Algorithm 8.4. The values of J_k in this algorithm should be kept small (< 5)

in an attempt to reduce dependencies among the columns of V. Dependencies may arise, however, and require the singular value postprocessing that was discussed at the conclusion of Section 4.1.1. A small J_k and correspondingly large K is not as significant of a concern in the approximate version. In the approximate RP algorithm, complete factorizations are no longer computed at every interpolation point. With iterative linear system solvers, for example, there may not be a significant cost advantage to staying at a few interpolation points.

Algorithm 8.4 Rational Krylov (Approximate RP Version)

Initialize: $m = 0$ and $Z_{\tilde{M}}$

For $k = 1$ to K,

  For $j_k = 1$ to J_k,

    (S8.4.1) If $j_k = 1$,

      $\tilde{v}_m = \Phi_m b$;

    else

      $\tilde{v}_m = \Phi_m(A - \zeta_m E)v_{m-1}$;

    end

    (S8.4.2) $v_m = \tilde{v}_m / \|\tilde{v}_m\|_2$;

    (S8.4.3) $\hat{a}_m = Z_{\tilde{M}}^T A v_m$ and $\hat{e}_m = Z_{\tilde{M}}^T E v_m$;

    (S8.4.4) $m = m + 1$;

  end

end

An implementation of Algorithm 8.4 requires one approximate solve, one matrix-vector product (mat-vec) with A, one mat-vec with E, and one mat-vec with $Z_{\tilde{M}}$. However, $Z_{\tilde{M}}$ is hopefully chosen in a way to reduce expenses. The one-sided RP algorithm also cuts cost by avoiding any operations involving A^T or E^T; the work per iteration is divided in half. Assuming $Z_{\tilde{M}}$ is chosen for compact storage and Φ_m involves sparse operations, then the overall memory required by the algorithm is $O(N + \tilde{M}M)$ elements. This value represents an order M reduction versus the approximate dual RA algorithm.

8.2.3 Comparisons

The successful introduction of approximate solves into the model-reduction techniques of Chapter 4 is not a trivial task. The theory governing the determination of modeling error or interpolation points is no longer exact. Fast implementations may no longer be based on elegant, short recursions, and perhaps most alarming, the debate over successful iterative methods and preconditioners for generating the approximate solutions is far from settled in the numerical linear algebra literature. Determining the right level of preconditioning for a fixed system of linear equations is oftentimes a task involving trial and error.

Nevertheless, the possible payoff with approximate solutions is great. In the remainder of this section, the impact of approximations on the model reduction of two moderately sized problems is considered. The breakdown of Lanczos-type approaches with approximate solves is demonstrated. However, we show that approximate rational interpolation is possible without exact matrix factorizations. Finally, some initial insights into working with approximate solves are related.

The first problem considered in the following examples arises from a discretization of the partial differential equation (PDE) [99],

$$\frac{\partial x}{\partial t} = \frac{\partial^2 x}{\partial z^2} + \frac{\partial^2 x}{\partial v^2} + 20\frac{\partial x}{\partial z} + 180x + f(v,z)u(t). \tag{8.9}$$

In (8.9), x is a function of time (t), vertical position (v) and horizontal position (z). The boundaries of interest in this problem lie on a square with opposite corners at $(0,0)$ and $(1,1)$. The function $x(t,v,z)$ is zero on these boundaries. This PDE can be discretized with centered difference approximations on a grid of $n_v \times n_z$ points [2]. The discretization grid, when $n_v = 3$ and $n_z = 5$, is shown in Figure 8.2. A state-space equation of dimension $N = n_v n_z$ results from the discretization. The sparsity pattern of the resulting A matrix, when $n_v = 7$ and $n_z = 12$, is shown in Figure 8.3. The input vector of the system corresponds to $f(v, z)$ and is composed of random elements. The output vector of the system is equated to the input vector for no other reason than simplicity.

For a second test problem, we consider n_v interconnects that are running parallel to each other horizontally and are coupled by mutual inductance vertically. For example, there are $n_v = 2$ parallel interconnects in Figure 8.4, each consisting of $n_z = 3$ segments of resistors, capacitors and inductors. The elements in the interconnects are randomly generated. The variance of the resistance values is 1000, the variance of the capacitance values is 10^{-13} and the variance of the inductance values is 10^{-10}. The input to the circuit is a current source at the leftmost segment of the top interconnect. The output of the circuit is the voltage at the rightmost segment of the bottom interconnect. For such a system, MNA equations of dimension $n_v(2n_z - 1)$ result. The first $n_v n_z$ quantities in the state vector are node voltages; the remaining ones are inductor currents. The sparsity pattern of the resulting $sE - A$ matrix, when $n_v = 3$ and $n_z = 30$, is shown in Figure 8.5. This sparsity pattern and the following examples were obtained via MATLAB.

Example 8.1 *This example explores the impact of errors in the linear equation solutions on various model-reduction implementations. Specifically, equations of the form*

$$\begin{aligned} (A - \sigma_m E)\underline{\tilde{v}}_m &= q_m \\ (A - \sigma_m E)^T \underline{\tilde{z}}_m &= w_m \end{aligned} \tag{8.10}$$

arise in the various RK implementations, which are solved to assorted degrees of numerical precision (16, 10, 6, *and* 2 *digits of accuracy in the following). Besides the variations in solver accuracy, different model-reduction implementations are considered: dual RA with a real interpolation point, RL with a real interpolation point, dual RA with a few imaginary interpolation points, RL with a few imaginary interpolation points, and dual RA with numerous adaptively placed, imaginary interpolation points. These combinations of model-reduction algorithms and solver accuracies were applied to a discretization of (8.9) on a* 7×12 *grid. The frequency response of this discretization is in Figure 8.6.*

Figure 8.7 plots the modeling error resulting from dual RA iterations (with a real shift at 2π*) as the solver accuracy varies. The* 'o' *line corresponds to* 16 *digits of solver*

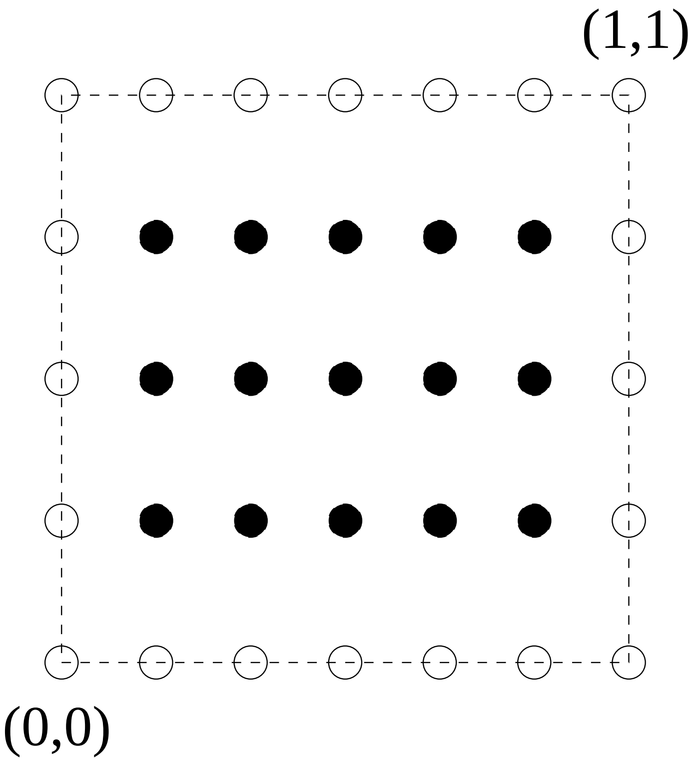

Figure 8.2: Discretized PDE Grid, 3×5 Case

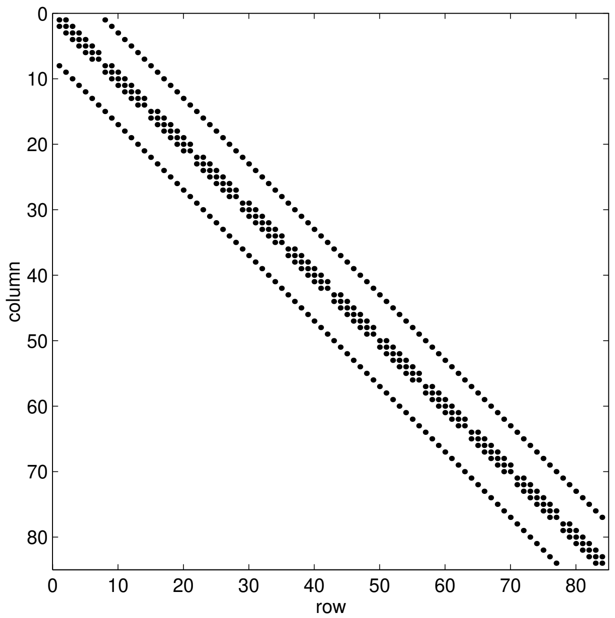

Figure 8.3: Discretized PDE Sparsity Pattern, 7×12 Case

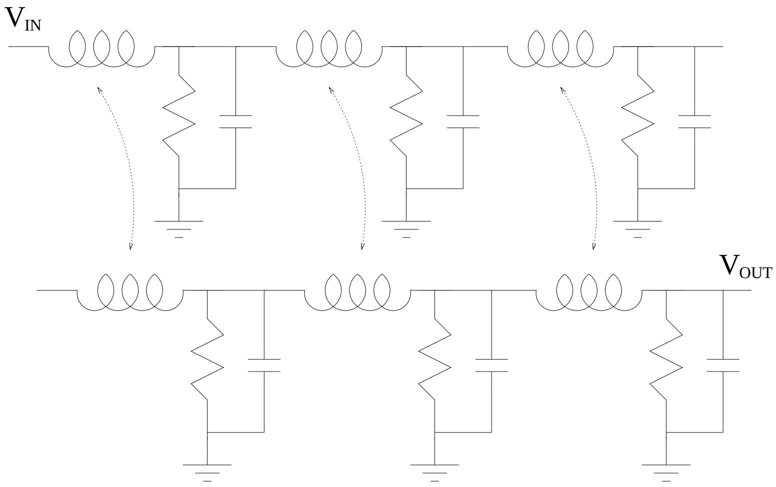

Figure 8.4: Coupled Interconnects, 2×3 Case

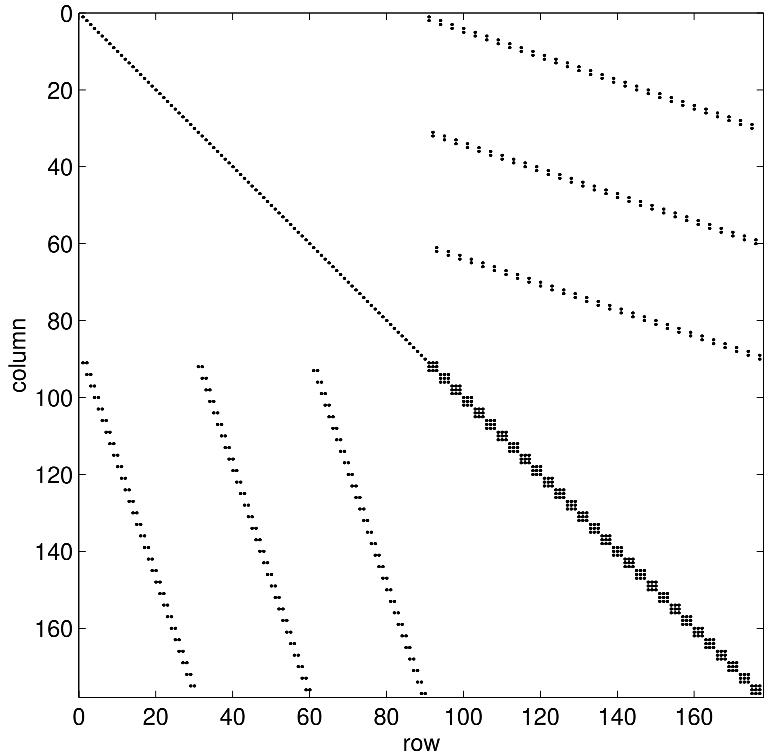

Figure 8.5: MNA Sparsity Pattern for Coupled Interconnects, 3×30 Case

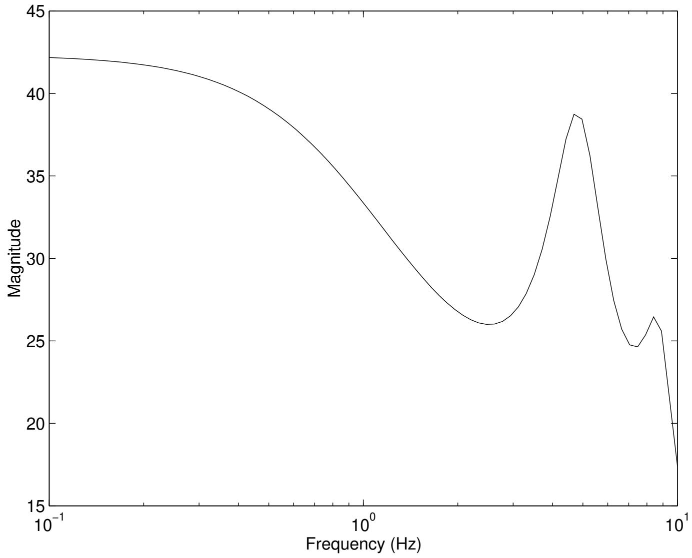

Figure 8.6: Frequency Response of Example 8.1

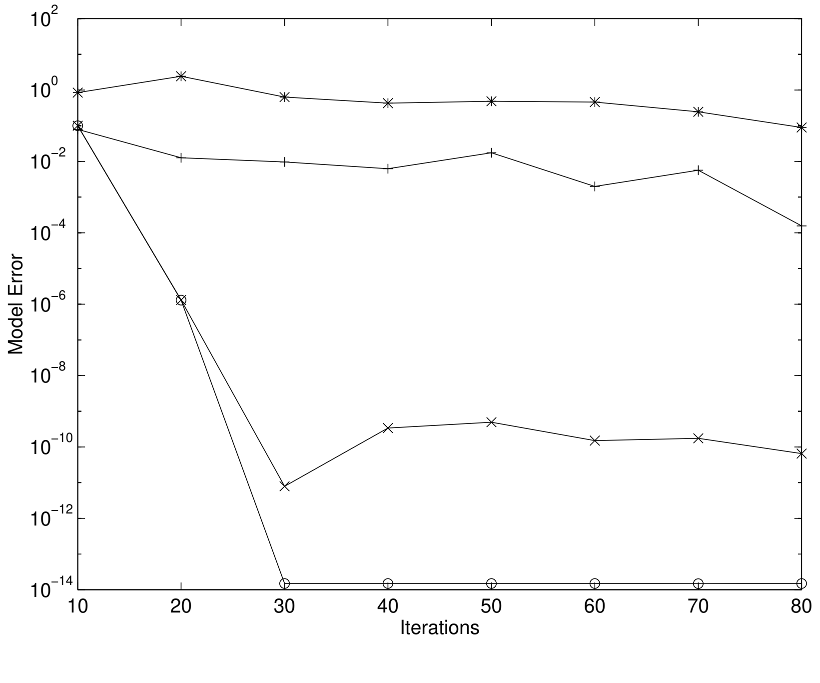

Figure 8.7: Finite Precision Dual RA Results for Example 8.1 (o= 16 digits, x= 10 digits, += 6 digits, $*$= 2 digits)

precision, the 'x' *line corresponds to* 10 *digits of precision, the* '+' *line to* 6 *digits, and the* '$*$' *line to only* 2 *digits. A quick analysis of these plots shows that the convergence of the reduced-order model tends to stagnate at a level that is related to the solver precision in a nonlinear fashion. When the noise level is reached in a given case, continued iterations during this stagnation period only add random directions to* V *and* Z*—directions not particularly suited for reducing the modeling error. This stagnation continues until the number of iterations* m *becomes nearly* N*. When* m *is* N*, the modeling error is guaranteed to drop to about machine precision if orthogonal* V *and* Z *were maintained (in all cases orthogonality was maintained up to the limit of machine precision). A dual RA reduced-order model with* $m = N$ *is simply a different realization of the original model. Although not shown, even the* '$*$' *line drops off sharply towards zero when* $m = N - 1$*.*

The results as the model-reduction algorithm varies are presented in Tables 8.1 through 8.4. In each table, the linear equations in the model-reduction algorithm are solved to a different degree of numerical precision. The various dual RA implementations continue to converge in a reasonable fashion as the degree of numerical precision drops. There is of course some difference across the columns due to the varying interpolation point schemes. Numerous imaginary interpolation points that are tuned to the modeling error consistently yield the best results. However, the performance with a single real interpolation point or two imaginary points is acceptable, as well, when dual RA is used for all but significant levels of solver noise.

