Structured and Parameter-Dependent Eigensolvers for Simulation-Based Design of Resonant MEMS

David Samuel Bindel

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2006-108
http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-108.html

August 22, 2006

Structured and Parameter-Dependent Eigensolvers for Simulation-Based Design of Resonant MEMS

by

David Samuel Bindel

B.S. (University of Maryland, College Park) 1999

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor James W. Demmel, Co-chair
Professor Sanjay Govindjee, Co-chair
Professor William M. Kahan
Professor Robert L. Taylor

Fall 2006

The dissertation of David Samuel Bindel is approved.

Co-chair	Date
Co-chair	Date
	Date
	Date

University of California, Berkeley

Fall 2006

Structured and Parameter-Dependent Eigensolvers for

Simulation-Based Design of Resonant MEMS

by

David Samuel Bindel

Abstract

Structured and Parameter-Dependent Eigensolvers for
Simulation-Based Design of Resonant MEMS

by

David Samuel Bindel

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor James W. Demmel, Co-chair

Professor Sanjay Govindjee, Co-chair

This dissertation is about computational tools to aid in the design of resonant Micro-Electro-Mechanical Systems (MEMS), tiny vibrating devices built by processes like those used to make integrated circuits. Vibrating MEMS are used in accelerometers and gyroscopes, in sensors to detect chemicals and to measure pressure, and in communication devices such as cell phones. MEMS engineers can use computer simulations to design devices using fewer costly and time-consuming prototype tests, but these simulations are only as useful as the models on which they are built. In this work, we contribute new mathematical models, numerical methods, and software tools to simulate resonant MEMS, and apply these tools to analyze specific devices. We describe physical models of damped vibrations of MEMS, including anchor loss and thermoelastic effects which are widely recognized as important, but not modeled in generality by existing tools. Though the resulting systems of equations are large and non-Hermitian, and depend nonlinearly on frequency, we use the equation structure

to develop efficient structured Krylov subspace projection methods for computing free vibrations and reduced-order models. We also provide efficient continuation methods for re-computing eigendecompositions under changes to design parameters or operating conditions. Our models and analysis methods are integrated into `HiQLab`, a new finite element tool with a particularly flexible architecture which we have designed. Using `HiQLab`, we simulate example resonator designs, and compare our results to laboratory measurements. Our simulations reveal a previously-unknown mode interference phenomenon, subsequently observed in experiments, which dramatically affects the amount of damping near certain critical values of geometric parameters.

Professor James W. Demmel
Dissertation Committee Co-chair

Professor Sanjay Govindjee
Dissertation Committee Co-chair

Contents

Acknowledgements

Jim Demmel and Sanjay Govindjee co-advised me during this work, and I thank them for sharing their technical insight and their curiosity.

Jim Demmel and Velvel Kahan shaped how I think about numerical computation and linear algebra; and Sanjay Govindjee, Bob Taylor, and Panos Papadopoulos shaped my understanding of finite elements and computational mechanics. Zhaojun Bai taught me about model reduction and the importance of preserving structure; Ming Gu taught me about semi-separable structure; and Mark Friedman introduced me to numerical bifurcation analysis and parameter-dependent eigenvalue problems. Kris Pister introduced me to MEMS and MEMS simulation, and Roger Howe provided encouragement and technical guidance for my efforts to simulate RF-MEMS.

Sunil Bhave first interested me in RF-MEMS, and provided several of my favorite example problems. Emmanuel Quévy built and measured the disk resonators we used to test our models. Ushnish Basu taught me about perfectly matched layers. Tsuyoshi Koyama contributed to the HiQLab code, and was responsible for most of the exploratory simulations of thermoelastic damping.

Finally, I am indebted to the faculty who taught me, in lectures and in informal discussions, and to fellow graduate students who were willing to "talk shop" and who answered my questions and listened to my impromptu lectures with equal good grace. More than anything else, it is these friends, colleagues, and guides who made my graduate training satisfying and worthwhile.

The work described here was in part supported by the National Science Foundation Grant ECS-0426660; the University of California MICRO program; Sun Microsystems; and the National Science Foundation Graduate Student Fellowship program.

Chapter 1

Introduction

This dissertation is about computational tools to aid in the design of resonant Micro-Electro-Mechanical Systems (MEMS), tiny vibrating devices built by processes like those used to make integrated circuits. Resonant MEMS are used in accelerometers and gyroscopes, in sensors to detect chemicals and to measure pressure, and in communication devices such as cell phones. Better computer simulations let engineers design MEMS using fewer costly and time-consuming "build-and-break" prototype tests, both by helping the designers develop insight and by giving them quantitative predictions of device behavior. In this dissertation, we provide:

- Mathematical models of the physics of resonant MEMS
- Efficient and accurate algorithms for analyzing these models
- `HiQLab`, a software system that includes our model and algorithms
- Analyses of specific resonant MEMS, and comparisons of our simulations to laboratory measurements.

We focus on resonant MEMS, but our ideas apply more broadly. In general, our numerical methods apply to eigenvalue or resonance calculations for parameter-

dependent problems, problems with complex symmetry, and problems involving certain perturbation structures. The `HiQLab` software architecture is also general, and includes features which are useful for other types of finite element simulations.

Our work comprises three parts in which we describe, respectively, physical models and mathematical tools, simulation software, and analyses of specific devices. In the first and largest part, we describe models of MEMS damping together with projection methods for computing free vibrations and forced response of the damped systems. We concentrate on anchor loss and thermoelastic damping, two damping mechanisms considered important by MEMS designers. Anchor loss is particularly interesting, as no other MEMS simulation tools we know provide anchor loss models. We provide general finite-element formulations to model anchor loss and thermoelastic damping, describe factors affecting the accuracy of our formulation, and show how the structure of the equations leads to methods to calculate individual modes and reduced-order models more quickly and accurately than may be done with standard methods. We also present new subspace continuation techniques that allow us to quickly reanalyze the linear dynamics of device models when the design parameters and nonlinear operating points change. The problem structures we use in our model reduction and subspace continuation methods are not specific to MEMS, as we show in our description of our work to integrate subspace continuation into the MATCONT numerical bifurcation analysis package [61].

In the second part of this dissertation, we describe `HiQLab`, our finite element tool for simulating resonant MEMS. By writing a new code, we provide special support for the new elements and solvers described in the first part of our work. The core of `HiQLab` is a small object-oriented library of finite element data structures built along standard lines; and the primary user interface is written in MATLAB, which allows us to quickly try new solution algorithms and to write short scripts for parameter studies and design optimization tasks. The main novelty in the `HiQLab` architecture

is a mesh description system built around the Lua scripting language, which provides such features as automatic physically-motivated rescaling of problem variables and specification of spatial fields via callback functions. `HiQLab` lacks the pre- and post-processing features of commercial finite element codes, but it includes algorithms and elements that other codes lack. Because `HiQLab` has an open, flexible architecture, it is a useful environment for developing new elements and algorithms.

In the final part of the dissertation, we use `HiQLab` to analyze two example resonant MEMS, and compare our analyses to laboratory measurements. Our analysis highlights interesting characteristics of both the software and the devices under analysis. In particular, one of the contributions of this section is the discovery of a previously-unknown interference phenomenon which leads to massively degraded device performance near certain frequencies. The mechanism by which this occurs, and the means of avoiding it, apply to other very high-frequency resonant MEMS devices as well.

1.1 Outline and Contributions

The detailed organization of the rest of the dissertation is as follows:

- In the remainder of this introductory chapter, we describe how resonant MEMS are used in applications, why it is important to understand damping in these devices, and why previous simulation tools do not suffice to provide designers with damping information.