The results with RL algorithms are presented in the last two columns of Tables 8.1 through 8.4. Even with significant digits of precision in Table 8.1, the RL results are slightly inferior to those with the rational Arnoldi implementations. This slight discrepancy is consistent with the RL approach's dependence on biorthogonality (recall Example 4.5). The convergence of the RL approaches rapidly worsens when fewer digits of accuracy are achieved in the linear equation solutions. In fact, the convergence of the RL results level off immediately when fewer than 6 digits of accuracy are maintained in Tables 8.3 and 8.4 . This behavior is consistent with the observations made in Section 8.2;

Table 8.1: Convergence with 16 Digits of Precision in Example 8.1

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	1.0012e-01	7.6131e-02	5.3675e-02	9.9563e-02	7.6131e-02
20	1.2973e-06	5.7945e-08	1.5544e-14	1.0582e-04	5.7944e-08
30	1.4979e-14	1.5185e-14	1.5557e-14	6.9162e-10	9.6934e-13
40	1.4957e-14	1.5154e-14	1.3143e-13	6.9102e-10	9.6930e-13
50	1.4911e-14	1.5205e-14	2.5463e-13	6.9102e-10	9.6930e-13
60	1.4974e-14	1.5257e-14	2.3132e-13	6.9102e-10	9.6930e-13
70	1.4860e-14	1.5199e-14	5.9524e-13	6.9102e-10	9.6930e-13
80	1.4932e-14	1.5194e-14	3.2808e-13	6.9102e-10	9.6930e-13

Table 8.2: Convergence with 10 Digits of Precision in Example 8.1

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	1.0012e-01	7.6131e-02	5.3322e-02	9.9582e-02	7.6131e-02
20	1.3054e-06	5.7963e-08	7.1350e-12	8.0087e-04	1.0609e-06
30	7.8977e-12	2.2175e-13	1.4679e-14	2.6930e-06	1.9547e-08
40	3.4019e-10	2.4578e-13	1.4752e-14	2.6937e-06	1.9547e-08
50	4.9277e-10	6.6366e-13	1.4727e-14	2.6937e-06	1.9547e-08
60	1.5035e-10	1.3442e-12	1.4653e-14	2.6937e-06	1.9547e-08
70	1.7546e-10	2.9901e-12	1.4665e-14	2.6937e-06	1.9547e-08
80	6.5200e-11	3.9822e-13	1.4857e-14	2.6937e-06	1.9547e-08

Table 8.3: Convergence with 6 Digits of Precision in Example 8.1

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	7.9312e-02	7.6256e-02	2.8675e-02	9.9623e-02	7.6128e-02
20	1.2610e-02	1.0527e-05	2.4593e-10	9.9211e-02	2.5076e-04
30	9.6187e-03	8.5144e-06	4.3110e-11	9.9052e-02	2.2276e-04
40	6.2384e-03	5.1706e-05	5.0282e-12	9.9039e-02	2.2332e-04
50	1.7408e-02	1.6411e-05	7.6168e-12	9.9196e-02	2.2354e-04
60	1.9966e-03	7.5595e-06	3.0844e-12	9.9049e-02	2.2350e-04
70	5.6909e-03	1.0981e-05	7.1168e-12	9.9054e-02	2.2349e-04
80	1.5544e-04	5.8382e-06	1.3037e-12	9.9056e-02	2.2357e-04

Table 8.4: Convergence with 2 Digits of Precision in Example 8.1

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	8.4919e-01	3.7454e-01	2.5081e-01	4.7903e-01	4.8004e-01
20	2.4262e+00	8.9720e-02	7.3547e-03	4.7906e-01	2.1937e-01
30	6.3352e-01	1.2191e-01	2.3288e-03	4.7906e-01	1.6319e-01
40	4.3093e-01	4.8418e-02	4.2221e-04	4.7906e-01	5.8513e-01
50	4.8595e-01	7.1805e-02	6.9019e-04	4.7906e-01	1.1733e-01
60	4.6206e-01	1.4629e-03	2.0862e-03	4.7906e-01	1.2958e-01
70	2.4451e-01	4.1994e-04	2.6235e-04	4.7906e-01	1.0641e-01
80	8.9394e-02	1.8301e-04	6.8138e-05	4.7906e-01	1.5627e-01

numerical errors in rational Lanczos do not dissipate with additional modeling iterations. The errors in the RL approximations continue to be large, even when m surpasses N.

Example 8.2 *We also repeated the above experiments for the previously described interconnect problem with $n_v = 3$ and $n_z = 30$. The frequency response of this system is displayed in Figure 8.8. The convergence of a dual RA implementation with a real shift at $2\pi 10^{10}$ is plotted in Figure 8.9 for several different levels of solver precision. Again, the* 'o' *line corresponds to* 16 *digits of solver precision, the* 'x' *line to* 10 *digits, the* '+' *line to* 6 *digits, and the* '$*$' *line to* 2 *digits. The results are plotted through the first* 100 *iterations, although N in this problem is* 177. *Unlike Figure 8.7, the lines in Figure 8.9 are not easily distinguishable. With high solver accuracy (*16 *digits of precision), convergence is only gradual because this problem's dynamics are not easily captured with a single interpolation point. Less solver accuracy (*10 *or* 6 *digits) is sufficient to reproduce this gradual convergence, as well. The results in Figures 8.7 and 8.9 seem to suggest that the level of solver accuracy in the $(m+1)^{st}$ iteration need only be consistent with the degree of model convergence after m iterations. This relation between the current model error and the required solver accuracy in the m^{th} iteration (if it does indeed exist) is clearly nonlinear, however. Characterizing the minimal level of solver accuracy for near-ideal, reduced-order model convergence is an interesting question requiring further investigation.*

Again in this example, we also consider the convergence behavior of various RK implementations. The dependence of this behavior on the solver precision is indicated in Tables 8.5 through 8.8. The most important feature in these plots is the stagnation in the RL implementations as solver accuracy diminishes. For the case of two digits of precision (which is probably the closest to reality in actual approximate solution techniques), the implemented RL approaches are completely useless. As noted above, the results of the various RA implementations are barely affected by the variations in solver precision.

In practice, a loss of numerical precision in the solver follows naturally from the approximations existing in inexact DS preconditioners or inner solver iterations. Yet

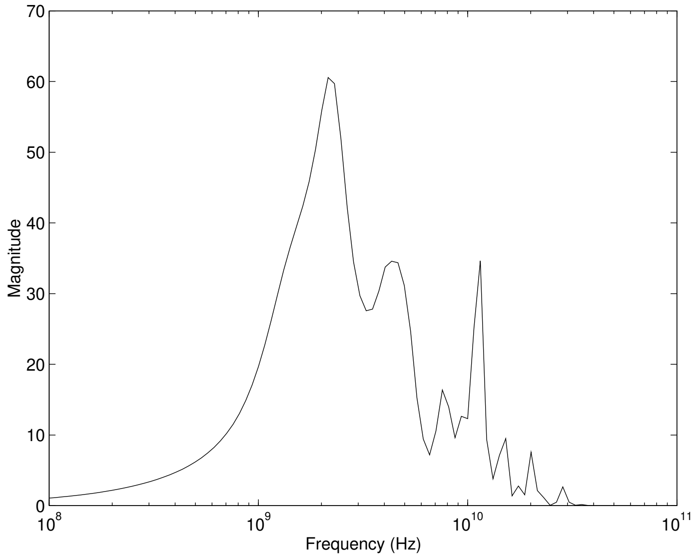

Figure 8.8: Frequency Response of Example 8.2

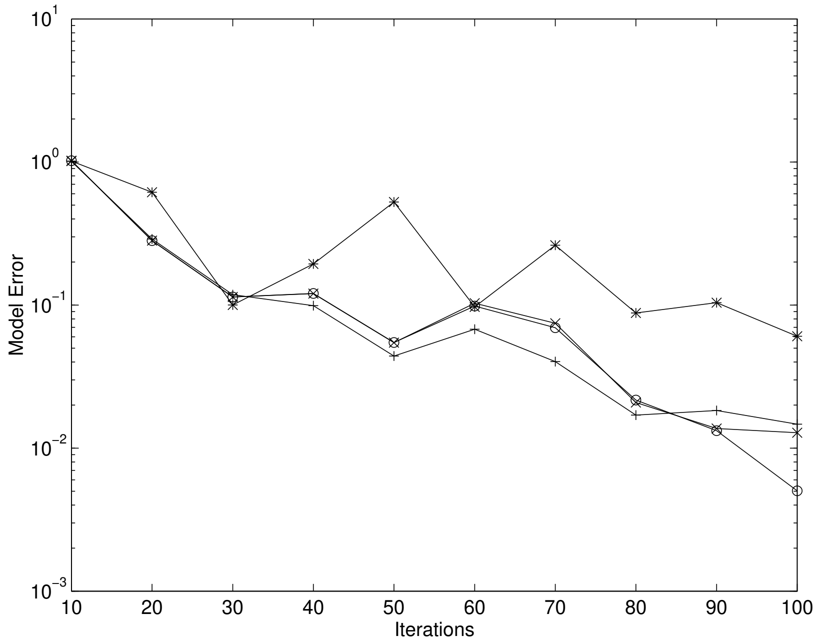

Figure 8.9: Finite Precision Dual RA Results for Example 8.2 (o= 16 digits, x= 10 digits, += 6 digits, $*$= 2 digits)

Table 8.5: Convergence with 16 Digits of Precision in Example 8.2

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	1.0227e+00	4.0696e-01	1.4265e+00	1.0227e+00	4.0696e-01
20	2.8240e-01	2.7610e-01	2.1883e-01	2.8240e-01	2.7610e-01
30	1.1404e-01	2.7239e-01	9.6314e-02	1.1404e-01	2.7239e-01
40	1.2071e-01	3.4685e-01	2.4581e-02	1.2071e-01	3.4685e-01
50	5.4785e-02	2.4587e-01	2.0032e-02	5.4785e-02	2.6438e-01
60	9.8426e-02	2.2455e-01	2.0912e-02	9.8425e-02	2.6073e-01
70	6.9414e-02	6.8882e-02	1.3267e-02	6.9414e-02	2.5526e-01
80	2.1697e-02	7.0448e-02	3.9350e-03	2.4651e-02	2.5606e-01
90	1.3254e-02	5.4822e-02	2.5981e-03	1.4218e-02	2.5698e-01
100	5.0322e-03	4.5268e-02	4.9065e-04	1.4703e-02	2.6290e-01

Table 8.6: Convergence with 10 Digits of Precision in Example 8.2

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	1.0227e+00	4.0696e-01	1.4265e+00	1.0227e+00	4.0696e-01
20	2.8240e-01	2.7610e-01	2.1882e-01	2.8239e-01	2.7610e-01
30	1.1406e-01	2.7239e-01	9.6314e-02	1.1413e-01	2.7239e-01
40	1.2035e-01	3.4674e-01	2.4581e-02	1.2072e-01	3.0056e-01
50	5.4705e-02	2.4605e-01	2.0032e-02	5.4746e-02	3.4728e-01
60	1.0319e-01	2.4037e-01	2.0912e-02	9.5094e-02	2.6343e-01
70	7.4712e-02	9.6700e-02	1.3266e-02	4.3219e-02	2.6649e-01
80	2.0836e-02	6.8930e-02	3.9349e-03	3.7064e-02	2.7249e-01
90	1.3713e-02	4.6907e-02	2.5981e-03	1.6420e-02	2.6041e-01
100	1.2824e-02	4.5367e-02	4.9067e-04	1.6671e-02	2.6200e-01

Table 8.7: Convergence with 6 Digits of Precision in Example 8.2

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	1.0112e+00	4.0446e-01	1.4250e+00	1.0199e+00	4.0720e-01
20	2.8986e-01	2.7599e-01	1.8035e-01	1.4775e+00	2.7624e-01
30	1.1816e-01	2.7328e-01	1.4771e-01	4.6811e-01	2.7257e-01
40	9.9176e-02	2.7947e-01	5.0217e-02	4.3410e-01	2.7457e-01
50	4.4016e-02	2.6507e-01	2.1662e-02	4.3203e-01	2.7487e-01
60	6.7788e-02	2.4560e-01	1.4191e-02	4.3610e-01	2.7429e-01
70	4.0316e-02	2.6098e-01	5.4195e-03	4.3834e-01	2.7637e-01
80	1.7033e-02	1.25975e-01	5.2231e-03	4.3850e-01	2.7490e-01
90	1.8339e-02	1.0604e-01	3.6121e-03	4.3846e-01	2.7584e-01
100	1.4709e-02	7.5908e-02	6.9394e-04	4.3846e-01	2.7380e-01

Table 8.8: Convergence with 2 Digits of Precision in Example 8.2

m	Modeling Error				
	Dual RA Real σ	Dual RA Imag σ	Dual RA Adapted σ	RL Real σ	RL Imag σ
10	1.0155e+00	1.3251e+00	1.2823e+00	1.0000e+00	2.7562e+00
20	6.1500e-01	5.7633e-01	3.3702e-01	1.0000e+00	2.7090e+00
30	1.0031e-01	8.9612e-01	8.7027e-02	1.0000e+00	1.1193e+01
40	1.9410e-01	3.3256e-01	7.2225e-02	1.0000e+00	4.7387e+00
50	5.2599e-01	3.7340e-01	3.8174e-02	1.0000e+00	4.0128e+00
60	9.7136e-02	3.9263e-01	3.2044e-02	1.0000e+00	4.2823e+00
70	2.6194e-01	4.1133e-01	3.0033e-02	1.0000e+00	1.8649e+00
80	8.8138e-02	4.8893e-01	1.4716e-02	1.0000e+00	7.7877e+00
90	1.0393e-01	5.4277e-01	1.2229e-02	1.0000e+00	2.6655e+00
100	6.0593e-02	3.7207e-01	1.3597e-02	1.0000e+00	1.4687e+01

regardless of the exact source of the inaccuracy, the approximate dual RA approach can lead to valid reduced-order models. This fact is demonstrated in the following examples on some larger versions of the two problems of interest. The results of this experimentation are preliminary and are not intended to reflect a final effort towards incorporating approximate solves. Merely demonstrating that exact factorizations of $(A - sE)$ are not required is apparently a novel result.

Example 8.3 *Consider a discretization of the PDE in (8.9) on a* 40×60 *grid. This grid leads to an* A *matrix of dimension* $N = 2400$ *with* 11800 *nonzero elements. The approximate dual RA algorithm is applied to this problem with a real shift of* 2π*. The systems of linear equations (8.10) are approximately solved with the GMRES method [8]. Each inner GMRES iteration incoporates an inner-preconditioner*† *formed with the* $ILUT(4,0)$ *method of [8]. That is, an ILUT preconditioner is applied to (8.10) prior to the execution of the GMRES method. The operator* Φ_m *combines both a fixed inner preconditioner and an iterative solver.*

The results of 100 *dual RA iterations with either five or twenty GMRES inner iterations are presented in Table 8.9. Thus,* 500 *total inner iterations take place in the first case and* 2000 *in the second (these iterations only involve matrix-vector products however). Different values of* ζ_m *(*∞ *and* σ_m*) are also presented in this table. Note that the* $\zeta_m = \sigma_m$ *case performs well when the DS preconditioner is poorer (fewer GMRES steps), but is unacceptable when the DS preconditioner is more accurate. The opposite behavior occurs when* ζ_m *is* ∞*, i.e.,* $(A - \zeta_m E)$ *is replaced with* E*. Although the best results are obtained in the second column, the results with a well-chosen* ζ_m *and only five GMRES iterations in column three are reasonably good.*

Example 8.4 *Consider six parallel interconnects that each consist of* 150 *segments. This problem has an* N *of size* 1794*, an* A *matrix with* 4476 *nonzero elements and an* E *matrix with* 6294 *nonzero elements. The approximate dual RA algorithm was utilized for*

†The DS preconditioner Φ_m is used to speed the convergence of the reduced-order model (the outer iteration) over a range of frequencies. An inner preconditioner is utilized in the inner solver iteration to speed the determination of a solution to (8.10).

model reduction with fifteen GMRES inner iterations and $\sigma = 2\pi 10^9$. In this example, though, we vary the inner preconditioner used in the inner iteration between $ILUT(2,0)$ and $ILUT(4,0)$. That is, more fill is allowed in the inner preconditioner in the latter case. The results of 100 *modeling iterations are in Table 8.10. The amount of inner preconditioning is critical in this problem; significant differences result from only moderate changes in Φ_m.*

Computing reasonably accurate solutions to the dual system of equations demands good DS preconditioners. Even when iterative Krylov techniques are involved in the solution, this inner iteration often involes an inner preconditioner, e.g., as in Example 8.3. It is felt that the use of improved inner preconditioners and inner iterations is crucial to the successful implementation of a faster, approximate model-reduction method, i.e., Section 8.2.2.

8.3 Relating Inner and Outer Recursions

In prior portions of this chapter, the generation of Φ_m is considered to be an independent subproblem in the model-reduction procedure. However, because the choice of Φ_m depends on the properties of A and E, it is logical to utilize every piece of information about A and E during the linear system solves. The search and constraint subspaces, which are iteratively generated during past model-reduction steps, provide useful information about A and E. Consider the computation of the next direction in the outer modeling iteration. Ideally, one solves the equation

$$(A - \sigma_{m+1} E)\underline{\tilde{v}}_{m+1} = (A - \zeta_{p_m} E) v_{p_m} \tag{8.11}$$

exactly. If this is not possible, some sort of approximation is required. The reduced-order model itself provides a guess for the solution to (8.11),

$$\tilde{v}^0_{m+1} = V_m(\hat{A}_m - \sigma_{m+1}\hat{E}_m)^{-1} Z_m^T (A - \zeta_{p_m} E) v_{p_m}. \tag{8.12}$$

This initial guess finds an approximate solution in the column space of V_m that satisfies a Petrov-Galerkin constraint with respect to Z_m. Using the projection associated with

Table 8.9: Convergence with Approximate Solutions in Example 8.3

m	Modeling Error			
	$\zeta_m = \infty$		$\zeta_m = \sigma_m$	
	5 GMRES steps	20 GMRES steps	5 GMRES steps	20 GMRES steps
10	9.9508e-01	1.3772e-03	5.9617e+00	1.6515e+00
20	1.1376e+00	1.9907e-04	9.0502e-02	1.0812e+00
30	2.1930e+00	4.5920e-05	3.7690e-01	1.1226e+00
40	1.2878e+00	2.7160e-04	9.2384e-03	1.2008e+00
50	3.4732e+00	3.4845e-04	3.2868e-02	1.6405e+00
60	3.4849e+00	8.7005e-05	2.7594e-02	3.2289e-01
70	1.2312e+00	8.2442e-05	8.1249e-02	1.2692e-01
80	2.0851e+00	5.0187e-05	1.1214e-01	1.2623e-02
90	1.0403e+00	2.7940e-04	4.0154e-01	7.1056e-02
100	2.5813e+00	2.0818e-05	4.5188e-02	3.3366e-01

Table 8.10: Convergence with Approximate Solutions in Example 8.4

m	Modeling Error	
	ILUT(2,0)	ILUT(4,0)
10	1.0000e+00	6.8291e-01
20	1.0005e+00	4.6680e-01
30	8.7317e-01	3.7234e-01
40	2.1176e+00	4.7784e-01
50	5.8921e+00	2.8841e-01
60	4.4015e+00	2.0572e-01
70	1.3687e+01	1.2033e+00
80	1.2908e+01	1.6671e-01
90	1.2099e+01	1.1474e-01
100	1.1945e+01	1.0359e-01

model reduction to generate an initial guess $\tilde{v}^0_{m+1}$ can lead to better inner-solver results than simply choosing $\tilde{v}^0_{m+1}$ to be a random vector or a vector of zeros. In a sense, (8.12) recycles existing work rather than trying to solve (8.11) from scratch. Unfortunately, (8.12) is not enough by itself. Because the purpose of solving (8.11) is to compute a new direction $\tilde{v}_{m+1}$, an approximate solution that lies entirely in the existing directions of V_m is not acceptable.

The initial guess in (8.12) must be improved upon. This correction can be accomplished by incorporating the initial guess into (8.11) to form the problem

$$(A - \sigma_{m+1}E)(\underline{\tilde{v}}_{m+1} - \tilde{v}^0_{m+1}) = (A - \zeta_{p_m}E)v_{p_m} - (A - \sigma_{m+1}E)\tilde{v}^0_{m+1}. \tag{8.13}$$

An approximation $\tilde{v}_{m+1}$ then follows by approximately solving

$$(A - \sigma_{m+1}E)\underline{\tilde{u}}_{m+1} = (A - \zeta_{p_m}E)v_{p_m} - (A - \sigma_{m+1}E)\tilde{v}^0_{m+1}$$

and adding $\tilde{v}^0_{m+1}$ to the result. The computed vector $\tilde{u}_{m+1}$ is an update that hopefully leads to a better approximation, $\tilde{v}_{m+1} = \tilde{v}^0_{m+1} + \tilde{u}_{m+1}$. This update is, in fact, the new direction introduced into V_{m+1}; the vector $\tilde{u}_{m+1}$ satisfies

$$\operatorname{colsp}\left\{\left[\begin{array}{c|c} V_m & \tilde{v}_{m+1}\end{array}\right]\right\} = \operatorname{colsp}\left\{\left[\begin{array}{c|c} V_m & \tilde{u}_{m+1}\end{array}\right]\right\},$$

because $\tilde{v}^0_{m+1}$ lies in the column space of V_m. The update can be approximated as

$$\tilde{u}_{m+1} = \Phi_{m+1}\{(A - \zeta_{p_m}E)v_{p_m} - (A - \sigma_{m+1}E)\tilde{v}^0_{m+1}\}, \tag{8.14}$$

where Φ_m approximates $(A - \sigma_{m+1}E)^{-1}$. As before, Φ_{m+1} can be a fixed DS preconditioner, an iterative solver, or a preconditioned iterative solver. The rightmost term in (8.14) incorporates existing information from the outer modeling iteration into the approximate solve of (8.11). We try to avoid recomputing already known information during the formation of $\tilde{u}_{m+1}$.

The computation of $\tilde{u}_{m+1}$ can actually be simplified further by rewriting the initial guess as

$$\begin{aligned}\tilde{v}^0_{m+1} &= V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1}Z_m^T(A - \sigma_{m+1}E)v_{p_m} \\ &\quad +(\sigma_{m+1} - \zeta_{p_m})V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1}Z_m^T E v_{p_m} \\ &= v_{p_m} + (\sigma_{m+1} - \zeta_{p_m})V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1}Z_m^T E v_{p_m}.\end{aligned}$$

Using this expression for the initial guess, the update (8.14) becomes

$$\begin{aligned}\tilde{u}_{m+1} &= (\sigma_{m+1} - \zeta_{p_m})\Phi_{m+1} E v_{p_m} \\ &\quad -(\sigma_{m+1} - \zeta_{p_m})\Phi_{m+1}(A - \sigma_{m+1}E)V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1} Z_m^T E v_{p_m} \\ &= (\sigma_{m+1} - \zeta_{p_m})\Phi_{m+1} \{E v_{p_m} \\ &\quad -(A - \sigma_{m+1}E)V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1} Z_m^T E v_{p_m}\}. \end{aligned} \tag{8.15}$$

There are two important things to note about (8.15). First, the update $\tilde{u}_{m+1}$ consists of the vector $\Phi_{m+1} E v_{p_m}$, which is perhaps the most naive approximation to the ideal new direction $(A - \sigma_{m+1}E)^{-1} E v_{p_m}$, and a correction vector

$$\Phi_{m+1}(A - \sigma_{m+1}E)V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1} Z_m^T E v_{p_m},$$

which incorporates information from the existing reduced-order model. When Φ_{m+1} nears $(A - \sigma_{m+1}E)^{-1}$, the vector $\Phi_{m+1} E v_{p_m}$ nears the ideal direction, while the correction $V_m(\hat{A}_m - \sigma_{m+1}\hat{E})^{-1} Z_m^T E v_{p_m}$ is contained in colsp$\{V_m\}$ and is, hence, irrelevant. On the other hand, if the reduced-order model is accurate (a fortunate event), then $V_m(\hat{A} - \sigma_{m+1}\hat{E})^{-1} Z_m^T$ nears $(A - \sigma_{m+1}E)^{-1}$ and the update $\tilde{u}_{m+1}$ becomes small. A second important feature of (8.15) is that the parameter ζ_{p_m} only comes into play as a scaling. However, scalings do not matter in constructing subspaces, so that the unscaled vector

$$\Phi_{m+1}\left\{E v_{p_m} - (A - \sigma_{m+1}E)V_m(\hat{A}_m - \sigma_{m+1}\hat{E}_m)^{-1} Z_m^T E v_{p_m}\right\} \tag{8.16}$$

is a perfectly appropriate new direction for augmenting V_m. Even though Φ_m may not be an exact DS preconditioner, the choice of $\tilde{v}_{m+1}^0$ as a starting guess leads to a new direction (8.16) that is independent of ζ. We have thus found a second way (exact DS preconditioning was the first) for generating V directions that are independent of the evaluation point ζ.

As a side note, it is claimed that the derivation of (8.16) provides an alternative path for obtaining Davidson's method for the eigenvalue problem [98]. We return to this topic in Section 9.1.3.

Example 8.5 *If the discretized PDE in Example 8.3 is solved with the improved initial guess in (8.12) for* 100 *modeling iterations, the data in Table 8.11 are the result. The use of the improved starting vector leads to a convergence which is slightly improved over the best cases in Example 8.3. However, the best results in Example 8.3 required careful choices for ζ, an issue that is no longer a concern here. The results in this example are independent of ζ_{m+1}.*

Table 8.11: Improved Starting Vector Results in Example 8.5

m	Modeling Error	
	5 GMRES steps	20 GMRES steps
10	1.1939e+00	1.0132e-02
20	4.7650e-01	4.3593e-04
30	6.1100e-02	1.0544e-03
40	2.7000e-02	1.8763e-04
50	1.4200e-02	3.2375e-04
60	4.1100e-02	2.3447e-04
70	4.6500e-02	7.1564e-05
80	3.1000e-03	5.2044e-04
90	9.5000e-03	6.1742e-05
100	4.5000e-03	1.8461e-04

There is at least one other approach for using the outer modeling projection subspaces as an aid in the solution of the linear systems of equations. The thrust of this approach is to iteratively solve the equation

$$(I - V_m V_m^T)(A - \sigma_{m+1} E)\underline{\tilde{v}}_{m+1} = (I - V V_m^T)(A - \zeta_{p_m} E) v_{p_m} \tag{8.17}$$

rather than (8.11). An approximate solution $\tilde{v}_{m+1}$ is iteratively constructed for (8.17), e.g., from the subspace

$$\mathcal{K}_j((I - V_m V_m^T)(A - \sigma_{m+1} E), (I - V V_m^T)(A - \zeta_{p_m} E) v_{p_m}),$$

which is orthogonal to the outer search subspace V_m. In this manner, effort is not wasted by computing $\tilde{v}_{m+1}$ in directions already present in V_m. A dual approach can also be obtained for $\tilde{z}_{m+1}$. The concept of keeping inner-iterations orthogonal to the outer was developed in [100] for solving fixed systems of linear equations. Modifications of Davidson's method also exist in the eigenvalue problem which keep $\tilde{v}_{m+1}$ orthogonal to the initial guess.

There is a drawback with approaches such as (8.16) and (8.17). The entire outer matrices V and Z must be available and utilized at every iteration. This utilization increases costs and prevents fast iterations such as the one suggested in Section 8.2.2. Of course, compromises might be possible that involve only the most recent directions in V and Z or some dominant directions in V and Z.

It is beyond the scope of this dissertation (and perhaps any single work) to completely characterize all issues pertinent to preconditioning, initial solution guesses and inner-outer relations. These topics run throughout iterative approaches in many applications and continue to be an active area of research. Hopefully, this chapter has demonstrated though that such solver techniques are applicable and of interest in the model reduction of large-scale dynamic systems.

CHAPTER 9

ITERATIVE EIGENVALUE SOLVERS

This chapter focuses on preconditioned iterative eigenvalue solvers (PIES), a class of methods that are closely tied to the model-reduction techniques of Chapters 4 and 8. The relations between PIES and the proposed model-reduction techniques are explained in Section 9.2. Prior to this discussion, however, the PIES are surveyed and classified according to a few well-defined choices. This is not to say, however, that all worthwhile iterative eigensolvers utilize preconditioners and fall into the proposed classification. The implicitly restarted Arnoldi algorithm [101] is an example of a notable nonpreconditioned approach for eigenvalue computations.

In this section, it is assumed that the problem to be solved is a standard, symmetric one, i.e., $A = A^T$ and $E = I$. Concentrating on this commonly occurring and much-studied problem allows for a more straightforward introduction to the eigenvalue themes. Extensions of the developed techniques to the more general pencil $(A - \lambda E)$ are oftentimes straightforward, yet even a comprehensive treatment of the symmetric problem is a significant task.

9.1 Existing Techniques

Approaches for solving the generalized problem, $A\mathbf{x} = \lambda\mathbf{x}$, are well-developed when the dimension of the problem, n, is small [11]. Computing even a few eigenvalues and eigenvectors when A is large and sparse, however, is typically a difficult task to this day. Although elegant and relatively inexpensive, the classical iterative methods of Arnoldi [75] and Lanczos [39] oftentimes fall short, particularly when the target is an interior portion of the spectrum. As a result, numerous authors (beginning apparently with Davidson in

1975 [98]) developed significant modifications and extensions of the Arnoldi and Lanczos methods. For the most part, these modifications incorporate preconditioning in some fashion to yield improved rates of convergence.

Nearly all PIES implement the Rayleigh-Ritz procedure [102] with respect to some subspace $\mathcal{S}$ and transformed matrix (TA). In our previous terminology, $\mathcal{S}$ is a search subspace corresponding to an orthogonal projection. The acquired eigenvector approximations lie in this search subspace. The approximate eigenvalues are acquired from the low-order pencil (Y^TTAY, Y^TTY), where the column space of $Y \in \mathbb{R}^{N\times M}$ is $\mathcal{S}$. As in model reduction, the eigenvalues of (Y^TTAY, Y^TTY) do depend upon the selection of both T and Y. In fact, the standard forms of T and Y can be characterized through a few well-defined choices.

Assuming that a single eigenvalue λ_d and a corresponding eigenvector $\mathbf{x}_d$ are desired, the column space of Y typically takes the form

$$\text{colsp}\,\{Y\} = \left[\begin{array}{cccc} y_1 & P_2(A-\zeta_2 I)Y_1\hat{\mathbf{x}}_{d_1} & P_3(A-\zeta_3 I)Y_2\hat{\mathbf{x}}_{d_2} & \ldots \end{array}\right] \tag{9.1}$$

in the eigenvalue literature. The matrix Y_m contains the first m columns of Y. The first column of Y, $Y_1 = y_1$ is specified by the user and it is typically chosen to be either a random vector or a vector of 1's. The matrices P_m play the role of varying ES preconditioners (ES stands for eigenvalue solver, playing the role of DS in previous chapters). The vector $Y_m\mathbf{x}_{d_m}$ is the eigenvector approximation after m iterations.

Given Y, an approximation for $\hat{\mathbf{x}}_d$ in its column space is sought. Similar to the model-reduction problem, this approximation arises out of the low-order approximation $(Y_m^TT_mAY_m, Y_m^TT_mY_m)$. Assume $\hat{\lambda}_{d_m}$ is the eigenvalue of $(Y_m^TT_mAY_m, Y_m^TT_mY_m)$ perceived to be closest to λ_d. Then, the reduced-order eigenvector $\hat{\mathbf{x}}_{d_m}$ of $(Y_m^TT_mAY_m, Y_m^TT_mY_m)$ corresponding to $\hat{\lambda}_d$ leads to an approximation $Y\hat{\mathbf{x}}_{d_m}$ for $\mathbf{x}_d$. This approximate eigenvector $Y\hat{\mathbf{x}}_{d_m}$ satisfies the Galerkin type condition

$$Y_m^T\{T_m(A-\hat{\lambda}_{d_m}I)(Y_m\hat{\mathbf{x}}_{d_m})\} = 0 \tag{9.2}$$

where $T_m(A-\hat{\lambda}_{d_m}I)Y_m\hat{\mathbf{x}}_{d_m}$ is the residual associated with the eigenvalue estimate $\hat{\lambda}_{d_m}$.

At each PIES iteration, (9.1) is augmented by multiplying the previous eigenvector approximation $Y_{m-1}\hat{\mathbf{x}}_{d_{m-1}}$ by $P_m(A - \zeta_m I)$, where $P_m \approx (A - \sigma_m I)^{-1}$. The scalars σ_m and ζ_m in $P_m(A - \zeta_m I)$ are selected in an attempt to emphasize the desired vector $\mathbf{x}_d$ in the new direction $P_m(A - \zeta_m I)Y_m\hat{\mathbf{x}}_{d_m}$. The specifics of $P_m(A - \zeta_m I)$ are considered in detail in Section 9.1.1. The choice of σ_m is closely related to the specification of the model-reduction interpolation point in Chapter 6.