- In Chapter 2, we review Galerkin projection methods for analyzing the frequency response of linear systems. We illustrate issues that arise in our later, more realistic models by analyzing a sequence of one-dimensional problems. Though this chapter includes no fundamentally new theorems, we contribute a

unifying presentation that ties together results from finite element analysis and numerical linear algebra.

- In Chapter 3, we describe how to compute damping due to radiation of elastic waves from a vibrating MEMS device into a large substrate. We simulate this radiation behavior using an absorbing *perfectly-matched layer* (PML), which we write as a complex-valued change of coordinates. We then give a novel formulation of finite elements for PMLs, which makes it easier to program PMLs for elastic problems. Our finite element equations have a complex symmetric structure, which we use to develop a new model reduction procedure which is about twice as accurate as standard algorithms using the same amount of work. In addition, we present a new error analysis and resulting automated methods for choosing PML parameters to minimize reflections in the discrete model. The contents of this chapter largely overlap our previously-published work [30].

- In Chapter 4, we describe thermoelastic damping, a well-known loss mechanism which particularly affects MEMS beams and other flexural devices. After reviewing the equations of thermoelasticity and their discretization, we then give a new perturbation method for computing thermoelastic damping of mechanical vibration modes. Our method generalizes Zener's method of orthogonal thermodynamic potentials [189] to work with finite elements, which can be used in much more complicated geometries than those that Zener considered.

- In Chapter 5, we describe the *continuation of invariant subspaces* (CIS) algorithm. The CIS algorithm is used to quickly compute continuous partial eigendecompositions of continuously parameter-dependent matrices. Such decompositions provide useful information to engineers who wish to understand how device behavior changes under changes in design parameters; they are also useful for understanding the dominant linear dynamics, and in particular the

system stability, near a dynamical equilibrium point in a general nonlinear system. Previous work on subspace continuation has focused on dense problems; because these methods are prohibitively expensive when the system dimension grows larger than a few hundred, they are unsuitable for analysis of most PDE problems. Our work combines invariant subspace continuation with Krylov subspace projection methods for the first time, resulting in an algorithm which computes continuous bases and is much faster than the previous method. We also describe new algorithms for adapting the continued subspace in order to improve the robustness of numerical stability analyses. In addition, we contribute a theoretical result of independent interest, new sufficient conditions for existence of a continuously-defined invariant subspace connecting subspaces at the end-points of a curve in matrix space. Our bounds can be arbitrarily tighter than previous results based on conventional perturbation expansions. We have published most of the contents of this chapter previously as a technical report [34].

- In Chapter 6, we describe the software architecture of the `HiQLab` simulation system. `HiQLab` includes the new models developed in previous chapters; in particular, it is the first available MEMS simulation code we know to compute anchor damping. We highlight novel aspects of the code, and particularly emphasize the flexibility gained by describing analyses in MATLAB and by describing meshes in the Lua scripting language. This flexibility makes it easy to prototype new elements and solvers, and to programmatically define meshes for use in parameter studies and optimization tasks. The abstraction mechanisms we use do not affect the expensive parts of typical computations, and so the flexibility of `HiQLab` comes with a negligible performance penalty.

- In Chapter 7, we describe analyses in `HiQLab` of a checkerboard resonator and

of a family of disk resonators. We use the checkerboard example to highlight the advantages of reduced-order modeling by building an application with which engineers can visualize forced vibrations. We can evaluate the reduced-model at each frequency point in milliseconds rather than seconds, which is fast enough to allow users to interactively scan through different forcing frequencies, viewing the response shape at each frequency. With the disk example, we demonstrate the effectiveness of our anchor loss models; we also uncover a previously-unknown physical phenomenon by which losses in a parasitic mode lead to drastic increases in damping near certain critical values of geometric design parameters. We discovered this behavior through our simulations; it was subsequently verified in laboratory experiments. Our analysis of the mechanism underlying this damping suggests that it will substantially affect a variety of high-frequency resonant MEMS designs; we give insight into what designs may suffer this effect, and how they can be changed to avoid the trouble. We validate the simulations of both the checkerboard and the disks by comparing the predicted behavior to experimental data. Parts of this chapter appear in our previously-published papers [33, 31, 30].

1.2 Resonant MEMS in applications

In this section, we review some applications in which resonant MEMS are used. For these applications, it is particularly important to have components with little damping. Our work to simulate different damping mechanisms is motivated by the engineering need to design these lightly-damped MEMS components.

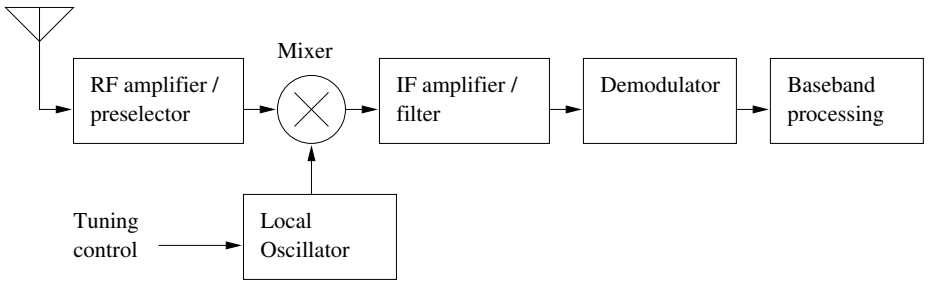

Figure 1.1. Block diagram of a super-heterodyne radio receiver (adapted from [112, §13.1]).

1.2.1 Resonant MEMS in radio systems

In the super-heterodyne architectures used in most modern radio devices, radio-frequency (RF) signals received by the antenna are translated to a lower intermediate frequency (IF) before being demodulated and processed. Figure 1.1 shows a simplified block diagram for a simple super-heterodyne receiver [112, §13.1]. Resonant mechanical devices are already likely to be used in all but the last stage in this receiver architecture. Because the mixer translates multiple RF frequencies to the same intermediate frequency, a surface acoustic wave (SAW) filter or film bulk acoustic resonator (FBAR) is used to pre-select the relevant RF range. The mixer used to effect the translation will itself use a frequency synthesizer built around a precisely tuned crystal oscillator. At the intermediate frequency, additional resonant mechanical passive components may be used in further filtering and demodulation. In all regards, modern radios – including cell phones – are mechanical systems as well as electrical ones.

Engineers use mechanical components because it would often be impossible to realize equivalent functionality with an integrated circuit. The key advantage of mechanical components is their high quality of resonance (Q), a dimensionless ratio of the total energy stored in a resonating system to the energy lost per cycle of oscillation. The quality factors of resonating mechanical systems are orders of magnitude higher

than the quality factors of purely electrical systems implemented in conventional integrated circuitry. Components with high quality factors are necessary in order to build very narrow-band filters, or filters which must have sharp edges in order to distinguish between narrowly separated channels in a frequency-division multiplexing system. High quality factors are also critical in building good frequency references. In general, the higher the quality factor, the lower the power consumption and the better the noise behavior; this "explains the traditional obsession of engineers with maximizing resonator Q" [112, §18.2].

The mechanical resonators in current architectures are discrete components, not integrated on a single chip with the electronics. In contrast, surface micro-machined MEMS (microelectromechanical system) resonators can be integrated into standard complementary metal-oxide semiconductor (CMOS) circuit technology, and so they have the potential to use less area and power, and cost less money, than existing resonators [134]. Beyond replacing their discrete counterparts, these integrated micromechanical devices may be used in large numbers to implement radio architectures that would be impossible to build with discrete components. Nguyen has proposed several MEMS-based architectural features which would allow order-of-magnitude power savings to be attained in the RF front-end, allowing mobile phones to operate for much longer on a single charge [133]. To implement these features requires micromechanical components with high quality factors.