9.1.1 Preconditioned matrices

The matrix $P_m(A - \zeta_m I)$ appeared throughout the model-reduction methods of the previous chapters, and it plays a central role in (9.1). As in model reduction, we begin by considering the case where P_m is an exact ES preconditioner, $(A - \sigma_m I)^{-1}$. The eigenvalues of $P_m(A - \zeta_m I)$ are then (recall Lemma 6.2),

$$\tilde{\lambda} = \frac{\lambda - \zeta_m}{\lambda - \sigma_m}. \tag{9.3}$$

In going from A to $P_m(A - \zeta_m I)$, λ is mapped to $\tilde{\lambda}$. Recall also from Section 6.1, that those eigenvalues of A that are mapped to well-separated positions in the spectrum of $P_m(A - \zeta_m I)$ converge rapidly. PIES attempt to choose σ_m and ζ_m that map λ_d to a well-separated position and that map the rest of the eigenvalues of A into a tight cluster. If this mapping is achieved, multiplication by $P_m(A - \zeta_m I)$ separates $\mathbf{x}_d$ from the rest of the cluster.

The parameter σ_m is always chosen to be an estimate of the desired eigenvalue in PIES. In this manner (recall the role of σ as an interpolation point), the portion of the complex plane around σ_m is emphasized. The parameter ζ_m is typically chosen to acquire one of the mappings in either Figure 6.1, the $\zeta_m = \infty$ case, or Figure 8.1, the $\zeta_m = \sigma_m$ case. The derivation of these mappings was already presented in model reduction. The $\zeta_m = \infty$ case relies on the properties of $(A - \sigma_m I)^{-1}$ to map the desired eigenvalue to the extreme outer edge of the spectrum. The $\zeta_m = \sigma_m$ case relies heavily on the properties of $(A - \sigma_m I)$ to map the desired eigenvalue towards the origin.

It is perhaps not surprising to learn that the choice of ζ_m does not matter when P_m is exactly $(A - \sigma_m I)^{-1}$. This result follows from Lemma 2.1 and is consistent with the independence of (2.20) on s. When $P_m = (A - \sigma_m I)^{-1}$, all mappings $A \to P_m(A - \zeta_m I)$ are equally good in terms of the resulting eigenvalue separation. However, ζ_m does become important when P_m is not exact. The choice $\zeta_m \approx \sigma_m$ is favored for exact P_m in the literature, apparently, because this mapping relies less on $(A - \sigma_m I)^{-1}$.

In many PIES, the scalars σ_m and ζ_m are updated every iteration to reflect the most recent approximations for λ_d. There is an advantage, though, to fixing these values and P_m at the start of the algorithm using some initial estimate for λ_d. In such an approach, the column space of Y becomes a Krylov subspace,

$$\operatorname{colsp}\{Y\} = \mathcal{K}(P(A - \zeta I), y_1),$$

and $P \approx (A - \sigma I)^{-1}$ need only be computed once. Symmetric Lanczos type methods can be used to compute Y [103, 104, 105]. Fortuitously, some form of the reduced-order pencil $(Y^T TAY, Y^T TY)$ is implicitly generated by an a Lanczos-type algorithm. In practice, the Lanczos method is restarted multiple times. The Lanczos method is executed for several iterations, new approximations to λ_d and $\mathbf{x}_d$ are found, the values of σ and ζ are updated, the starting vector is updated, and the process is repeated. In this manner, σ and ζ are occasionally updated. Although this infrequent update may slow convergence, the costs per iteration may drop in that a new P is not needed in every iteration.

9.1.2 Reduced-order pencils

Even though the matrix Y_m is fixed after m PIES iterations, the new eigenvector approximation $Y_m \hat{\mathbf{x}}_{d_m}$ is not. The vector $\hat{\mathbf{x}}_{d_m}$ depends on the choice for the transformation T_m in the low-order problem $(Y_m^T T_m A Y_m, Y_m^T T_m Y_m)$. It is stressed that the choice of T_m effects only the eigenvectors/values of the reduced-order approximation and not the eigenvectors/values of the original problem. The goal in choosing T_m is to find the best possible eigenvector approximation in $\operatorname{colsp}\{Y_m\}$ given a reasonable amount of computational effort. Several possibilities for T_m appear in the literature.

9.1.2.1 Strategy 1, $T_m = I$

This simple approach leads to the reduced-order pencil $(Y_m^T A Y_m, Y_m^T Y_m)$. This T_m is trivial to implement, making it a common choice in many existing implementations. However, for a given Y_m, only the exterior approximate eigenvalues/vectors of $(Y_m^T A Y_m, Y_m^T Y_m)$ tend to be optimal approximations for the corresponding eigenvalues of A [102]. Better approximations for an interior eigenvalue of A with a given Y_m may be achievable with a different T_m [106].

9.1.2.2 Strategy 2, $T_m = (A - \sigma_m I)^{-1}$

This choice for T_m is the one that leads to the best (in some sense) possible approximations for eigenvalues of A near (some possibly interior value) σ_m. This observation follows from the relations:

$$\begin{aligned}
&Y_m^T (A - \sigma_m I)^{-1} A Y_m \hat{\mathbf{x}} = \hat{\lambda} Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} \\
&\quad \Leftrightarrow Y_m^T (A - \sigma_m I)^{-1} (A - \sigma_m I + \sigma_m I) Y_m \hat{\mathbf{x}} = \hat{\lambda} Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} \\
&\quad \Leftrightarrow Y_m^T Y_m + \sigma_m Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} = \hat{\lambda} Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} \\
&\quad \Leftrightarrow Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} = (\hat{\lambda} - \sigma_m)^{-1} Y_m^T Y_m \hat{\mathbf{x}}.
\end{aligned} \tag{9.4}$$

The eigenvectors/values of $(Y^T (A - \sigma_m I)^{-1} A Y_m, Y_m^T (A - \sigma_m I)^{-1} Y_m)$ are directly related to the eigenvectors/values of $(Y^T (A - \sigma_m I)^{-1} Y_m, Y_m^T Y_m)$ by (9.4), but this latter pencil is simply Strategy 1 ($T = I$) applied to the matrix $(A - \sigma_m I)^{-1}$. The eigenvalues on the exterior of $(A - \sigma_m I)^{-1}$ correspond to those of A near σ_m, and these are the ones best approximated by $(Y^T (A - \sigma_m I)^{-1} Y_m, Y_m^T Y_m)$ with $T_m = I$.

The main shortcoming of this approach is that it requires an exact factorization of $(A - \sigma_m I)$ to acquire T_m. Although this cost is acceptable in methods already utilizing exact ES preconditioners, $P_m = (A - \sigma_m I)^{-1}$, it is not appropriate in general.

9.1.2.3 Strategy 3, $T_m = (A - \sigma_m I)$

This choice was proposed as a more efficient avenue for treating interior eigenvalues [107]. To evaluate this transformation, again rewrite it in terms of a second, equivalent eigenvalue problem. Define the quantities,

$$\begin{aligned} S^T S &= Y_m^T (A - \sigma_m I) Y_m, \\ \tilde{Y} &= (A - \sigma_m I) Y_m S, \\ \tilde{x} &= S \hat{x}. \end{aligned}$$

The reduced-order eigenvalue problem with $T_m = (A - \sigma_m I)$ can then be rewritten as

$$\begin{aligned} & Y_m^T (A - \sigma_m I) A Y_m \hat{\mathbf{x}} = \hat{\lambda} Y_m^T (A - \sigma_m I) Y_m \hat{\mathbf{x}} \\ & \quad \Leftrightarrow Y_m^T (A - \sigma_m I)^2 Y_m \hat{\mathbf{x}} = (\hat{\lambda} - \sigma_m) Y_m^T (A - \sigma_m I) Y_m \hat{\mathbf{x}} \\ & \quad \Leftrightarrow \tilde{\mathbf{x}} = (\hat{\lambda} - \sigma_m) S^{-T} Y_m^T (A - \sigma_m I)^{-1} Y_m S^{-1} \tilde{\mathbf{x}} \\ & \quad \Leftrightarrow \tilde{\mathbf{x}} = (\hat{\lambda} - \sigma_m) \tilde{Y}_m^T (A - \sigma_m I)^{-1} \tilde{Y}_m \tilde{\mathbf{x}} \\ & \quad \Leftrightarrow \tilde{Y}_m^T (A - \sigma_m I)^{-1} \tilde{Y}_m \tilde{\mathbf{x}} = (\hat{\lambda} - \sigma_m)^{-1} \tilde{\mathbf{x}}. \end{aligned}$$

Hence, an approximation found with Strategy 3 is equivalent to finding an approximation in the column space of $\tilde{Y}_m$ with Strategy 2. Strategy 3 finds the best (in a sense) possible eigenvector approximations in $\tilde{Y}_m$ for those eigenvalues of A near σ_m. Eigenvector approximations lying in $\tilde{Y}$ are frequently referred to as harmonic Ritz vectors [108].

This strategy finds eigenvector approximations in $\tilde{Y}_m$, a corrupted version of Y_m, and thus, the quality of the results may suffer. Strategy 2 is cheaper than Strategy 1, though, because exact factors or inverses need be treated here. Limited experimentation in the literature suggests that $T_m = (A - \sigma_m I)$ is preferred to $T = I$ for interior eigenvalues [107].

9.1.2.4 Strategy 4, $T_{m_l} = T_{m_r} = (A - \sigma_m I)^{-1}$

In this final approach, left and right transformations are utilized to obtain the reduced-order pencil,

$$(Y_m^T (A - \sigma_m I)^{-1} A (A - \sigma_m I)^{-1} Y_m, Y_m^T (A - \sigma_m I)^{-2} Y_m. \tag{9.5}$$

Thus, Strategy 4 can be thought of as a two-sided version of Strategy 2. Similar desirable approximations to eigenvalues of A near σ_m are therefore expected at the cost of involving

$(A - \sigma_m I)^{-1}$ again. These observations follow from the relations:

$$\begin{aligned}
&Y_m^T (A - \sigma_m I)^{-1} A (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} = \hat{\lambda} Y_m^T (A - \sigma_m I)^{-2} Y_m \hat{\mathbf{x}} \\
&\quad \Leftrightarrow Y_m^T (I + \sigma_m (A - \sigma_m I)^{-1})(A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} = \hat{\lambda} Y_m^T (A - \sigma_m I)^{-2} Y_m \hat{\mathbf{x}} \\
&\quad \Leftrightarrow Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}} = (\hat{\lambda} - \sigma_m) Y_m^T (A - \sigma_m I)^{-2} Y_m \hat{\mathbf{x}} \\
&\quad \Leftrightarrow Y_m^T (A - \sigma_m I)^{-2} Y_m \hat{\mathbf{x}} = (\hat{\lambda} - \sigma_m)^{-1} Y_m^T (A - \sigma_m I)^{-1} Y_m \hat{\mathbf{x}}.
\end{aligned}$$

The eigenvector approximations of Strategy 4 are those that would result from a straightforward restriction of $((A - \sigma_m I)^{-2}, (A - \sigma_m I)^{-1})$ by Y_m. Although not common in the existing eigenvalue literature, Strategy 4 becomes important in Section 9.2.

9.1.3 Existing implementations

A bedazzingly array of PIES now exists in the literature, yet most of them can be traced back to two techniques. One of these seminal techniques is the method of Davidson [98]. Davidson's method chooses $T_m = I$ (see Strategy 1 of Section 9.1.2) and $\zeta_m = \sigma_m$ (recall Section 9.1.1). The second technique is denoted, for lack of a consistent historical title, the shift-and-invert PIES. Originating in [103], the shift-and-invert PIES chooses $T_m = (A - \sigma_m I)^{-1}$ and $\zeta_m = \infty$. In its original form, shift-and-invert PIES kept σ_m fixed for multiple iterations to allow for the use of the symmetric Lanczos method.

A version of the shift-and-invert PIES is provided as Algorithm 9.1. This shift-and-invert Rayleigh-Ritz method is a relatively straightforward extension of the well-known shift-and-invert eigenvalue iteration [3]. However, rather than use only the most recent direction to approximate the desired eigenvector, shift-and-invert PIES incorporates all previously computed directions into the search subspace colsp$\{Y_m\}$. These computed directions are kept orthogonal by (S9.1.4) for improved numerical stability. The scalar σ_{m+1} is chosen to be the eigenvalue $\hat{\lambda}_{d_m}$ of $(Y_m^T (A - \sigma_m)^{-1} A Y_m, Y_m^T (A - \sigma_m)^{-1} Y_m)$. In practice, one actually computes the eigenvalue $\hat{\theta}_{d_m}$ of $Y_m^T (A - \sigma_m I)^{-1} Y_m$, which is largest in magnitude and uses the result (9.4) to obtain $\sigma_{m+1} = (\hat{\theta}_{d_m} - \sigma_m)^{-1}$.

If σ_{m+1} is kept fixed rather than updated in (S9.1.2), the symmetric Lanczos algorithm (Algorithm 2.1 with $v_m = w_m = y_m$ and $G = (A - \sigma I)^{-1}$) can be used to compute Y. Moreover, the tridiagonal matrix $Y^T (A - \sigma I)^{-1} Y$ arises naturally with the Lanczos

Algorithm 9.1 Shift-and-Invert PIES

Initialize: an orthogonal vector $y_1 = Y_1$ and eigenvalue guess σ_1;

For $m = 1$ to M,

(S9.1.1) Compute $(\hat{\theta}_{d_m}, \hat{\mathbf{x}}_{d_m})$ from the Rayleigh quotient $Y_m^T(A - \sigma_m I)^{-1} Y_m$;

(S9.1.2) $\sigma_{m+1} = \frac{1}{\hat{\theta}_{d_m} - \sigma_m}$;

(S9.1.3) $\tilde{y}_{m+1} = (A - \sigma_{m+1} I)^{-1}(Y_m \hat{\mathbf{x}}_{d_m})$;

(S9.1.4) $\hat{y}_{m+1} = \tilde{y}_{m+1} - Y_m Y_m^T \tilde{y}_{m+1}$;

(S9.1.5) $y_{m+1} = \frac{\hat{y}_{m+1}}{||\hat{y}_{m+1}||_2}$;

end

method. The largest eigenvalue of this tridiagonal matrix is $\hat{\theta}_{d_M}$, which in turn leads to the eigenvalue estimate, $\hat{\lambda}_{d_M} = (\hat{\theta}_{d_m} - \sigma)^{-1}$.

Because shift-and-invert PIES use both exact ES preconditioners and transformation Strategy 2 of Section 9.1.2, a rapid convergence to λ_d oftentimes occurs. Unfortunately, the involvement of $(A - \sigma_m I)^{-1}$ either explicitly or implicitly is not practical in many problems. For this reason, Davidson's method is the foundation of many currently popular PIES methods. Davidson's method and its many generalizations avoid the use of exact ES preconditioners.

A version of Davidson's method is provided as Algorithm 9.2. As noted above, this method utilizes the left transformation $T = I$ and sets ζ_m equal to σ_m. An orthogonal basis for the search subspace is formed in Y. The original method of Davidson chose the inverse of the diagonal of $(A - \sigma_m I)$ as the ES preconditioner P_m. Other, more recent extensions of the approach utilize more general ES preconditioners [19].

Difficulties arise with Davidson's method, however, if P_m becomes too good of an approximation to $(A - \sigma_m I)^{-1}$. In this case, $\tilde{y}_{m+1}$ in (S9.2.3) is approximately $\hat{\mathbf{x}}_{d_m}$ and no new direction is obtained. To avoid this difficulty, one can perturb ζ_m slightly away from σ_m. A popular choice in the literature [109] is to select $\zeta_{m+1} = \sigma_{m+1} + \delta_{m+1}$, where

Algorithm 9.2 Davidson's Method

Initialize: an orthogonal vector $y_1 = Y_1$;

For $m = 1$ to M,

(S9.2.1) Compute $(\hat{\lambda}_{d_m}, \hat{\mathbf{x}}_{d_m})$ from $Y_m^T A Y_m$;

(S9.2.2) $r_m = (A - \hat{\lambda}_{d_m} I)(Y_m \hat{\mathbf{x}}_{d_m})$;

(S9.2.3) $\tilde{y}_{m+1} = P_{m+1} r_m$;

(S9.2.4) $\hat{y}_{m+1} = \tilde{y}_{m+1} - Y_m Y_m^T \tilde{y}_{m+1}$;

(S9.2.5) $y_{m+1} = \frac{\tilde{y}_{m+1}}{||\tilde{y}_{m+1}||_2}$;

end

---

the perturbation is

$$\delta_{m+1} = \frac{\hat{\mathbf{x}}_{d_m}^T Y_m^T P_{m+1}(A - \sigma_{m+1} I) Y_m \hat{\mathbf{x}}_{d_m}}{\hat{\mathbf{x}}_{d_m}^T Y_m^T P_{m+1} Y_m \hat{\mathbf{x}}_{d_m}}. \tag{9.6}$$

Computing (9.6) directly is relatively cheap. An even more efficient implementation of this correction is possible through a method of Jacobi [110]. The perturbation in (9.6) leads to a new search direction, $P_{m+1}(A - (\sigma_{m+1} + \delta_{m+1})I)Y_m \hat{\mathbf{x}}_{d_m}$, which is orthogonal to the previous eigenvector estimate $Y_m \hat{\mathbf{x}}_{d_m}$. In this manner, new information is added to Y at every iteration.