1.2.2 Resonant MEMS in sensors

High frequency electromechanical resonators are also important components in sensing systems. Acoustic wave devices, and particularly SAW-based devices, have been used as sensors for many years [174]. Frequency measurement is relatively precise and robust in the face of electronic imperfections [107, p.260]. At the same time, a res-

onator's frequency response depends on geometry, material properties, and boundary conditions, so that shifts in resonance of SAW devices can be used to sense a variety of physical properties, including chemical concentrations [7], temperature [94], flow rate [104], and liquid density and viscosity [121]. Resonant silicon microsensors made with MEMS processes have been used for a similar variety of measurements [155]. Even a device as simple as a cantilever silicon beam has seen over two decades of development for sensing, and cantilever-based resonating biosensors continue to be the subject of active development [46].

The resonant frequency of a resonating sensor should be sensitive to changes in the measured variable, and insensitive to other environmental parameters (such as temperature). Consequently, it is important to understand how changes to the design change both the nominal frequency and the frequency sensitivity. Resonant sensors should also have high quality factors. If the quality factor is too low, frequency changes cannot be measured precisely. Low quality factors can also lead to less stable measurements, since shifts in the resonant frequency might be due not only to the measured quantity, but also to the influence of the amplifiers and control circuitry [155].

1.3 Damping in resonant MEMS

In the applications described in the last section, it is important to have components with high quality factors. A variety of damping mechanisms can lower the quality factors in resonant MEMS, including gas damping, material losses, thermoelastic damping (TED), and anchor losses. Which loss mechanism matters most to a particular design depends on the device geometry, the materials used, the environment, and the operating frequency range. All of these damping mechanisms have been studied experimentally for different types of cantilever beam resonators [93, 183, 180, 181, 76, 184];

for more general perspectives, we refer to the recent report by Brotz [39] and other general reviews [119]. In the rest of this section, we briefly review existing work on modeling each of these damping mechanisms. For the rest of the dissertation, though, we will focus on TED and anchor losses. These two mechanisms are known to dominate loss in some high-frequency MEMS; and unlike gas damping and some forms of material loss, TED and anchor loss cannot be eliminated by better processing or packaging methods.

1.3.1 Gas damping

Except in vacuum-packed devices and in some high-frequency bulk resonators, gas damping is usually the dominant energy loss mechanism in resonant MEMS. In air at standard temperature and pressure, the incompressible Navier-Stokes equations provide a good model for the flow around larger MEMS devices. At these scales, viscous forces dominate inertial forces, usually by one or two orders of magnitude, so that the inertial terms can be dropped from the Navier-Stokes equation; what remains is the equation for creeping flow (Stokes' equation) [24, §4.8]. Many of the interesting flows in MEMS devices involve thin films of fluid that fill narrow gaps between structures; by using the large aspect ratios in these films, the equations governing fluid flow simplify even further. The two most common cases are when the surfaces bounding a film move parallel to each other, leading to shear flows [47]; or when they move perpendicular to each other, leading to squeeze-film effects [154]. For more complicated motions and geometries, and for testing the accuracy of simplified models, researchers have turned to fast boundary element methods for the Stokes equation [178, 185], or sometimes to finite element methods [21, 182].

Continuum models based on the incompressible Navier-Stokes equations do not cover all cases of interest for MEMS designers. The ratio between the mean free

path and a characteristic length scale is known as the Knudsen number; and when the Knudsen number is not small, sub-continuum deviations from the Navier-Stokes model begin to play a role. For Knudsen numbers greater than 10^{-2}, slip flow along walls starts to become important; for Knudsen numbers between 10^{-1} and 10 the flow is in the transitional regime, where the bulk of the fluid no longer conforms to the incompressible constant-viscosity assumptions of the Navier-Stokes equation; and for Knudsen numbers greater than 10, one enters the regime of free molecular flow, where interactions between gas molecules are much less frequent than interactions between a gas molecule and the solid boundary [115]. At standard temperature and pressure, the mean free path of a molecule in air is 65 nm [115]. Gaps smaller than ten times this mean free path (6.5 microns) are not uncommon, and half-micron gaps are certainly possible, so slip flow conditions and compressibility effects are important in real devices even in ordinary air. Free molecular flows occur in devices packaged at low pressures.

Broadly speaking, there have been two types of models for studying gas damping when the Knudsen number is not small. First, there are models derived from statistical mechanics in which a distribution of gas molecule velocities is used to compute the momentum transferred from the vibrating device into the gas [48, 131, 193, 105, 184]. Second, there are models which incorporate corrections to the continuum model: in particular, these include modifications to Reynolds' equations which introduce a pressure-dependent effective viscosity [170, 172, 76]. Additional modifications have been proposed which incorporate not only compressibility effects, but also the effects of gas flow through vent holes [171, 21, 111, 92].

Gas damping was recognized early as a major energy loss mechanism for microresonators [131], as it is for macroscopic resonators. Consequently, the role of gas damping in microresonators has been widely studied in both models and experiments.

We refer the reader to [115, 186] for more comprehensive literature reviews on the subject.

1.3.2 Material losses

Real materials have hysteretic behavior: the stress depends not only on the current strain, but also on the recent strain history. Changes in strain result in changes to the distribution of defects and chemical impurities in a material, and it takes time for these modified distributions to relax back to equilibrium. The irreversible work lost in these various relaxation phenomena is lumped under the heading of material losses.

In studies of vibrating metal microcantilevers, the hysteresis effects in the bulk of the material played a large role [93]. But many MEMS devices are not made of metal; instead, they are made of high-quality single crystals of silicon, or of films of polycrystalline silicon or silicon alloys. The internal losses in these materials are much smaller than the internal losses in metals [155, 93]. However, because of the large surface area to volume ratio in small devices, hysteretic losses in the material at the surface can also be important. Surface damage during etching, the presence of native oxide layers, and adsorbed contaminants may all play roles in this surface damping [126, 183, 180, 181]. In experiments on submicron cantilevers, the scaling of the overall damping was consistent with the scaling of surface losses; furthermore, heat treatments meant to remove adsorbed material halved the damping in these devices [183, 180].

1.3.3 Thermoelastic damping (TED)

A material expands when the temperature is raised; thermodynamic consistency demands that this cannot be a distinct cause and effect, but that changes in volume should lead to changes in temperature. Therefore pressure waves passing through most media create spatial variations in temperature, and the relaxation of those spatial variations through heat diffusion leads to attenuation of the pressure wave.

As early as 1868, Kirchhoff studied the attenuation of sound waves through thermal relaxation, but the effect was not much studied in solids until a series of papers by Clarence Zener starting in 1937 [187, 188, 190, 143]. In these papers and in a 1948 research monograph [189], Zener described the mechanism of thermoelastic damping (TED), derived approximate formulas for the damping effects in beams and other simple geometries, and compared his predictions to experimental measurements. The theory of thermoelasticity developed rapidly starting in the early 1950s, particularly after Biot in a 1956 paper derived the governing equations in variational terms, based on firm foundations of irreversible thermodynamics [35]. For further description of the development of thermoelasticity, we refer to Nowacki's monograph [136] and to a survey paper by Chadwick [42].

Certainly by the 1950s, designers of mechanical filters and resonators were aware of thermoelastic damping and related effects [122]. Thermoelastic effects continued to be a concern as resonant microsystems were developed. In 1990, Roszhart published a paper in which he found that damping in silicon microbeams in a vacuum showed good agreement with predictions based on Zener's theory [145]. In subsequent investigations of damping in MEMS, TED has been repeatedly identified as a significant, or even dominant, contribution [41, 67, 95]. These investigations have typically relied on Zener's approximate solution for flexural waves, or on refinements to Zener's approximation derived under similar kinematic assumptions about the de-

vice motion [118, 135, 153]. Gorman and Duwel have used finite-element simulations in FEMLAB to numerically solve the coupled equations of elasticity [67, 83], but because of the prominent role played by flexural mechanisms in many MEMS designs, variants of Zener's approximation have generally dominated the literature.

1.3.4 Anchor loss

Another damping mechanism is anchor loss (also called clamping loss). Some of the energy in a vibrating MEMS structure can leak through an anchoring structure and into the substrate, there to be dissipated by internal loss mechanisms. Though these clamping mechanisms are widely cited as a possible loss mechanism [41, 119], and several authors have recommended balanced designs which minimize motion at the anchor (and presumably thus minimize energy transfer through the anchor) [155, 164, 126, 177, 2, 176, 97], remarkably little work has been done to quantify anchor losses. Most quantitative theoretical work on anchor loss in MEMS has focused on the case of a beam attached to an infinite plate [93, 180, 54, 181, 88]. To our knowledge, analytical work done by Hao and Ayazhi [87] and numerical work by Park and Park [138, 139] and by us [31, 30] are the only examples of quantitative prediction of anchor losses in three-dimensional MEMS structure.

1.4 Computer-aided design for resonant MEMS

Researchers have worked on specialized computer-aided design (CAD) tools for nearly three decades. For a survey of the state of the art over the first two decades, we refer particularly to the retrospective article by Senturia [149]; for a perspective on the increasing role of macromodels, we refer to [127]. The monograph by Nathan

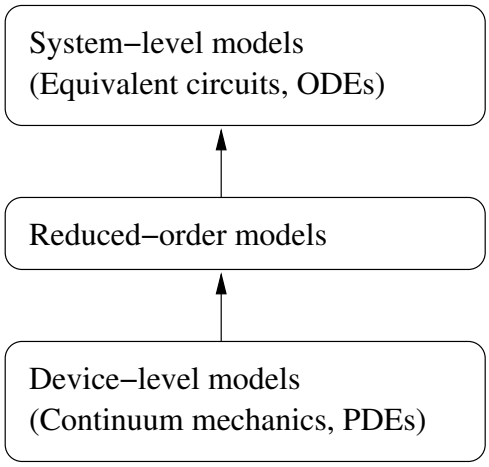

Figure 1.2. Modeling levels for MEMS design (adapted from [149])

and Baltes [130] provides an excellent survey of CAD methods for a wide variety of transducers.

There are two traditions of MEMS CAD tools: those built to work from continuum-level descriptions of the underlying physics, and those built on higher-level structural modeling ideas. The two tool sets are generally complementary, with circuit simulators providing system-level information from simple component models and finite-element analyses of individual components providing the parameters for the simplified models. Over time, automatically constructed macromodels have come to serve as a bridge between physical and structural models, and so model reduction techniques are now a major research area [127]. Major vendors of MEMS simulation systems now provide access to both types of simulations: Intellisuite [103] and CoventorWare [102] use the ABAQUS engine [1] for mechanical problems, and provide their own system simulation software; while tools in the MEMSCAP suite [125] work with ANSYS [8] to produce reduced-order device models in Verilog-A and VHDL [68].

ANSYS markets its multiphysics finite element code directly to MEMS designers [9]; ABAQUS is used by Intellisuite and CoventorWare, but is not marketed

directly as a MEMS analysis tool. The other finite element code most marketed to MEMS designers is COMSOL Multiphysics [129]. All these codes are capable of computing the resonant frequencies and mode shapes for an undamped mechanical resonator, but they provide more limited support for simulation of damping mechanisms. ANSYS, ABAQUS, and COMSOL Multiphysics all include viscoelastic material models; CoventorWare, Intellisuite, and MEMSCAP inherit these models. All these systems also provide models of gas damping effects. Finite element models of thermoelastic effects have received less attention than models of gas damping, but as of version 10.0 (August 2005), ANSYS includes elements with thermoelastic effects and supports some time-harmonic thermoelastic damping analysis [10]; prior to that, other researchers used COMSOL Multiphysics (then FEMLAB) for similar finite element calculations of thermoelastic damping [83, 67]. To our knowledge, none of these systems includes models of anchor losses.

We will be concerned with simulating individual devices more than the larger systems in which those devices operate. As different designs mature, the ultimate goal is to provide compact models that predict the behavior of these designs without recourse to finite elements and other continuum-level methods. Such compact models can then be used in a larger system simulation framework. The model reduction algorithms we describe in this dissertation form one approach to building these compact models.

Chapter 2

Mathematical preliminaries

The mathematical components of this dissertation treat various projection methods for forced and free vibration problems. We now turn to the mathematical background for these methods. Our goal is to provide a unified treatment of some existing results about finite element methods for PDEs and Krylov subspace methods for finite-dimensional systems.

These results are well-established, but are generally presented in different parts of the literature; our contribution is to present these ideas together in one place.

2.1 Eigenvalues and approximation

2.1.1 Forced response of a second-order system

We start with an autonomous single-input single-output (SISO) linear system in the time domain:

$$\begin{aligned} Mu_{tt}(t) + Ku(t) &= f(t) \\ f(t) &= f_0\phi(t) \\ y(t) &= l^*u(t). \end{aligned}$$

These equations describe a vibrating system which is driven in the direction f_0. This force creates some response in the system, and we measure this response in some direction using the functional l^*. Assuming the system starts at rest ($u(0) = \dot{u}(0) = 0$), these equations uniquely define the relation between the input signal $\phi(t)$ and the output signal $y(t)$.

If we take the Laplace transform of the time-domain equation, we get the frequency-domain equation

$$\begin{aligned} (s^2M + K)\mathcal{L}[u](s) &= \mathcal{L}[f](s) \\ \mathcal{L}[f](s) &= f_0\mathcal{L}[\phi](s) \\ \mathcal{L}[y](s) &= l^*\mathcal{L}[u](s). \end{aligned}$$

The transformed input and output signals are related algebraically as

$$\mathcal{L}[y](s) = H(s)\mathcal{L}[\phi](s), \tag{2.1}$$

where $H(s)$ is the *transfer function* defined by

$$H(s) := l^*(s^2M + K)^{-1}f_0. \tag{2.2}$$

The transfer function describes the steady-state response of the system; that is, if $\phi(t) = e^{i\omega t}$ and $H(i\omega)$ is finite, then $|y(t) - H(i\omega)e^{i\omega t}| \to 0$ as $t \to \infty$. In a physical

experiment, we can measure $H(i\omega)$ by probing the system with a sinusoidal input signal and measuring the response after any transient behavior dies away. Therefore, transfer functions will be useful to us both as a theoretical tool and as something which can be compared to experiment.

If the system (2.1) represents an undamped mechanical vibration, then M will be symmetric and positive definite and K will be symmetric and positive semi-definite. In this case, there is an M-orthonormal basis $V = [v_1, \ldots, v_n]$ which diagonalizes K; that is, $V^*MV = I$, and $V^*KV = \mathrm{diag}(\omega_1^2, \ldots, \omega_n^2)$ for real values of ω. In this case, we can write

$$H(s) = \sum_{j=1}^{n} \frac{c_j}{s^2 + \omega_j^2} \tag{2.3}$$

where

$$c_j := (l^* v_j)(v_j^* f_0). \tag{2.4}$$

The response of the system is a rational function, and we can describe it completely in terms of the position and strength of its poles, both of which we determine from an eigendecomposition. This is true even for more complicated systems: an eigenvalue problem determines the singularities, and a complete picture of the singularities gives a complete picture of the transfer function.