Apparently, all variations of the original shift-and-invert and Davidson's methods can be categorized according to their choices for T_m and ζ_m. Table 9.1 attempts to sort methods contained in many (but certainly not all) papers accordingly. The point of this table is not to minimize the contributions of the listed papers; rather, it is to emphasize that central themes exist throughout the PIES literature involving T_m and ζ_m. The interested reader should examine these papers for their varied contributions concerning ES preconditioning, σ_m selection and eigenvalue convergence. Two recent surveys of iterative eigenvalue methods are also available [18, 111].

The papers listed in Table 9.1 tend to progress chronologically from left to right. It should be noted that the entry in the second row is complementary to that in the first and third rows. This difference follows from the fact that both the first row and the

Table 9.1: Classifying Existing PIES

	$\zeta_m = \infty$	$\zeta_m = \sigma_m$	$\zeta_m = \sigma_m + \delta_m$
$T = I$		[98, 19, 112, 113, 105]	[109, 114, 110]
$T_m = (A - \sigma_m I)^{-1}$	[103]		
$T_m = (A - \sigma_m I)$		[107]	[110]

second column of Table 9.1 tend to require exact ES preconditioners. As long as exact ES preconditioners are being computed, both the powerful transformation $T_m = (A - \sigma_m I)^{-1}$ and the easily computed case $\zeta_m = \infty$ should be used.

9.1.4 Approaches for several eigenvalues

Traditionally, several eigenvalues of (A, E) are computed via a block extension of an algorithm. One simply replaces the vector y_{m+1} in (9.1) with a block of vectors $y_{m+1} \in \mathbb{R}^{N \times K}$,

$$\tilde{y}_{m+1} = P_{m+1}(A - \zeta_{m+1} I) Y_m \left[\begin{array}{ccc} \hat{\mathbf{x}}_{d_m}^{(1)} & \ldots & \hat{\mathbf{x}}_{d_m}^{(K)} \end{array} \right]. \tag{9.7}$$

The matrix on the far right of (9.7) contains eigenvector estimates for the desired K eigenvalues. Alternatively, one can individually update each new direction, i.e.,

$$\tilde{y}_{m+1} = \left[\begin{array}{ccc} P_{m+1}^{(1)}(A - \zeta_{m+1}^{(1)} I) Y_m \hat{\mathbf{x}}_{d_m}^{(1)} & \ldots & P_{m+1}^{(K)}(A - \zeta_{m+1}^{(K)} I) Y_m \hat{\mathbf{x}}_{d_m}^{(K)} \end{array} \right],$$

and hopefully improve the resulting convergence. The parameters $P_{m+1}^{(k)}$ and $\zeta_{m+1}^{(k)}$ can be tuned to each individual eigendirection. Such an approach is developed for Davidson's method in [115].

Further interesting results are obtained if these $P_m^{(k)}$ are exact ES preconditioners, the parameters $\sigma_m^{(k)}$ and $\zeta_m^{(k)}$ are fixed for all m, and the initial eigenvector guesses $\hat{\mathrm{x}}_{d_1}^{(k)}$ are all identical. In this case, the column space of Y_m is an ever-popular rational Krylov subspace,

$$\bigcup_{k=1}^{K} \left((A - \sigma^{(k)} I)^{-1}, y_1 \right). \tag{9.8}$$

As stated earlier, the initial work on rational Krylov subspaces and the associated RA algorithm was directed at the eigenvalue problem [80]. The desired subspace can be constructed with a one-sided rational Arnoldi algorithm [91]. In this context, the rational Arnoldi algorithm serves as the multiple eigenvalue generalization of the shift-and-invert PIES. An algorithm for the one-sided rational Arnoldi algorithm is presented as Algorithm 9.3. This algorithm works to compute one eigenvalue approximation at a time (corresponding to the outer k loop) by constructing one Krylov subspace in (9.8) at a time. While converging to a given eigenvalue, the parameter $\sigma^{(k)}$ remains fixed so that an Arnoldi-type method results.

Note that the matrix E is included in Algorithm 9.3 for consistency with the RK algorithms in Chapter 4. For consistency with the simplifying assumptions made in this section, simply think of E as an identity and A as a symmetric matrix.

Algorithm 9.3 One-sided Rational Arnoldi

Initialize: $m = 1$ and an orthogonal vector y_1;

For $k = 1$ to K,

  Set $\sigma^{(k)}$ as an estimate for the next desired eigenvalue λ_{d_k};

  $q_{m+1} = Ey_1$;

  While λ_{d_k} is not found,

    (S9.3.1) $\tilde{y}_{m+1} = (A - \sigma^{(k)}E)^{-1}q_{m+1}$;

    (S9.3.2) $\hat{y}_{m+1} = \tilde{y}_{m+1} - \sum_{l=1}^{m} y_l\gamma_{l,m}$ where $\gamma_{l,m} = y_l^T\tilde{y}_{m+1}$;

    (S9.3.3) $y_{m+1} = \hat{y}_{m+1}/\gamma_{m+1,m}$ where $\gamma_{m+1,m} = \|\hat{y}_{m+1}\|_2$;

    (S9.3.4) $m = m + 1$;

    (S9.3.5) $q_{m+1} = Ey_m$;

  end

end

At this point, the casual reader may choose to proceed to Section 9.2. We spend the remainder of this section demonstrating how the one-sided RA algorithm can implicitly generate a pencil (Y^TTAY, Y^TTY) involving the Strategy 2 transformation $(A-\sigma^{(k)}I)^{-1}$. The presented approach for computing the approximate eigenvectors is a novel one (it differs from that proposed in [15, 91]), but is consistent with the eigenvalue approximations generated by the shift-and-invert PIE in the single eigenvalue case.

Algorithm 9.3 computes an orthogonal Y corresponding to the subspace in (9.8). However, Y must still be applied to some matrix pencil to determine approximate eigenvalues and eigenvectors in every step. Fortuitously, a desired low-order pencil is implicitly formed by the one-sided RA algorithm. This behavior and the following derivation is similar to that seen for the RL algorithm in Sections 4.1.3 and 4.1.4. Our starting point is the matrix relationship

$$AY_{m+1}\hat{E}_{m+1,m} = EY_{m+1}\hat{A}_{m+1,m}, \tag{9.9}$$

which was derived in [91] and holds after m iterations of the one-sided RA algorithm. Due to (S9.3.2) and (S9.3.3), the matrices $\hat{E}_{m+1,m}$ and $\hat{A}_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ are upper-Hessenberg matrices whose m^{th} columns take the forms

$$\begin{bmatrix} \gamma_{1,m} \\ \vdots \\ \gamma_{m+1,m} \\ 0 \end{bmatrix} \quad \text{and} \quad i_l + \sigma_m \begin{bmatrix} \gamma_{1,m} \\ \vdots \\ \gamma_{m+1,m} \\ 0 \end{bmatrix}. \tag{9.10}$$

The vector i_l is either the first or m^{th} column of an identity matrix (it does not matter in this discussion) and the scalar $\sigma_m \in \{\sigma^{(1)}, \ldots, \sigma^{(K)}\}$ is the specific interpolation point used in the m^{th} iteration. Note the similarity between (9.10) and (4.18). Defining $\tilde{A}_{m+1,m} = \hat{A}_{m+1,m} - \sigma_m\hat{E}_{m+1,m}$ and noting (9.9), results in

$$(A - \sigma_m E)Y_{m+1}\hat{E}_{m+1,m} = EY_{m+1}\tilde{A}_{m+1,m} = EY_m\tilde{A}_{m,m}. \tag{9.11}$$

The last equality in (9.11) holds, because the $(m+1)^{st}$ row of $\tilde{A}_{m+1,m}$ consists of zeros (the last column of the upper-Hessenberg $\tilde{A}_{m+1,m}$ is just i_m). Multiplying (9.11) on the left by $Y_m^T(A - \sigma_m E)^{-1}$ yields

$$\hat{E}_{m,m} = Y_m^T(A - \sigma_m E)^{-1}EY_m\tilde{A}_{m,m},$$

so that

$$\begin{aligned}
&Y_m^T(A-\sigma_m E)^{-1}(A-\hat{\lambda}E)Y_m\tilde{A}_m \\
&\quad = \; Y_m^T(A-\sigma_m E)^{-1}\{(A-\sigma_m E+(\sigma_m-\tilde{\lambda})E\}Y_m\tilde{A}_m && (9.12)\\
&\quad = \; (\sigma_m-\hat{\lambda})\hat{E}_m+\tilde{A}_m \\
&\quad = \; \hat{A}_m-\hat{\lambda}\hat{E}_m. && (9.13)
\end{aligned}$$

Thus, if $\hat{\lambda}$ and $\hat{\mathbf{x}}$ are an eigenvalue and eigenvector of $(\hat{A}_m, \hat{E}_m)$, then $(\hat{\lambda}, \tilde{A}_m\hat{\mathbf{x}})$ are an eigenvalue and eigenvector of

$$(Y_m^T(A-\sigma_m E)^{-1}AY_m, Y_m^T(A-\sigma_m E)^{-1}EY_m). \tag{9.14}$$

However, (9.14) is exactly the pencil sought by Strategy 1 of Section 9.1.2 for finding approximations to eigenvalues near σ_m. In the m^{th} iteration, the desired eigenvalue approximation $\hat{\lambda}_{d_m}$ nearest σ_m can be directly acquired from the matrices $\hat{A}_m$ and $\hat{E}_m$ constructed in Algorithm 9.3. This occurrence is ideal, because we are trying to locate the eigenvalue nearest σ_m in the m^{th} iteration. Only (9.14), the desired low-order pencil for σ_m, is implicitly generated at the m^{th} iteration, because doing so requires that the vector $(A-\sigma_m I)^{-1}y_m$ is somewhere available. This information is, in fact, available in the vector $\tilde{y}_{m+1}$, which was formed by multiplying y_m by $(A-\sigma_m I)^{-1}$.

The eigenvector approximation $Y_m\tilde{A}_m\hat{\mathbf{x}}_{d_m}$ arising from (9.14) is not the approximate eigenvector $Y_{m+1}\hat{E}_{m+1,m}\hat{\mathbf{x}}_{d_m}$ suggested by [91]. To provide further motivation for the former choice, we write the residual $\mathbf{r}_m$ corresponding to this eigenvector as

$$\begin{aligned}
\mathbf{r}_m &= (A-\hat{\lambda}_{d_m}E)Y_m\tilde{A}_m\hat{\mathbf{x}}_{d_m} \\
&= (A-\hat{\lambda}_{d_m}E)Y_{m+1}(\hat{A}_{m+1,m}-\sigma_m\hat{E}_{m+1,m})\hat{\mathbf{x}}_{d_m} \\
&= (A-\sigma_m E)Y_{m+1}(\hat{A}_{m+1,m}-\sigma_m\hat{E}_{m+1,m})\hat{\mathbf{x}}_{d_m} \\
&\quad +(\sigma_m-\hat{\lambda}_{d_m})EY_{m+1}(\hat{A}_{m+1,m}-\sigma_m\hat{E}_{m+1,m})\hat{\mathbf{x}}_{d_m},
\end{aligned}$$

and continue by noting (9.9) and writing

$$
\begin{aligned}
\mathbf{r}_m &= (A - \sigma_m E) Y_{m+1} (\hat{A}_{m+1,m} - \sigma_m \hat{E}_{m+1,m}) \hat{\mathbf{x}}_{d_m} \\
&\quad + (\sigma_m - \hat{\lambda}_{d_m})(A - \sigma_m E) Y_{m+1} \hat{E}_{m+1,m} \hat{\mathbf{x}}_{d_m} \\
&= (A - \sigma_m E) Y_{m+1} (\hat{A}_{m+1,m} - \hat{\lambda}_{d_m} \hat{E}_{m+1,m}) \hat{\mathbf{x}}_{d_m}.
\end{aligned}
$$

Lastly, the fact that $(\hat{A}_{m,m} - \lambda_{d_m} \hat{E}_{m,m}) \hat{\mathbf{x}}_{d_m}$ equals zero yields the residual expression

$$
\mathbf{r}_m = \alpha (A - \sigma_m E) y_{m+1}, \tag{9.15}
$$

where α is the dot-product of $\hat{\mathbf{x}}_{d_m}$ with the last row of $(\hat{A}_{m+1,m} - \hat{\lambda}_{d_m} \hat{E}_{m+1,m})$. The residual (9.15) that results from choosing the approximate eigenvector as $Y_m \tilde{A}_m \hat{\mathbf{x}}_{d_m}$ in the m^{th} iteration is $(A - \sigma_m E)$ orthogonal to the column space of Y_m. This orthogonality is consistent with the Galerkin condition in (9.2). The approximate eigenvector choice $Y_{m+1} \hat{E}_{m+1,m} \hat{\mathbf{x}}_{d_m}$ only leads to E orthogonality for the residual [91].

9.2 Arriving at PIES from Model Reduction

Many previous chapters allude to connections between the proposed model-reduction techniques and the eigenvalue methods of Section 9.1. In this section, the techniques developed for model reduction are considered in the context of finding eigenvalues. Variations on the themes in Section 9.1 result. By performing this exercise, it is hoped that new insights into and perhaps modifications of existing PIES result can eventually be obtained.

For simplicity, a model-reduction problem that is consistent with the eigenvalue assumptions (A is symmetric, E is the identity matrix, and $b = c$) is considered. This assumption is not restrictive, because the model-reduction techniques that serve as our starting point are already known for the general case.

The key to both model reduction and PIES is the form of the search subspaces. For the symmetric case, the model-reduction subspace is the column space of V. In model reduction, the next direction in the search subspace is computed as $\Phi_m (A - \zeta_m I) v_{m-1}$, while the new direction in PIES is $\Phi_m (A - \zeta_m I) Y_{m-1} \hat{\mathrm{x}}_{d_{m-1}}$. Section 9.1.3 notes that these new directions take equivalent forms when the Φ_m are constant with respect to a given interpolation point, e.g., Krylov subspaces result in (9.8) for the shift-and-invert PIES. However, what happens when Φ_m is inexact and varying? In particular, how does Davidson's method relate to model reduction? The answer to these questions come from Section 8.3, where relations between outer modeling and inner solver iterations were discussed. Recall the update direction in (8.16) that was derived for model reduction,

$$\Phi_{m+1}\left\{v_{p_m} - (A-\sigma_{m+1}I)V_m(\hat{A}_m - \sigma_{m+1}I_m)^{-1}V_m^T v_{p_m}\right\}. \tag{9.16}$$

This vector is the update direction that results when the outer iteration is used to generate an initial guess for the inner solve. However, as σ_{m+1} approaches the eigenvalue $\hat{\lambda}_{d_m}$ of $\hat{A}$ (they are set equal in Davidson's method), the vector $(\hat{A}_m - \sigma_{m+1}I_m)^{-1}V_m^T v_{p_m}$ in (9.16) becomes the orthogonal eigenvector $\hat{\mathbf{x}}_{d_m}$ times a large scaling factor. Due to this large scaling of $\hat{\mathbf{x}}_{d_m}$, the $\Phi_{m+1}v_{p_m}$ term in (9.16) drops out and one is left with the new direction of Davidson's method. Hence, Davidson's method is an inner-outer type iterative method that implicitly computes $\Phi_m(A-\zeta_m I)v_{m-1}$ by utilizing an initial solver guess that is based on the outer iteration. The only problem that can arise in Davidson's method is when Φ_{m+1} becomes exact (recall Section 9.1.2). In this case, the vector $\Phi_{m+1}v_{p_m}$ yields the ideal new direction in (9.16); yet it is dropped in Davidson's method. Corrections to this problem utilize orthogonality between the existing outer subspace and the computed new direction to avoid difficulties [109, 110]. As noted at the end of Section 8.3, this approach is another way of relating the inner and outer iterations.

The only other point of difference between the model reduction and eigenvalue subspaces is the choice for the first vector in the subspace. PIES typically choose some vector, say b, for its starting vector y_1. The first column of V in model reduction is $\Phi_1 b$. Think of v_1 as simply an improved guess for the starting vector; $(A-\sigma_1 E)^{-1}b$ is more likely a better approximation for the desired eigenvector then b. If this supposedly better starting guess is used for y_1, then (assuming fixed shift selection) the eigenvalues of (Y^TAY, Y^TY) and (V^TAV, V^TV) are identical. There is an alternative description of the differences between V and Y, however. Consider the simplest case, where the column space of V is $\mathcal{K}_M((A-\sigma I)^{-1}, (A-\sigma I)^{-1}b)$ and the column space of Y is $\mathcal{K}_M((A-\sigma I)^{-1}, b)$.

Then, trivially, V equals $(A - \sigma I)^{-1}Y$ and the low-order pencil of model reduction is

$$V^T(A - \lambda I)V = Y^T(A - \sigma I)^{-1}(A - \lambda I)(A - \sigma I)^{-1}Y. \tag{9.17}$$

The right side of (9.17) and thus, the low-order pencil from model-reduction result from the Strategy 4 approach of Section 9.1.2 for generating the reduced-order pencil. The expression (9.17) does not hold exactly in the event of inexact ES preconditioners. It does illustrate that only minor differences exist between V and Y and, perhaps, the different strategies of Section 9.1.2.