A complete eigendecomposition may be mathematically useful, but it is also difficult and expensive to compute. However, the role of eigenvalue analysis in much of engineering is not to obtain a complete picture of the system response, but to obtain an approximate picture which is valid in some local frequency range. In the example above, we know that if ω_j is an isolated eigenvalue, then as $\omega \to \omega_j$, the behavior of $H(i\omega)$ is dominated by the first term in a Laurent expansion,

$$H(i\omega) = -\frac{c_j}{2\omega_j(\omega - \omega_j)} + O(1). \tag{2.5}$$

More generally, to approximate $H(i\omega)$ for some frequency range $[\omega_{\min}, \omega_{\max}]$, we might build approximations based on the behavior of H at any singularities that occur near

those frequencies. For example, a typical approach to problems in structural vibration is to compute the few eigenmodes with lowest frequency, and to only consider the contributions from those modes when approximating the system response. However, this approach to approximating a transfer function is a convenience, not a necessity; there is nothing sacred about the use of an eigendecomposition.

2.1.2 General forced response and quality factors

Models of undamped vibration often have the form (2.1) with M symmetric positive-definite and K symmetric positive-semidefinite. Models with damping cannot take this form. The most common phenomenological models of damping in structures take the form

$$Mu_{tt}(t) + Bu_t(t) + Ku(t) = f(t), \tag{2.6}$$

while integral models of linear hysteretic material response lead to models of the form

$$Mu_{tt}(t) + \int_{-\infty}^{t} G(t-\tau)u(\tau)\,d\tau = f(t). \tag{2.7}$$

These are still linear time-invariant systems, which can be converted into algebraic equations through Laplace transforms. The transformed systems take the form

$$K_{\text{dynamic}}(s)\mathcal{L}[u](s) = \mathcal{L}[f](s). \tag{2.8}$$

which leads to the single-input single-output transfer function

$$H(s) = l^* K_{\text{dynamic}}(s)^{-1} f_0. \tag{2.9}$$

The singularities of $H(s)$ are now given by solutions to a *nonlinear* eigenvalue problem

$$\det(K_{\text{dynamic}}(s)) = 0. \tag{2.10}$$

As before, we seek partial information about the singularities of $K_{\text{dynamic}}(s)$ as a means to approximate $H(s)$. In particular, we are interested in isolated poles close

to the imaginary axis. Suppose $s_* = i\omega_*$ is such a pole; then for frequencies ω near to $\mathrm{Re}(\omega_*)$, we will generically have a peak in the response magnitude $|H(i\omega)|$. The shape of this peak is determined by how close ω_* lies to the real axis. We define the *quality factor* Q for ω_* to be

$$Q = \frac{|\omega_*|}{2\,\mathrm{Im}(\omega_*)}. \tag{2.11}$$

For $Q \gg 1$, the value of Q measures the shape of the peak in $|H(i\omega)|$ in the following sense. Using the first term in a Laurent expansion about ω_*, we write

$$H(s) \approx \frac{c}{s - s_*} \tag{2.12}$$

If we write $s_* = -\alpha + \beta i$, then we have

$$|H(i\omega)|^2 \approx \frac{|c|^2}{(\omega - \beta)^2 + \alpha^2}, \tag{2.13}$$

so that

$$|H(i(\beta \pm \alpha))|^2 \approx \frac{1}{2}|H(i\beta)|^2. \tag{2.14}$$

Therefore, $|H(i\omega)|^2$ has a peak centered around β and with a half-height width of 2α. The ratio of the width to the center frequency is

$$\frac{\beta}{2\alpha} \approx Q. \tag{2.15}$$

This formula is used to determine the quality factor associated with an experimentally measured peak.

2.2 Radiation damping and resonance

When $H(s)$ has isolated poles near the imaginary axis, we use partial knowledge of those poles to construct approximations of $H(i\omega)$ for some range of frequencies. When the poles of $H(s)$ are clustered, or when they are far from the imaginary axis, we prefer other means of approximation. However, a low-order rational approximation

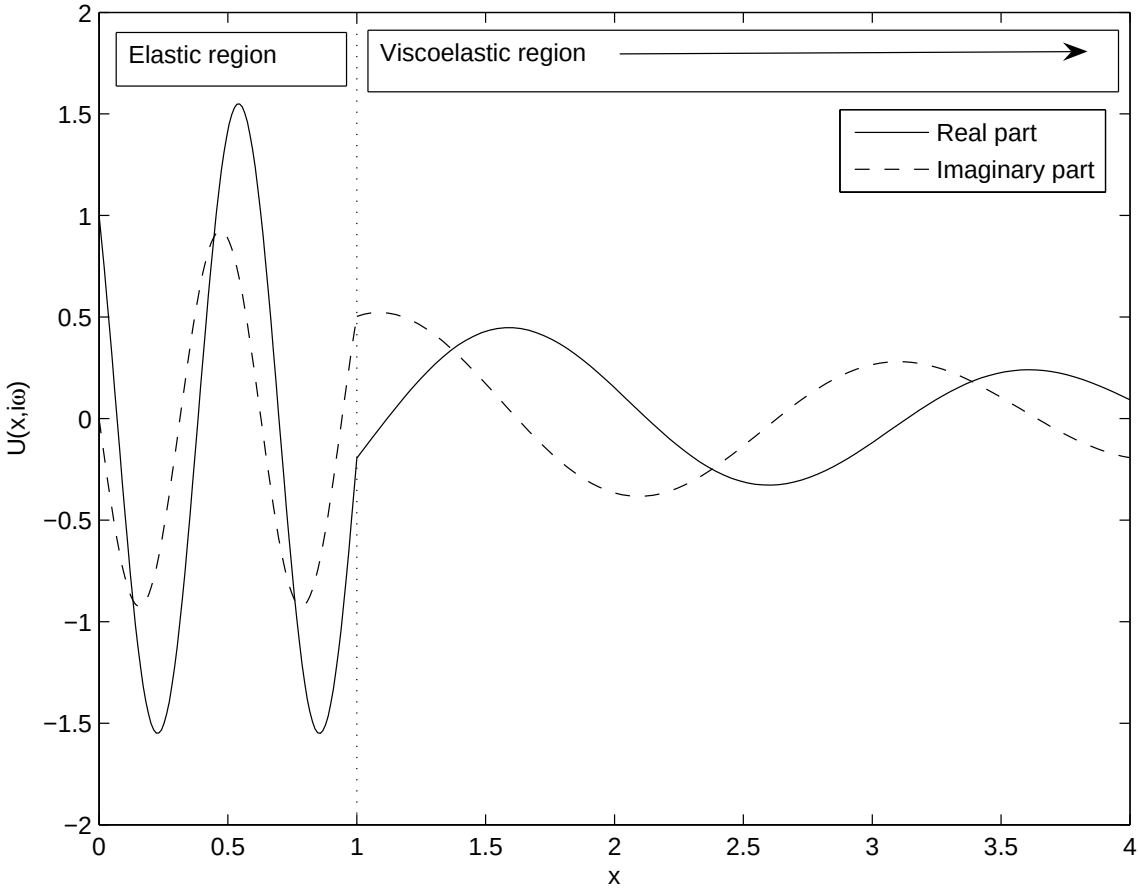

Figure 2.1. Model problem to illustrate resonance. A short elastic region on $[0, 1]$ is attached to a long viscoelastic region on $[1, L + 1]$ with very different material properties. The problem is clamped at $x = L + 1$, and subject to time-harmonic displacement conditions at $x = 0$.

of $H(s)$ may be locally quite accurate, even though the poles of the approximation appear unrelated to the poles of $H(s)$. In this event, the poles of the approximation provide more useful information than the poles of the approximated function.