CHAPTER 10

CONCLUSIONS AND FUTURE WORK

In this final chapter, the results of the previous nine chapters are recapitulated with an emphasis on DS preconditioners and bases for the projection subspaces. Suggestions are also provided for further improvements and extensions of these results.

10.1 Summary of Results

Rational interpolation is readily achieved by projection onto unions of Krylov subspaces as in (3.1) and (3.2). A projection algorithm for implementing rational interpolation must therefore choose both the DS preconditioners used in (approximately) forming the subspaces and the specific bases representing the subspaces. Modulo some assumptions on the conditioning of the reduced-order pencil $Z^T(A-\sigma E)V$, any bases will lead to rational interpolation. Decisions to form biorthogonal or orthogonal bases are only pertinent with respect to efficiency and reliability, but not to rational interpolation itself. Biorthogonality, with respect to a certain matrix, leads to the rational Lanczos algorithm, an efficient approach that generalizes all existing single-point Lanczos techniques for Padé approximation. In theory, this biorthogonality can be maintained and the projection implicitly performed with recursions whose lengths are proportional to the number of interpolation points. Unfortunately, this biorthogonality is lost in practice, the convergence of the RL results are slightly slowed, and the algorithm is susceptible to errors in the DS preconditioner. These difficulties can be avoided by utilizing orthogonal bases. This choice leads to the dual rational Arnoldi algorithm, a relatively robust, but more costly approach. The dual RA approach is suited for inexact DS preconditioners. Its orthogonality tends to eliminate column dependencies in V or Z, which otherwise arise in finite precision and, in particular, when many moments are matched about a single interpolation point. A novel approach based on a low-order singular value decomposition can alternatively be employed to extract dependent columns. This technique, associated with the proposed rational power method, is applied to the reduced-order model and avoids costly Gram-Schmidt computations on the vectors in the large-dimensional subspaces. By avoiding any form of (bi)orthogonalization, the RP algorithm is particularly well-suited for parallel implementations. In particular, if Z is known a priori (the approach is one-sided), then V and the reduced-order model can be concurrently computed on multiple processors with nearly negligible communications. Error estimates can be used to avoid redundant work on these processors.

In addition to the type of basis, the DS preconditioners must be selected. Ideally, these DS preconditioners are exact inverses of $(A - \sigma^{(k)}E)$, leading to moment matching at the interpolation point $\sigma^{(k)}$. However, approximate DS preconditioners (solves) are reasonable in versions of the RA and RP algorithms and may reduce computational costs. Experiments suggest that the accuracy of the approximate solves should be consistent with that desired in the reduced-order model, if significant deterioration in the modeling convergence is to be avoided. These approximations can be implemented via fixed preconditioners, inner iterations of a Krylov-based iterative solver, or a combination of inner iterations and inner preconditioners. Injecting information from the outer modeling iteration into the solver can also improve the solver's performance. Regardless of how the DS preconditioner is formed, its contents depend on the choice for the interpolation point. Imaginary interpolation points, $\sigma^{(k)}$, lead to reduced-order model convergence at frequencies in the locality of $|\sigma^{(k)}|$. Real interpolation points, on the other hand, lead to courser convergence over broader frequency ranges. In terms of cost versus accuracy trade-offs, a limited number of real interpolation points are typically preferred in sequential rational Krylov implementations. For purposes of load balancing, parallel implementations are better served by numerous, dynamically placed imaginary interpolation points. In either implementation, the placement of interpolation points and/or the number of moments matched per point should be based on available information feedback from error analyses.

The error can be implicitly estimated from residual computations or explicitly estimated by comparisons of different reduced-order models (whose differences are themselves due to complementary choices of past interpolation points). The residuals are at times easier to compute, but the model comparison tends to be more accurate.

10.2 Future Possibilities

The aim of this dissertation is to provide a strong foundation for the reduced-order modeling of LTI, SISO dynamic systems with Krylov-like projection methods. Beyond this work, there are possible tunings based on problem dependent issues, e.g., taking advantage of symmetry, which can be addressed with simple modifications of the methods in this dissertation. There are also implementational details, e.g., tuning an adaptive interpolation point strategy, which can be addressed with coded trials and errors. From a purer research standpoint, there are five remaining areas for future work that are apparent at this time.

10.2.1 MIMO systems

As noted in Section 3.4.2, multiple-input multiple-output systems are common and frequently treated through block versions of the projection algorithms. However, block versions become expensive for even moderate numbers of inputs and outputs. New ideas for acquiring the fundamental directions in the subspaces (3.18) and (3.19) are needed. Perhaps certain poles of the system should be emphasized in a modal-type approach.

10.2.2 Rank deficiencies

Section 4.1.1 proposes an SVD approach for handling rank deficient V or Z matrices by postprocessing the reduced-order model. Further theoretical work and experimentation are needed to completely understand the details and reliability of this method. In particular, choosing the frequency (or perhaps frequencies) at which the SVD (or SVD's) is evaluated requires additional insight. Furthermore, acceptable condition numbers for the postprocessed, reduced-order model must be examined.

10.2.3 Multilevel parallelism

A C-version of the parallel Algorithm 7.1 is being planned. It will be implemented for distributed architectures via the MPI (message passing interface) directives. However, in extremely large problems, it is doubtful that a matrix pencil $(A - \sigma^{(k)}E)$ can be stored/treated by a single processor unit. A second level of parallelism that is based on partitioning the basic matrix operations will be required. In this manner, hundreds of processors can be brought to bear on the problem, while hopefully retaining efficiency.

10.2.4 Approximate solvers

There are nearly a countless number of iterative solvers and inner preconditioners that can be considered for approximately solving (8.3). For simplicity, the well-known GMRES algorithm is utilized in Chapter 8. The risks and benefits of a two-sided iterative solver, such as QMR, should be studied as well. Furthermore, connections between the outer modeling and inner solver recursions appear to be important and should be considered in detail. Lastly, connections should be obtained between the approximate solves and interpolation point placement. It is conjectured that the use of approximate solves makes an increasing number of interpolation points more feasible in sequential implementations. This statement was not explored or exploited in Chapter 8.

10.2.5 Related problems

Beyond the issues arising directly from this dissertation, there are several classes of problems that can possibly benefit from the developed LTI model-reduction techniques:

1. *Shifted Systems of Linear Equations.* Many of the model-reduction techniques, e.g., interpolation point placement based on the residual calculations, the various

RK implementations, parallelism, etc., should be directly transferable to solving arbitrary systems of shifted linear equations.

2. *Eigenvalue Problems.* The eigenvalue problem was discussed in detail in Chapter 9 along with its connections to model reduction. General approaches for computing several eigenvalues over regions are still needed. Issues such as interpolation point placement and parallelism need to be further explored as well.

3. *Nonlinear or Time-varying Problems.* Extending the model reduction beyond LTI dynamic systems provides several new challenges. For distributed systems, e.g., transmission line equations, the higher order moments of the system will generally no longer be available via simple repetitive multiplications by a matrix. Also, if the linear pencil $(A - sE)$ is replaced by a higher (perhaps infinite) order matrix polynomial $\mathcal{A}(s)$, then explicitly computing the approximation $Z^T \mathcal{A}(s) V$ is no longer straightforward. One might linearize the model-reduction process over frequency regions and attempt to merge the results. Similarly, in time-dependent problems, LTI reduced-order models might be constructed for different regions of time. The sensitivity of a given model's accuracy, with respect to time, must then be determined and approaches for updating the projection matrices V and Z over time must be obtained.

APPENDIX A

LEMMA PROOFS

Proof of Lemma 2.1 The result follows from the equalities

$$\begin{aligned}(A-\sigma E)^{-1}(A-sE) &= (A-\sigma E)^{-1}(A-sE+(\sigma-\sigma)E)\\ &= I+(\sigma-s)(A-\sigma E)^{-1}E. \quad \blacksquare\end{aligned}$$

Proof for Lemma 2.2 We show that $\mathcal{K}_j(\eta G+I,g)\subseteq\mathcal{K}_j(G,g)$ by induction. The dual relation $\mathcal{K}_j(G,g)\subseteq\mathcal{K}_j(\eta G+I,g)$ follows in a similar fashion. The subspaces $\mathcal{K}_j(\eta G+I,g)$ and $\mathcal{K}_j(G,g)$ are trivially identical when $j=1$. Now assume that (2.19) holds for some $\tilde{j}\geq 1$. It must be shown that $(\eta G+I)^{\tilde{j}}g\in\mathcal{K}_{\tilde{j}+1}(G,g)$. By assumption, $\tilde{g}=(\eta G+I)^{\tilde{j}-1}g$ is in $\mathcal{K}_{\tilde{j}}(G,g)$ and therefore, $G\tilde{g}$ is in $\mathcal{K}_{\tilde{j}+1}(G,g)$. Thus, the vector

$$(\eta G+I)^{\tilde{j}}g=(\eta G+I)\tilde{g}=\eta G\tilde{g}+\tilde{g}$$

lies in $\mathcal{K}_{\tilde{j}+1}(G,g)$ and the first half of the proof is complete. $\blacksquare$

Proof for Lemma 3.1 The assumption that v lies in the column space of V implies that there exists a vector g such that $v=Vg$. Noting the biorthogonality of V and W, it follows that $VW^Tv=VW^TVg=Vg=v$. $\blacksquare$

Proof for Lemma 3.2 We begin by defining a matrix

$$W^T=\left(Z^T(A-\sigma E)V\right)^{-1}Z^T(A-\sigma E) \tag{A.1}$$

which satisfies $W^TV=I$. The proof follows by induction. For $j=1$,

$$V\left(Z^T(A-\sigma E)V\right)^{-1}Z^Tb=VW^T(A-\sigma E)^{-1}b=(A-\sigma E)^{-1}b.$$

The last equality follows from Lemma 3.1 because $(A-\sigma E)^{-1}b \in \text{colsp}\{V\}$. If we now assume that the desired result holds up to some $j \le J_b$, then the expression of interest corresponding to j is

$$\begin{aligned}
&V\left\{(Z^T(A-\sigma E)V)^{-1}Z^TEV\right\}^{j-1}(Z^T(A-\sigma E)V)^{-1}Z^Tb \\
&= V(Z^T(A-\sigma E)^TV)^{-1}Z^TE\left\{(A-\sigma E)^{-1}E\right\}^{j-2}(A-\sigma E)^{-1}b \\
&= V(Z^T(A-\sigma E)^TV)^{-1}Z^T(A-\sigma E)\left\{(A-\sigma E)^{-1}E\right\}^{j-1}(A-\sigma E)^{-1}b \\
&= VW^T\left\{(A-\sigma E)^{-1}E\right\}^{j-1}(A-\sigma E)^{-1}b \\
&= \left\{(A-\sigma E)^{-1}E\right\}^{j-1}(A-\sigma E)^{-1}b.
\end{aligned}$$

The first equality follows from the inductive assumption. The last equality follows from Lemma 3.1 because $\left\{(A-\sigma E)^{-1}E\right\}^{j-1}(A-\sigma E)^{-1}b \in \text{colsp}\{V\}$. By induction, the desired result must hold for any $j \le J_b$. ■

Proof for Lemma 3.3 The desired result is simply the dual to that in Lemma 3.2. For this reason, we only present the $j=1$ case and note that the balance of the argument proceeds in the fashion of the previous proof. Define W again as in (A.1). Then, for $j=1$,

$$c^TV\left(Z^T(A-\sigma E)V\right)^{-1}Z^T = c^TVW^T(A-\sigma E)^{-1} = c^T(A-\sigma E)^{-1}.$$

The last equality follows from the use of Lemma 3.1 in conjunction with the following observation: if $W = (A-\sigma E)^TZT$ for some nonsingular matrix $T \in \mathbb{R}^{M\times M}$ and a matrix $Z \in \mathbb{R}^{N\times M}$ satisfying $\mathcal{K}_J\left((A-\sigma E)^{-T}E^{-T}, (A-\sigma E)^{-T}c\right) \subseteq \text{colsp}\{Z\}$, then $\mathcal{K}_J\left(E^T(A-\sigma E)^{-T}, c\right) \subseteq \text{colsp}\{W\}$.

To see this observation, begin by noting that

$$\tilde{z}_j = \left\{(A-\sigma E)^{-T}E^T\right\}^{j-1}(A-\sigma E)^{-T}c^T \subseteq \text{colsp}\{Z\} \tag{A.2}$$

is assumed for $j=1,\ldots,J$. Thus, $\tilde{z}_j$ can be written as Zg_j for some vector g_j and any $j \le J$. Multiplying $\tilde{z}_j$ on the left by $(A-\sigma E)^T$ yields

$$(A-\sigma E)^T\tilde{z}_j = (A-\sigma E)^TZTT^{-1}g_j = WT^{-1}g_j = W\tilde{g}_j \tag{A.3}$$

for $j = 1, \ldots, J$. By (A.2) and (A.3), $\left\{E^{-T}(A-\sigma E)^{-T}\right\}^{j-1} c$ is in the column space of W for $j = 1, \ldots, J$. A basis for $\mathcal{K}_J(E^T(A-\sigma E)^{-T}, c)$ and hence, $\mathcal{K}_J(E^T(A-\sigma E)^{-T}, c)$ itself must therefore lie in colsp $\{W\}$. ■

Proof for Lemma 4.1 We prove (4.1). The key is to note that $(A-\sigma^{(k)}E)^{-1}$ can be rewritten as

$$\begin{aligned}(A-\sigma^{(k)}E)^{-1} &= (A-\sigma^{(k)}E)^{-1}(A-\sigma^{(k+1)}E)(A-\sigma^{(k+1)}E)^{-1}\\ &= (A-\sigma^{(k)}E)^{-1}(A-\sigma^{(k)}E+(\sigma^{(k)}-\sigma^{(k+1)})E)(A-\sigma^{(k+1)}E)^{-1}\end{aligned}$$

to yield

$$(\sigma^{(k)}-\sigma^{(k+1)})(A-\sigma^{(k)}E)^{-1}E(A-\sigma^{(k+1)}E)^{-1} = (A-\sigma^{(k)}E)^{-1}-(A-\sigma^{(k+1)}E)^{-1}. \quad \text{(A.4)}$$

Using (A.4), expression (4.1) follows via induction. If $j = 1$, multiplying (A.4) on the right by b gives

$$(A-\sigma^{(k)}E)^{-1}E(A-\sigma^{(k+1)}E)^{-1}b = (\sigma^{(k)}-\sigma^{(k+1)})^{-1}\{(A-\sigma^{(k)}E)^{-1}-(A-\sigma^{(k+1)}E)^{-1}\}b$$

and (4.1) is satisfied. Next, assume that (4.1) holds for $j = 1, \ldots, J-1$. Multiplying (A.4) on the right by $E\left\{(A-\sigma^{(k+1)}E)^{-1}E\right\}^{J-2}(A-\sigma^{(k+1)}E)^{-1}b$ yields

$$\begin{aligned}&(\sigma^{(k)}-\sigma^{(k+1)})(A-\sigma^{(k)}E)^{-1}E\left\{(A-\sigma^{(k+1)}E)^{-1}E\right\}^{J-1}(A-\sigma^{(k+1)}E)^{-1}b\\ &= (A-\sigma^{(k)}E)^{-1}E\left\{(A-\sigma^{(k+1)}E)^{-1}E\right\}^{J-2}(A-\sigma^{(k+1)}E)^{-1}b\\ &\quad -\left\{(A-\sigma^{(k+1)}E)^{-1}E\right\}^{J-1}(A-\sigma^{(k+1)}E)^{-1}b. \qquad \text{(A.5)}\end{aligned}$$

Under the assumption that (4.1) holds for $j = J-1$, (A.5) shows that (4.1) also holds for $j = J$. The induction step and thus (4.1) hold, in general. The proof of (4.2) is the dual to that provided for (4.1). ■

Proof for Lemma 5.1 We prove (5.6) and leave (5.5) as its dual. Combining the right sides of (S4.1.2) and (S4.1.3) of the general RK algorithm leads to the relation

$$(A-\sigma_m E)^T Z_m \begin{bmatrix} \bar{z}_m \\ \beta_m^z \end{bmatrix} = w_{p_m+1}.$$

Through the right sides of (S4.1.4) and (S4.1.5) this relation can be rewritten as

$$
\begin{aligned}
(A-sE)^T Z_m \begin{bmatrix} \bar{z}_m \\ \beta_m^z \end{bmatrix} &= w_{p_m+1} + (\sigma_m - s) E Z_m \begin{bmatrix} \bar{z}_m \\ \beta_m^z \end{bmatrix} \\
&= w_{p_m+1} + (\sigma_m - s) W_{m+1} \begin{bmatrix} \bar{w}_2 & \cdots & \uparrow \\ \beta_2^w & & \bar{w}_{m+1} \\ & \ddots & \downarrow \\ & & \beta_{m+1}^w \end{bmatrix} \begin{bmatrix} \bar{z}_m \\ \beta_m^z \end{bmatrix}. \qquad \text{(A.6)}
\end{aligned}
$$