In the remainder of this section, we illustrate our point with a one-dimensional PDE model (Figure 2.1). This model describes wave propagation in an inhomogeneous elastic medium with two regions of material: a short elastic region in $[0, 1]$, and a long viscoelastic region in $[1, L + 1]$. For input and output signals, we choose the displacement and force at $x = 0$; the transfer function relating these signals is the dynamic stiffness at $x = 0$. When the properties of the two regions are very different, the transfer function will have pronounced peaks; but these peaks cannot be attributed to eigenvalues in an ordinary sense. We will be able to understand the peaks more clearly by taking the limit as $L \to \infty$ and looking at hidden singularities, or resonance poles, in this limiting case. The resonance poles will then take the place of eigenvalues as the basis for constructing useful local approximations to the system behavior.

2.2.1 Viscoelastic wave solutions

We start from a one-dimensional Laplace-domain viscoelastic wave equation:

$$s^2 \rho U(x, s) \quad = \quad E(s) U_{xx}(x, s) \tag{2.16}$$

where $E(\mathbb{R}) \subset \mathbb{R}$ and any poles of E lie on the negative real axis. Typically, $E(s)$ is chosen to be a rational function. One of the simplest such viscoelastic models is Zener's model:

$$E(s) = E_0 \frac{1 + s\tau_1}{1 + s\tau_0} \tag{2.17}$$

where τ_1 and τ_0 are characteristic times associated with stress and strain relaxation. Note that $E(i\omega)$ approaches constant values for $\omega \to \infty$ and $\omega \to 0$. The fact that

$E(i\omega)$ is close to constant over large frequency ranges matters more to what follows than any particular functional form.

For fixed choice of $s = i\omega \neq 0$, solutions to the wave equation take the form

$$U(x,s) = c_1 \exp(-ikx) + c_2 \exp(ikx) \tag{2.18}$$

where the wave number $k = k(\omega)$ satisfies the dispersion relation $\omega^2\rho = E(i\omega)k^2$. The solution set determined by the dispersion relation is double-valued; by convention, we choose as the principal value for k the one such that $Im(k) < 0$, so that $\exp(-ikx) \to 0$ as $x \to \infty$. For the branch cut along the positive real k axis, we choose the solution such that $k > 0$.

2.2.2 One-dimensional model equations

We now write our one-dimensional model in the Laplace domain:

$$\begin{aligned}
s^2\rho U(x,s) &= E_1 U_{xx}(x,s),\ x \in (0,1) && (2.19)\\
s^2\rho U(x,s) &= E_2(s) U_{xx}(x,s),\ x \in (1, L+1) && (2.20)\\
U(0,s) &= \Phi(s) && (2.21)\\
U(L+1,s) &= 0 && (2.22)\\
Y(s) &= E_1 U_x(0,s). && (2.23)
\end{aligned}$$

We also require that the solution $U(x,s)$ and the stress field $EU_x(x,s)$ both be continuous with respect to x. Our goal will be to study the behavior of the transfer function $H(s)$ such that $Y(s) = H(s)\Phi(s)$.

Let k_1 and k_2 denote the wave numbers in $(0,1)$ and $(1, L+1)$, respectively. Then we may write solutions in the form

$$\hat{u}(x) = \begin{cases} d_1 \sin(k_1(x-1)) + d_2 \cos(k_1(x-1)), & x \in (0,1) \\ d_3 e^{ik_2(x-1)} + d_4 e^{-ik_2(x-1)}, & x \in (1, L+1) \end{cases} \tag{2.24}$$

where the coefficients are determined by the boundary conditions

$$d_1 \sin(-k_1) + d_2 \cos(-k_1) = \Phi(s) \tag{2.25}$$

$$d_3 e^{ik_2 L} + d_4 e^{-ik_2 L} = 0 \tag{2.26}$$

and the continuity and force balance conditions

$$d_2 = d_3 + d_4 \tag{2.27}$$

$$E_1 k_1 d_1 = iE_2 k_2 (d_3 - d_4). \tag{2.28}$$

The transfer function $H(s)$ for this problem is given by

$$H(s) = \frac{\hat{\sigma}(0,s)}{\hat{u}(0,s)} = -E_1 k_1 \frac{d_1 \cos(k_1) + d_2 \sin(k_1)}{d_1 \sin(k_1) - d_2 \cos(k_1)}. \tag{2.29}$$

Define the constants

$$\delta = e^{-2ik_2 L} \tag{2.30}$$

$$\xi = \frac{E_1 k_1}{E_2 k_2} \tag{2.31}$$

$$\tilde{\xi} = \xi \frac{1-\delta}{1+\delta}. \tag{2.32}$$

Then we can use the side conditions to find

$$\frac{d_1}{d_2} = i\tilde{\xi}, \tag{2.33}$$

so that

$$H(s) = -E_1 k_1 \frac{\cos(k_1) + i\tilde{\xi} \sin(k_1)}{\sin(k_1) - i\tilde{\xi} \cos(k_1)}. \tag{2.34}$$

2.2.3 Asymptotics of $H(s)$

As $E_1/E_2 \to \infty$, the stiffness of the viscoelastic region becomes small compared to the stiffness of the elastic region, and we recover the behavior of a free boundary at $x = 1$. In this limit, $|\xi| \to \infty$, and $H(s)$ converges pointwise almost everywhere to

$$H_{\text{free}}(s) = E_1 k_1 \tan(k_1). \tag{2.35}$$

Similarly, when $E_2/E_1 \to \infty$, the system behaves as though there were a clamped boundary at $x = 1$, and $H(s)$ converges pointwise almost everywhere to

$$H_{\text{clamp}}(s) = -E_1 k_1 \cot(k_1). \tag{2.36}$$

Thus if E_1 and E_2 are much different in magnitude, a reasonable first approximation to $H(s)$ is to model the large viscoelastic domain by a free or clamped boundary condition. However, such an approximation will be poor in the neighborhood of $k_1 = (n+1/2)\pi$ (for $|E_1| \gg |E_2|$) or $k_1 = n\pi$ (for $|E_2| \ll |E_1|$). We need more details to approximate $H(s)$ accurately near these peaks.

Now consider the limiting behavior as $L \to \infty$. By convention, we choose the principal value of k_2 such that $\text{Im}(k_2) < 0$, except when the choices for k_2 lie on the real axis. Away from this branch cut, we therefore have that $\delta \to 0$ as $L \to \infty$, and $H(s)$ asymptotically approaches the limiting function

$$H_{\text{unbounded}}(s) = -E_1 k_1 \frac{\cos(k_1) + i\xi \sin(k_1)}{\sin(k_1) - i\xi \cos(k_1)}. \tag{2.37}$$

The convergence of $H(s)$ to $H_{\text{unbounded}}(s)$ is uniform on any compact subset of the s plane which does not intersect the branch cut for k_2 (which is also a branch cut for $H_{\text{unbounded}}$) and does not contain any poles of $H_{\text{unbounded}}$.

The singularities of $H_{\text{unbounded}}$ are given by solutions to the transcendental equation

$$\sin(k_1) - i\xi \cos(k_1) = 0. \tag{2.38}$$

In general, equation (2.38) is too difficult to solve directly, but we can hope to approach it by perturbation analysis. Suppose we are in the almost-fixed case, i.e. $|\xi| \ll 1$; and further suppose that the value of ξ is nearly constant for k_1 close to $n\pi$. Then rewrite (2.38) as

$$\tan(k_1) = i\xi. \tag{2.39}$$

By letting ξ_0 denote the value of ξ at $k_1 = n\pi$ and taking the first term in a Taylor expansion for the tangent, we have

$$k_1 \approx n\pi + i\xi_0. \tag{2.40}$$

A major difficulty in the above argument is the hypothesis that ξ is nearly constant for k_1 close to $n\pi$. In particular, it may happen that the branch cut for k_2 (and hence ξ) passes between $n\pi$ and $n\pi + i\xi_0$. That is, $H_{\text{unbounded}}$ is a multi-valued function with a branch cut inherited from the branch cut for k_2. In this case, the pole whose location we have just estimated may lie on a second sheet, tucked away just behind the branch cut. Such a hidden pole is called a *resonance pole*, and the effect it has on the transfer function for real frequency values is no different from the effect of an ordinary pole close to the real axis. We may use resonance poles of an unbounded-domain approximation as the basis for approximating $H(s)$, or for computing quality factors, just as we did with the real poles.

Resonance poles are well-studied objects in classical and quantum mechanical scattering theory [132, Chapter 12], [144, pp. 51–60]. For a lively introduction to resonance poles and eigenvalues and their relation, we refer to the recent review paper by Zworski [194].

2.2.4 Summary of approximations

The sequence of approximations we just described is as follows:

1. We described an initial model consisting of a small, lossless vibrating region interacting with a large, lossy domain. We also chose a particular scalar transfer function to describe the response of the model.

2. We observed that if the properties of the small region are very different from those of the large region, then we may approximate the effects of the larger

domain by a free or fixed boundary condition. However, the transfer function obtained by this approximation has poles at real frequencies (imaginary s values), and the approximate transfer function will exhibit large relative error near those poles.

3. We observed that as the lossy domain grows larger, the transfer function approaches the transfer function for an infinite-domain problem. In this case, convergence occurs everywhere away from the singularities of the infinite-domain transfer function (including the branch cut). Therefore, for a large enough lossy domain, we can use the infinite-domain solution to approximate the original problem over a real frequency range, even near peaks in $H(i\omega)$.

4. We saw that, depending on the location of the branch cut, some of the peaks in the infinite-domain problem could be attributed to resonance poles, which we described as singularities in a second sheet of definition of the infinite-domain transfer function.

Let us consider again the implications of these last two points. As the size of the lossy domain grows in our model problem, the eigenvalues of the problems on ever-larger finite domains can converge to the resonances in the unbounded domain problem. We have noted that the finite-domain transfer functions will converge to the principal value for the infinite-domain transfer function, except where that latter has branch cuts and poles. At resonances, the finite-domain transfer functions must therefore converge to the principal value of the infinite-domain transfer function.

So by what means can the apparent effects of the resonance still lurk in the finite-domain transfer functions? For ever larger finite domains, one will have ever more dense clustering of eigenvalues close to the branch cut for the infinite-domain problem. That clustering of eigenvalues serves to discretize the influence of the continuous spectrum in the infinite-domain problem, in the same way that one might discretize

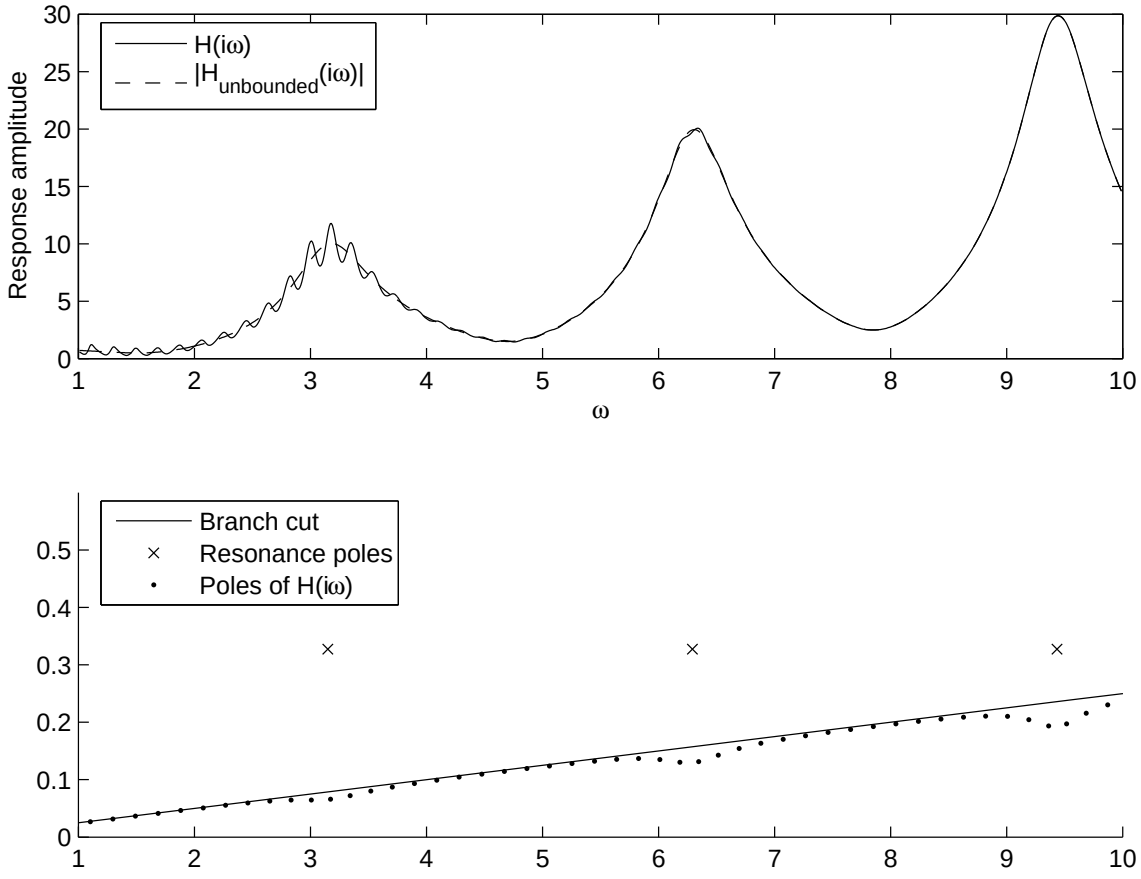

Figure 2.2. Comparison of bounded- and unbounded-domain transfer functions for a version of the test problem in Figure 2.1. For higher frequencies, $H_{\text{unbounded}}(i\omega)$ and $H(i\omega)$ are in good agreement. The poles of $H(i\omega)$ cluster along the branch cut of $H_{\text{unbounded}}(i\omega)$ in a way that mimics the effect of the resonance poles of $H_{\text{unbounded}}(i\omega)$.

the effects of a continuous charge distribution on a surface using an approximation based on a finite number of point charges. For frequency parameters sufficiently far from the cluster, the aggregate influence of the poles of the finite-domain problem takes on the character of the poles of the infinite-domain problem. We illustrate this situation in Figure 2.2.