Combining the expressions (A.6) into matrix form for $m = 1$ to M and multiplying on the right by the inverse of

$$
\begin{bmatrix} \beta_1 & \bar{z}_2 & \cdots & \uparrow \\ & \ddots & & \bar{z}_m \\ & & \ddots & \downarrow \\ & & & \beta_m^z \end{bmatrix}
$$

yields (5.6). ■

Proof for Lemma 6.1 If λ_n is an eigenvalue of (A, E), then $A\mathbf{x}_n = \lambda E\mathbf{x}_n$. An equivalent relation, $(A-\sigma E)^{-1}E\mathbf{x}_n = \frac{1}{\lambda_n - \sigma}\mathbf{x}_n$, follows from Lemma 2.1. Because the original system is stable, the spectrum of (A, E) lies to the left of the imaginary axis and the spectrum of $(A-\sigma)^{-1}E$ is bounded in the complex plane by the complex function $f(\omega) = \frac{1}{\iota\omega - \sigma}$. However, $f(\omega)$ simply defines a circle centered at $\frac{-1}{2\sigma}$ with radius $\frac{1}{2\sigma}$, because the relation

$$
\left(\frac{\sigma}{\sigma^{(2)} + \omega^2} - \frac{1}{2\sigma}\right)^2 + \left(\frac{\omega}{\sigma^{(2)} + \omega^2}\right)^2 = \left(\frac{1}{2\sigma}\right)^2
$$

holds where $\mathrm{Real}(f(\omega)) = \frac{-\sigma}{\sigma^{(2)} + \omega^2}$ and $\mathrm{Imag}(f(\omega)) = \frac{-\omega}{\sigma^{(2)} + \omega^2}$. ■

Proof for Lemma 6.2 The desired result follows from the relations

$$
\begin{aligned}
A\mathbf{x}_n = \lambda_n E\mathbf{x}_n \;&\leftrightarrow\; (A-\sigma E)^{-1}E\mathbf{x}_n = \mathbf{x}_n(\lambda_n - \sigma)^{-1} \\
&\leftrightarrow\; (I + (\sigma - s)(A-\sigma E)^{-1}E)\mathbf{x}_n = \left(1 + \frac{\sigma - s}{\lambda_n - \sigma}\right)\mathbf{x}_n \\
&\leftrightarrow\; (A-\sigma E)^{-1}\{(A-\sigma E) + (\sigma - s)E\}\mathbf{x}_n = \frac{\lambda_n - s}{\lambda_n - \sigma}\mathbf{x}_n. \quad ■
\end{aligned}
$$

APPENDIX B

SELECTED MATLAB IMPLEMENTATIONS

The development of the dual rational Arnoldi and rational Lanczos algorithms are important contributions of this work. Provided below are the implementations of these algorithms that were executed in MATLAB for Examples 4.5 and 7.1.

A Dual Rational Arnoldi Implementation:

```
function [Am,Em,bm,cm] = RK_DRA(A,E,b,c,J,S);
% Rational Krylov Method (Dual Rational Arnoldi Version)
%
% INPUTS:
%        A,E,b,c = system to be modeled
%        J = number of moments to be matched per point
%        S = column vector of interpolation points
% OUTPUTS:
%        Am,Em,bm,cm = reduced-order model

% Parameter initialization
m = 1;
J = round(J/2);
[K,one] = size(S);
[N,N] = size(A);
V = []; Z = [];
% Factorize sparse matrices via minimal column ordering
```

```
L = []; U = []; P = []; Q = [];
p = zeros(K,N);
for k=1:K,
  X = (A-S(k)*E);
  p(k,:) = colmmd(X);
  [Lt,Ut,Pt] = lu(X(:,p(k,:)));
  L = [L,Lt]; U = [U,Ut]; P = [P,Pt];
  I = sparse(eye(N,N)); Q = [Q,I(:,p(k,:))];
end
% Construct V and Z
for j=1:J,
  for k=1:K,
    kk = (k-1)*N+1:k*N;
    if j==1,
      v = Q(:,kk)*(U(:,kk)\(L(:,kk)\(P(:,kk)*b)));
      z = P(:,kk)'*(L(:,kk)'\(U(:,kk)'\(Q(:,kk)'*c)));
    else
      v = Q(:,kk)*(U(:,kk)\(L(:,kk)\(P(:,kk)*(E*V(:,m-1)))));
      z = P(:,kk)'*(L(:,kk)'\(U(:,kk)'\(Q(:,kk)'*(E'*Z(:,m-1)))));
    end
    if m > 1, v = v - V*(V'*v); z = z - Z*(Z'*z); end
    V = [V,v/norm(v)];  Z = [Z,z/norm(z)];
    m = m+1;
  end
end
% Compute the reduced-order model
Am = Z'*(A*V); Em = Z'*(E*V); bm = Z'*b; cm = V'*c;
```

A Rational Lanczos Implementation:

```
function [Am,Em,bm,cm] = RK_RL(A,E,b,c,J,S);
% Rational Krylov Method (Rational Lanczos Version)
%
% INPUTS:
%       A,E,b,c = system to be modeled
%       J = number of moments to be matched per point
%       S = column vector of interpolation points
% OUTPUTS:
%       Am,Em,bm,cm = reduced-order model

% Parameter initialization
m = 1;
J = round(J/2);
[K,one] = size(S);
[N,N] = size(A);
V = []; W = []; w = c;
% Reduced-order model initialization
Am=zeros(J*K,J*K); Em=zeros(J*K,J*K); bm=zeros(J*K,1); cm=zeros(J*K,1);
% Factorize sparse matrices via minimal column ordering
L = []; U = []; P = []; Q = [];
p = zeros(K,N);
for k=1:K,
  X = (A-S(k)*E);
  p(k,:) = colmmd(X);
  [Lt,Ut,Pt] = lu(X(:,p(k,:)));
  L = [L,Lt]; U = [U,Ut]; P = [P,Pt];
  I = sparse(eye(N,N)); Q = [Q,I(:,p(k,:))];
end
```

```
% Construct V and W
for j=1:J,
  for k=1:K,
    kk = (k-1)*N+1:k*N;
    if j==1,
      v = Q(:,kk)*(U(:,kk)\(L(:,kk)\(P(:,kk)*b)));
    else
      v = Q(:,kk)*(U(:,kk)\(L(:,kk)\(P(:,kk)*(E*V(:,m-1)))));
    end
    if m > 1,
      Gamma = W(:,max(1,m-K-1):m-1)'*v;
      Beta  = V(:,max(1,m-K-1):m-1)'*w;
      v = v - V(:,max(1,m-K-1):m-1)*Gamma;
      w = w - W(:,max(1,m-K-1):m-1)*Beta;
      v = v-V(:,max(1,m-K-1):m-1)*(W(:,max(1,m-K-1):m-1)'*v);
      w = w-W(:,max(1,m-K-1):m-1)*(V(:,max(1,m-K-1):m-1)'*w);
    end
    gamma = sqrt(abs(w'*v)*norm(v)/norm(w));
    beta  = sign(w'*v)*sqrt(abs(w'*v)*norm(w)/norm(v));
    if m == 1, beta_1 = beta; gamma_1 = gamma; end
    V = [V,v/gamma];  W = [W,w/beta];
    w = E'*(P(:,kk)'*(L(:,kk)'\(U(:,kk)'\(Q(:,kk)'*W(:,m)))));
    if m > 1,
      Em(m-1,max(1,m-K-1):m) = [Beta',beta];
      Am(m-1,:) = S_old*Em(m-1,:);
      Am(m-1,m-1) = Am(m-1,m-1) + 1;
    end
    S_old = S(k);
    m = m+1;
```

```
  end
end
% Finish off the Am and Em matrices
Beta  = V(:,max(1,m-K-1):m-1)'*w;
Em(J*K,J*K-K:J*K) = Beta';
Am(J*K,:) = S(K)*Em(J*K,:);
Am(J*K,J*K) = Am(J*K,J*K)+1;
% Compute the reduced-order input and output vectors
bm = (Am(:,1)-S(1)*Em(:,1))*gamma_1;
cm(1) = beta_1;
```

A Quasi-Parallel Rational Power Implementation:

```
function [Am,Em,bm,cm] = RK_PRP(A,E,b,c,J,f);
% Rational Krylov Method (Quasi-Parallel Rational Power Version)
%
% INPUTS:
%       A,E,b,c = system to be modeled
%       J = number of ``parallel'' iterations to perform
%       f = vector containing frequency point grid
% OUTPUTS:
%       Am,Em,bm,cm = reduced-order model

nprocs = 8;

% initialize the pdf to a uniform distribution
[pt_tot,one] = size(f);
pdf = ones(size(f))/pt_tot;
pt_sep = round(0.01*pt_tot)+1;
% initialize the projection matrices
```

```
V1 =[]; Z1=[]; V2=[]; Z2=[];

for j=1:J,
  for p=1:nprocs,

    % choose another point for model 1 based on current pdf
    rvar = rand(1);
    pt_cnt = 1;
    PDF = pdf(1);
    while rvar > PDF,
      pt_cnt = pt_cnt+1;
      PDF = PDF + pdf(pt_cnt);
    end
    % edit pdf to keep current points 1% distanced
    for l=max(1,pt_cnt-pt_sep):min(pt_tot,pt_cnt+pt_sep), pdf(l)=0; end
    pdf = pdf/sum(pdf);
    % compute the new directions for model 1
    v1 = (A-i*E*f(pt_cnt))\b;
    V1 = [V1,real(v1)/norm(real(v1)),imag(v1)/norm(imag(v1))];
    z1 = randn(size(b))+i*randn(size(b));
    Z1 = [Z1,real(z1)/norm(real(z1)),imag(z1)/norm(imag(z1))];
    % choose another point for model 2 based on current pdf
    rvar = rand(1);
    pt_cnt = 1;
    PDF = pdf(1);
    while rvar > PDF,
      pt_cnt = pt_cnt+1;
      PDF = PDF + pdf(pt_cnt);
    end
```

```
  % edit pdf to keep current rounds of points 1% distanced
  for l=max(1,pt_cnt-pt_sep):min(pt_tot,pt_cnt+pt_sep), pdf(l)=0; end
  pdf = pdf/sum(pdf);

  % compute the new directions for model 2
  v2 = (A-i*E*f(pt_cnt))\b;
  V2 = [V2,real(v2)/norm(real(v2)),imag(v2)/norm(imag(v2))];
  z2 = randn(size(b))+i*randn(size(b));
  Z2 = [Z2,real(z2)/norm(real(z2)),imag(z2)/norm(imag(z2))];
end

% Compute model number 1
A1 = Z1'*(A*V1); E1 = Z1'*(E*V1); b1 = Z1'*b; c1 = V1'*c;
[M,junk] = size(A1);
[Uc,Sc,Vc] = svd(A1-abs(f(1))*E1);
k = 1; while (Sc(1,1)/Sc(k,k) < 1e8) & (k<M), k = k+1; end; k=k-1;
A1 = Uc(:,1:k)'*A1*Vc(:,1:k); E1 = Uc(:,1:k)'*E1*Vc(:,1:k);
b1 = Uc(:,1:k)'*b1; c1 = Vc(:,1:k)'*c1;
% Compute model number 2
A2 = Z2'*(A*V2); E2 = Z2'*(E*V2); b2 = Z2'*b; c2 = V2'*c;
[M,junk] = size(A2);
[Uc,Sc,Vc] = svd(A2-abs(f(1))*E2);
k = 1; while (Sc(1,1)/Sc(k,k) < 1e8) & (k<M), k = k+1; end; k=k-1;
A2 = Uc(:,1:k)'*A2*Vc(:,1:k); E2 = Uc(:,1:k)'*E2*Vc(:,1:k);
b2 = Uc(:,1:k)'*b2; c2 = Vc(:,1:k)'*c2;
% Compute the difference between the two models to update the pdf
diff = zeros(size(f));
for k = 1:pt_tot,
  diff(k) = abs(c1'*((i*f(k)*E1-A1)\b1)-c2'*((i*f(k)*E2-A2)\b2));
```

```
  end
  pdf = diff/sum(diff);
end

Am=A1; Em=E1; bm=b1; cm=c1;
```

REFERENCES

[1] J. N. Reddy, *An Introduction to the Finite Element Method.* New York, NY: McGraw-Hill, 1985.

[2] D. Kahaner, C. Moler, and S. Nash, *Numerical Methods and Software.* Englewood Cliffs, NJ: Prentice Hall, 1989.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed., Baltimore, MD: John Hopkins University Press, 1989.

[4] N. P. van der Meijs and T. Smedes, "Accurate interconnect modeling: towards multi-million transistor chips as microwave circuits," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, 1996, pp. 244–251.

[5] R. D. Masiello, "It's put up or shut up for grid control," *IEEE Spectrum*, vol. 33, pp. 50–51, 1996.

[6] R. W. Freund, G. H. Golub, and N. M. Nachtigal, "Iterative solution of linear systems," *Acta Numer.*, vol. 1, pp. 57–100, 1992.

[7] Y. Saad, *Numerical Methods for Large Eigenvalue Problems.* Manchester, UK: Manchester University Press, 1992.

[8] Y. Saad, *Iterative Methods for Sparse Linear Systems.* Boston, MA: PWS Publishing Co., 1996.

[9] A. J. Laub, "Numerical linear algebra aspects of control design computations," *IEEE Trans. Autom. Control*, vol. 30, pp. 97–108, 1985.

[10] P. M. Van Dooren, "The generalized eigenstructure problem in linear system theory," *IEEE Trans. Autom. Control*, vol. 26, pp. 111–129, 1981.

[11] C. B. Moler and G. W. Stewart, "An algorithm for generalized matrix eigenvalue problems," *SIAM J. Numer. Anal.*, vol. 10, pp. 241–256, 1973.

[12] M. M. Alaybeyi, J. Y. Lee, and R. A. Rohrer, "Numerical integration algorithms and aysmptotic waveform evaluation (AWE)," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Santa Clara, CA, 1992, pp. 76–79.

[13] A. C. Antoulas, "A behavioral approach to model reduction," in *Proc. 34th IEEE Conf. Decision Control*, New Orleans, LA, 1995, pp. 490–491.

[14] E. Chiprout, E. Grimme, A. Devgan, and T. Nguyen, "Interconnect analysis in timing simulation using partitioning and multipoint block Arnoldi model reduction," in preparation, 1997.

[15] A. Ruhe, "Rational Krylov algorithms for nonsymmetric eigenvalue problems II: matrix pairs," *Linear Algebr. Appl.*, vol. 197, pp. 283–295, 1984.

[16] L. Fortuna, G. Nunnari, and A. Gallo, *Model Reduction Techniques with Applications in Electrical Engineering.* London, UK: Springer-Verlag, 1992.

[17] J. G. Doyle, B. A. Francis, and A. R. Tannenbaum, *Feedback Control Theory.* New York, NY: Macmillan, 1992.

[18] K. Meerbergen and D. Roose, "Matrix transformations for computing rightmost eigenvalues of large sparse nonsymmetric eigenvalue problems," *IMA J. Numer. Anal.*, vol. 16, pp. 297–346, 1996.

[19] R. B. Morgan and D. S. Scott, "Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 817–825, 1986.

[20] B. N. Datta and Y. Saad, "Arnoldi methods for large Sylvester-like observer matrix equations," *Linear Algebr. Appls.*, vol. 154-156, pp. 225–244, 1991.

[21] R. Freund, "Solution of shifted linear systems by quasi-minimal residual iterations," in *Numerical Linear Algebra,* L. Reichel, A. Ruttan, and R. Varga, eds., Berlin: W. de Gruypter, 1993, pp. 101–121.

[22] S. Choudhary, "On numerical solutions of large sparse linear systems and applications," PhD dissertation, Northern Illinois University, De Kalb, IL, 1994.

[23] L. A. Aguirre, "Quantitative measure of modal dominance for continuous systems," in *Proc. 32nd IEEE Conf. Decision Control*, San Antonio, TX, 1993, pp. 2405–2410.

[24] D. Bonvin and D. A. Mellichamp, "A unified derivation and critical review of modal approaches to model reduction," *Int. J. Control*, vol. 35, pp. 829–848, 1982.

[25] R. E. Skelton, *Dynamic System Control.* New York, NY: John Wiley & Sons, 1988.

[26] G. Franklin, J. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*, 2nd ed., Reading, MA: Addison-Wesley, 1991.

[27] Y. Shamash, "Viability of methods for computing stable reduced-order models," *IEEE Trans. Autom. Control*, vol. 26, pp. 1285–1286, 1981.

[28] B. C. Moore, "Principal component analysis in linear systems: controllability, observability and model reduction," *IEEE Trans. Autom. Control*, vol. 26, pp. 17–32, 1981.

[29] D. F. Enns, "Model reduction with balanced realizations: an error bound and frequency weighted generalizations," in *Proc. 23rd IEEE Conf. Decision Control*, Las Vegas, NV, 1984, pp. 127–132.

[30] R. H. Bartels and G. W. Stewart, "Algorithm 432: Solution of the matrix equation $AX + XB = C$," *Commun. ACM*, vol. 15, pp. 820–826, 1972.

[31] C. T. Chen, *Linear System Theory and Design.* New York, NY: Holt, Rinehart and Winston, 1984.

[32] W. B. Gragg and A. Lindquist, "On the partial realization problem," *Linear Algebr. Appl.*, vol. 50, pp. 277–319, 1983.

[33] G. A. Baker Jr., *Essentials of Padé approximation.* New York, NY: Academic Press, 1975.

[34] B. D. O. Anderson and A. C. Antoulas, "Rational interpolation and state space variable realizations," *Linear Algebr. Appl.*, vol. 138, pp. 479–509, 1990.

[35] O. H. Bosgra, G. Schoolstra, and M. Steinbuch, "Robust control of a compact disc player," in *Proc. 31st IEEE Conf. Decision Control*, Tucson, AZ, 1992, pp. 2596–2600.

[36] P. Wortelboer, "Frequency-weighted balanced reduction of closed-loop mechanical servo-systems," PhD dissertation, Technical University Delft, Delft, Neth., 1994.

[37] D. Boley and G. Golub, "The nonsymmetric Lanczos algorithm and controllability," *Syst. Control Lett.*, vol. 16, pp. 97–105, 1991.

[38] C. Brezinski, *Padé-Type Approximation and General Orthogonal Polynomials.* Basel: Birkhauser, 1980.

[39] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Natl. Bur. Stand.*, vol. 45, pp. 255–282, 1950.

[40] A. Bultheel and M. Van Barel, "Padé techniques for model reduction in linear system theory: a survey," *J. Comput. Appl. Math.*, vol. 14, pp. 401–438, 1986.

[41] Y. Shamash, "Linear system reduction using Padé approximation to allow retention of dominant modes," *Int. J. Control*, vol. 21, pp. 257–272, 1975.

[42] C. Hwang and M. Y. Chen, "A multi-point continued fraction expansion for linear system reduction," *IEEE Trans. Autom. Control*, vol. 31, pp. 648–651, 1986.

[43] C. Kenney, A. J. Laub, and S. Stubberud, "Frequency response computation via rational interpolation," *IEEE Trans. Autom. Control*, vol. 38, pp. 1203–1213, 1993.

[44] H. Xiheng, "FF-Padé method of model reduction in frequency domain," *IEEE Trans. Autom. Control*, vol. 32, pp. 243–246, 1987.

[45] L. T. Pillage and R. A. Rohrer, "Aysmptotic waveform evaluation for timing analysis," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 352–366, 1990.

[46] E. Chiprout and M. S. Nakhla, *Asymptotic Waveform Evaluation and Moment Matching for Interconnect Analysis.* Boston, MA: Kluwer Academic Publishers, 1994.

[47] K. Gallivan, E. Grimme, and P. Van Dooren, "Asymptotic waveform evaluation via a Lanczos method," *Appl. Math. Lett.*, vol. 7, pp. 75–80, 1994.

[48] P. Feldman and R. W. Freund, "Efficient linear circuit analyis by Padé approximation via a Lanczos method," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 639–649, 1995.

[49] C. D. Villemagne and R. E. Skelton, "Model reduction using a projection formulation," *Int. J. Control*, vol. 46, pp. 2141–2169, 1987.

[50] B. Nour-Omid and R. W. Clough, "Dynamic analysis of structures using Lanczos coordinates," *Earthq. Eng. Struct. Dyn.*, vol. 12, pp. 565–577, 1984.

[51] I. U. Ojalvo and M. Newman, "Vibration modes of large structures by an automatic matrix-reduction method," *AIAA J.*, vol. 8, pp. 1234–1239, 1970.

[52] E. L. Wilson, M. W. Yuan, and J. M. Dickens, "Dynamic analysis by superposition of Ritz vectors," *Earthq. Eng. Struct. Dyn.*, vol. 10, pp. 813–821, 1982.

[53] T. J. Su and R. R. Craig Jr., "Krylov vector methods for model reduction and control of flexible structures," *Adv. Control Dynamic Syst.*, vol. 54, pp. 449–481, 1992.

[54] H. M. Kim and R. R. Craig Jr., "Structural dynamics analysis using an unsymmetric block Lanczos algorithm," *Int. J. Numer. Methods Eng.*, vol. 26, pp. 2305–2318, 1988.

[55] H. M. Kim and R. R. Craig Jr., "Computational enhancement of an unsymmetric block Lanczos algorithm," *Int. J. Numer. Methods Eng.*, vol. 30, pp. 1083–1089, 1990.

[56] M. M. M. Al-Husari, B. Hendel, I. M. Jaimoukha, E. M. Kasenally, D. J. N. Limebeer, and A. Portone, "Vertical stabilisation of tokamak plasmas," in *Proc. 30th IEEE Conf. Decision Control*, Brighton, UK, 1991, pp. 1165–1170.

[57] D. L. Boley, "Krylov space methods on state-space control models," *Circuits Syst. Signal Process.*, vol. 13, pp. 733–758, 1994.

[58] P. M. Van Dooren, "Numerical linear algebra techniques for large scale matrix problems in systems and control," in *Proc. 31st IEEE Conf. Decision Control*, Tucson, AZ, 1992, pp. 1933–1938.

[59] I. M. Jaimoukha and E. M. Kasenally, "Oblique projection methods for large scale model reduction," *SIAM J. Matrix Anal. Appl.*, vol. 16, pp. 602–627, 1995.

[60] E. J. Grimme, D. C. Sorensen, and P. M. Van Dooren, "Model reduction of state space systems via an implicitly restarted Lanczos method," *Numer. Algorithms*, vol. 12, pp. 1–31, 1996.

[61] P. Feldman and R. W. Freund, "Reduced-order modeling of large linear subcircuits via a block Lanczos algorithm," in *Proc. 32nd ACM/IEEE Design Automation Conf.*, San Fransisco, CA, 1995, pp. 474–479.

[62] P. Feldman and R. W. Freund, "Reduced-order modeling of large passive linear circuits by means of the SyPVL algorithm," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, 1996, pp. 280–287,

[63] K. J. Kerns, I. L. Wemple, and A. T. Yang, "Stable and efficient reduction of substrate model networks using conguence transforms," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, 1995, pp. 207–214.

[64] L. M. Silveira, M. Kamon, I. Elfadel, and J. White, "A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, San Jose, CA, 1996, pp. 288–294.

[65] C. Lanczos, "Solution of systems of linear equations by minimized iterations," *J. Res. Natl. Bur. Stand.*, vol. 49, pp. 33–53, 1952.

[66] M. H. Gutknecht, "A completed theory of the unsymmetric Lanczos process and related algorithms, part I," *SIAM J. Matrix Anal. Appl.*, vol. 13, pp. 594–639, 1992.

[67] M. H. Gutknecht, "A completed theory of the unsymmetric Lanczos process and related algorithms, part II," *SIAM J. Matrix Anal. Appl.*, vol. 15, pp. 15–58, 1994.

[68] B. N. Parlett, D. R. Taylor, and Z. S. Liu, "A look-ahead Lanczos algorithm for unsymmetric matrices," *Math. Comp.*, vol. 44, pp. 105–124, 1985.

[69] R. W. Freund, M. H. Gutknecht, and M. H. Nachtigal, "An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices," *SIAM J. Sci. Comp.*, vol. 14, pp. 137–158, 1993.

[70] Y. Saad, "A flexible inner-outer preconditioned GMRES algorithm," *SIAM J. Sci. Comput.*, vol. 14, pp. 461–469, 1993.

[71] H. A. van der Vorst and C. Vuik, "GMRESR: A family of nested GMRES methods," *Numer. Linear Algebr. Appl.*, vol. 1, pp. 369–386, 1994.

[72] G. Peters and J. H. Wilkinson, "Inverse iteration, ill-conditioned equations and Newton's method," *SIAM Review*, vol. 21, pp. 339–360, 1979.

[73] B. N. Parlett and D. S. Scott, "The Lanczos algorithm with selective orthogonalization," *Math. Comput.*, vol. 33, pp. 217–238, 1979.

[74] N. N. Abdelmalek, "Round off error analysis for Gram-Schmidt method and solution of linear least squares problems," *BIT*, vol. 11, pp. 345–368, 1971.

[75] W. E. Arnoldi, "The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Q. Appl. Math.*, vol. 9, pp. 17–29, 1951.

[76] C. I. Byrnes and A. Lindquist, "The stability and instability of partial realizations," *Syst. Control Lett.*, vol. 2, pp. 99–105, 1982.

[77] E. Grimme, "An implicitly restarted Lanczos method for the model reduction of stable, large-scale systems," Master's thesis, University of Illinois, Urbana, IL, 1993.

[78] I. M. Jaimoukha and E. M. Kasenally, "Implicitly restarted subspace methods for stable partial realisations," to appear *SIAM J. Matrix Anal. Appl.*, 1997.

[79] J. Cullum and W. E. Donath, "A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace of large sparse real symmetric matrices," in *Proc. 13th IEEE Conf. Decision Control*, Phoeniz, AZ, 1974, pp. 505–509.

[80] A. Ruhe, "Rational Krylov methods for eigenvalue computation," *Linear Algebr. Appl.*, vol. 58, pp. 391–405, 1984.

[81] D. Skoogh, "An implementation of a parallel rational Krylov algorithm," PhD dissertation, Göteborg University and Chalmers University of Technology, Göteborg, Swed., 1996.

[82] K. Gallivan, E. Grimme, and P. Van Dooren, "A rational Lanczos method for model reduction," *Numer. Algorithms*, vol. 12, pp. 33–63, 1996.

[83] V. L. Druskin and L. A. Knizherman, "Error bounds in the simple Lanczos procedure for computing functions of symmetric matrices and eigenvalues," *Comput. Maths. Math. Phys.*, vol. 31, pp. 20–30, 1991.

[84] H. F. Walker, "Implementation of the GMRES method using Householder transformations," *SIAM J. Sci. Stat. Comput.*, vol. 9, pp. 152–163, 1988.

[85] A. Greenbaum, "Behavior of slightly perturbed Lanczos and Conjugate-Gradient recurrences," *Linear Algebr. Appl.*, vol. 113, pp. 7–63, 1989.

[86] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*, 2nd ed., New York, NY: Van Nostrand Reinhold, 1994.

[87] R. Sanaie, E. Chiprout, M. Nakhla, and Q. J. Zhang, "A fast method for frequency and time domain simulation of high speed VLSI interconnects," *IEEE Trans. Microw. Theory Tech.*, vol. 42, pp. 2562–2571, 1994.

[88] N. A. Bruinsma and M. Steinbuch, "A fast algorithm to compute the H_∞-norm of a transfer function matrix," *Syst. Control Lett.*, vol. 14, pp. 287–293, 1990.

[89] A. Grace, A. J. Laub, J. N. Little, and C. Thompson, *Control System Toolbox.* Natick, MA: The MathWorks, 1990.

[90] H. Heeb, A. E. Ruehli, J. E. Bracken, and R. A. Rohrer, "Three dimensional circuit oriented electromagnetic modeling for VLSI interconnects," in *Proc. IEEE Int. Conf. Computer Design*, Cambridge, MA, 1992, pp. 218–221.

[91] A. Ruhe, "The rational Krylov algorithm for nonsymmetric eigenvalue problems III: Complex shifts for real matrices," *BIT*, vol. 34, pp. 165–176, 1994.

[92] I. M. Longman, "Best rational function approximation for Laplace transform inversion," *SIAM J. Math. Anal.*, vol. 5, pp. 574–580, 1974.

[93] T. N. Lucas, "Optimal model reduction by multipoint Padé approximation," *J. Franklin Inst.*, vol. 330, pp. 79–93, 1993.

[94] V. Kumar, A. Grama, A. Gupta, and G. Karypis, *Introduction to Parallel Computing: Design and Analysis of Algorithms.* Redwood City, CA: Benjamin/Cummings, 1994.

[95] Y. Saad, "ILUT: A dual threshold incomplete LU factorization," *Numer. Linear Algebra Appl.*, vol. 1, pp. 387–402, 1994.

[96] M. J. Grote and T. Huckle, "Parallel preconditioning with sparse approximate inverses," *SIAM J. Sci. Comp.*, 1997.

[97] S. F. Ashby, "Minimax polymomial preconditioning for Hermitian linear systems," *SIAM J. Matrix Anal. Appl.*, vol. 12, pp. 766–789, 1991.

[98] E. R. Davidson, "The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices," *J. Comput. Phys.*, vol. 17, pp. 87–94, 1975.

[99] Y. Saad, "Projection and deflation methods for partial pole assignment in linear state feedback," *IEEE Trans. Autom. Control*, vol. 33, pp. 290–297, 1988.

[100] E. de Sturler and D. Fokkema, "Nested Krylov methods and preserving orthogonality," in *Proc. 6th Copper Mountain Multigrid Conf.*, Copper Mountain, CO, 1993, pp. 111–125.

[101] D. C. Sorensen, "Implicit application of polynomial filters in a k-step Arnoldi method," *SIAM J. Matrix Anal. Appl.*, vol. 13, pp. 357–385, 1992.

[102] B. N. Parlett, *The Symmetric Eigenvalue Problem.* Englewood Cliffs, NJ: Prentice-Hall, 1980.

[103] T. Ericsson and A. Ruhe, "The spectral transformation method for the numerical solution of large sparse generalized eigenvalue problems," *Math. Comp.*, vol. 35, pp. 1251–1268, 1980.

[104] R. B. Morgan and D. S. Scott, "Preconditioning the Lanczos algorithm for sparse symmetric eigenvalue problems," *SIAM J. Sci. Stat. Comput.*, vol. 14, pp. 585–593, 1993.

[105] K. Meerbergen and D. Roose, "The restarted Arnoldi method applied to iterative linear system solvers for computation of right-most eigenvalues," *SIAM J. Matrix Anal. Appl.*, vol. 18, pp. 1–20, 1997.

[106] D. S. Scott, "The advantages of inverted operators in Raleigh-Ritz approximations," *SIAM J. Sci. Stat. Comput.*, vol. 3, pp. 68–75, 1982.

[107] R. B. Morgan, "Computing interior eigenvalues of large matrices," *Linear Algebr. Appl.*, vol. 154-156, pp. 289–309, 1991.

[108] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, "Approximate solutions and eigenvalue bounds from Krylov spaces," *Numer. Linear Algebr. Appl.*, vol. 2, pp. 115–134, 1995.

[109] J. Olsen, P. Jørgensen, and J. Simons, "Passing the one-billion limit in full configuration-interaction (FCI) calculations," *Chem. Phys. Lett.*, vol. 169, pp. 463–472, 1990.

[110] G. L. G. Sleijpen and H. van der Vorst, "A Jacobi-Davidson iteration method for linear eigenvalue problems," *SIAM J. Matrix Anal. Appl.*, vol. 17, pp. 401–425, 1996.

[111] H. A. van der Vorst and G. H. Golub, "150 years old and still alive: Eigenproblems," Technical Report SCCM-96-11, Stanford University, Palo Alto, CA, 1996.

[112] R. B. Morgan, "Davidson's method and preconditioning for generalized eigenvalue problems," *J. Comput. Phys.*, vol. 89, pp. 241–245, 1990.

[113] M. Crouzeix, B. Philippe, and M. Sadkane, "The Davidson method," *SIAM J. Sci. Comput.*, vol. 15, pp. 62–76, 1994.

[114] A. Stathopoulos, Y. Saad, and C. F. Fischer, "Robust preconditioning of large sparse symmetric eigenvalue problems," *J. Comput. Appl. Math*, vol. 64, pp. 197–216, 1995.

[115] M. Sadkane, "Block-Arnoldi and Davidson methods for unsymmetric large eigenvalue problems," *Numer. Math.*, vol. 54, pp. 195–211, 1993.

VITA

Eric James Grimme was born in Cincinnati, Ohio, on September 28, 1970. He entered The Ohio State University in 1988 as a National Merit Scholar and graduated with honors in Electrical Engineering in 1992. In 1994, he received the Master of Science Degree in Electrical Engineering from the University of Illinois at Urbana-Champaign, where he was a University Fellow. During his doctoral studies at the University of Illinois, he worked as a Department of Energy Computational Science Fellow. His research interests lie in the area of computational techniques for systems, controls and circuits. He is a member of Eta Kappa Nu and the Institute of Electrical and Electronics Engineers.