The location of a branch cut in our infinite-domain problem was the result of a choice: we chose a particular value for k_2 which was consistent with decay in the far field, and that led us to the principal value for our infinite-domain transfer function. We could have chosen differently; and had we chosen differently, the resonances of the system would have been revealed as ordinary poles in the principal definition. One way to effect such a change in the location of a branch cut is to choose solutions which decay not for large values of x, but for large values of some modified coordinate $\tilde{x}$ which is stretched into the complex plane. We return to this notion in Chapter 3 when we discuss perfectly matched layers.

2.3 Galerkin methods

In the previous section, we considered the system

$$K_{\text{dynamic}}(s)\mathcal{L}[u](s) = \mathcal{L}[f](s), \tag{2.41}$$

and sought to understand the behavior of $\mathcal{L}[u]$ as a function of s via the singularities of $K_{\text{dynamic}}(s)$. In this section, we discuss Galerkin approximations for systems of the same general form, which we write

$$A(\lambda)u(\lambda) = f. \tag{2.42}$$

We will suppress the parameter λ unless it is explicitly needed. As before, we consider the behavior of $u(\lambda)$ as a function of λ, and also the eigenproblem of finding nontrivial

solutions (u, λ) to the homogeneous system where $f = 0$. Our goal is to provide a brief unified description of the finite element for PDEs and Krylov subspace methods which appear in later chapters.

Let $A(\lambda) : \mathcal{V} \to \mathcal{W}$ be an invertible linear operator between two Hilbert spaces. Partitioning the spaces into orthogonal complements $\mathcal{V} = \mathcal{V}_1 \oplus \mathcal{V}_2$ and $\mathcal{W} = \mathcal{W}_1 \oplus \mathcal{W}_2$, we write

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}. \tag{2.43}$$

With respect to this partitioning, we write a Galerkin solution $\hat{u} \in \mathcal{V}_1$ with the trial space $\mathcal{W}_1$ as

$$\begin{aligned} A_{11}\hat{u}_1 &= f_1 \\ \hat{u}_2 &= 0. \end{aligned}$$

When $\mathcal{V} = \mathcal{W}$ and $\mathcal{V}_1 = \mathcal{W}_1$, we call this a *Bubnov-Galerkin* approximation method. Otherwise, it is a Petrov-Galerkin method.

Assuming A_{11} is invertible, the error in the Galerkin approximation is

$$e = u - \hat{u} = \begin{bmatrix} -A_{11}^{-1}A_{12} \\ I \end{bmatrix} e_2. \tag{2.44}$$

Over all possible approximations in $\mathcal{V}_1$, the smallest possible error norm is $\|e_2\|$. Therefore, by taking norms of (2.44) we have the *quasi-optimality* property

$$\|u - \hat{u}\| \leq C_{\text{opt}} \min_{v \in \mathcal{V}_1} \|u - v\|, \tag{2.45}$$

where

$$C_{\text{opt}} := \left\| \begin{bmatrix} -A_{11}^{-1}A_{12} \\ I \end{bmatrix} \right\|. \tag{2.46}$$

Inequalities like (2.45) give us error bounds provided that we can show a uniform bound on C_{opt} (a stability property) and a bound on the error in the best approximation to u from $\mathcal{V}_1$ (a consistency property). This is the approach to error analysis

of Galerkin approximations that is usually taken in the finite element literature; see for example the development as presented in [49, 159, 38].

2.3.1 Stability for solution of linear systems

In most of our discussions of projection methods in later chapters, we focus on ways of enriching a Krylov subspace to contain better approximate solutions. In this subsection, we discuss when the Galerkin procedure will be able to choose a good approximation from a given subspace. For the moment, we do not consider the dependence on the parameter λ.

Positive operators

We now consider stability for the Bubnov-Galerkin method when A is a positive operator on $\mathcal{V}$. We call A positive (or coercive or $\mathcal{V}$-elliptic) provided there is an $\alpha > 0$ such that

$$\forall v \in \mathcal{V},\ \mathrm{Re}(v^* A v) \geq \alpha \|v\|^2. \tag{2.47}$$

Positive operators are invertible, and satisfy the bound $\|A^{-1}\| \leq \alpha^{-1}$. Moreover, in the Bubnov-Galerkin method for positive A, we also have that A_{11} and A_{22} are positive operators with the same constant α that A has. Therefore, we may write

$$C := \left\| \begin{bmatrix} -A_{11}^{-1} A_{12} \\ I \end{bmatrix} \right\| = \left\| \begin{bmatrix} -A_{11}^{-1} & 0 \\ 0 & A_{22}^{-1} \end{bmatrix} A \begin{bmatrix} 0 \\ I \end{bmatrix} \right\| \leq \alpha^{-1} \|A\|.$$

Note that this bound (Céa's inequality [38, p. 64]) is completely independent of the choice of $\mathcal{V}_1$.

When A is positive and self-adjoint, it defines an inner product; if A is bounded, the topology under the induced norm $\|\cdot\|_A$ is equivalent to the ordinary topology on $\mathcal{V}$. With respect to the A-inner product, the constant in Céa's inequality is one;

that is, Bubnov-Galerkin approximation is optimal in the A-norm. This optimality result is usually a starting point for introductions to finite element error bounds and for explanations of the conjugate gradient method. For indefinite problems like those we face, the situation becomes more complicated.

Indefinite operators

Let us again consider Bubnov-Galerkin approximation for an operator A on $\mathcal{V}$, but now without the restriction that A be positive. The *field of values* (numerical range) of A [106, p. 267] is

$$\mathcal{F}(A) := \{v^* A v : v \in \mathcal{V}\}. \tag{2.48}$$

The field of values is a convex set, and when A is real, the field of values is symmetric about the real axis. Therefore, for real problems, there are only three choices: the closure of the field of values will be strictly positive (A positive) or negative (A negative), or it must contain zero (A indefinite). If the closure of $\mathcal{F}(A)$ contains zero, then there can be no bound on $\|A_{11}^{-1}\|$ without hypotheses on $\mathcal{V}_1$. Therefore, in the indefinite case the choice of spaces affects both stability and consistency, and the two issues cannot be completely separated. Similarly, in the case of Petrov-Galerkin methods, the trial spaces $\mathcal{V}_1$ and weight spaces $\mathcal{W}_1$ must be compatible in order to ensure stability in the approximation problem.

Indefinite operators (or equivalent indefinite bilinear forms) occur regularly in the finite element literature in the context of mixed variational formulations. For such problems, the conditions necessary to ensure boundedness of A_{11}^{-1} are known as the inf-sup or Babuska-Brezzi conditions [38, Chapter 12]. In the finite-dimensional setting of Krylov subspace methods, singularity or near-singularity of the computed A_{11} is associated with breakdown in BiCG algorithm [22]. Because of the issues associated with stability in these cases, there are many finite element and Krylov subspace

methods for indefinite problems which are based on least-squares formulations or similar alternatives to the Galerkin ansatz. In this dissertation, however, we will continue to focus on Galerkin methods.

Weakly indefinite operators

The equations for the time-harmonic response of a forced structure at low frequencies (not too large relative to the fundamental frequency) are indefinite, but only mildly so. We now consider stability of Bubnov-Galerkin approximation in this case. First, suppose there is an invariant subspace $\mathcal{V}_a$ of low dimension such that for all $v \in \mathcal{V}^\perp$, $v^*Av \geq \alpha\|v\|^2$. For $\mathcal{V}_1$ containing $\mathcal{V}_a$, write $\mathcal{V}_1$ as the direct sum of orthogonal complements, $\mathcal{V}_1 = \mathcal{V}_a \oplus \mathcal{V}_b$, so that

$$A_{11} = \begin{bmatrix} A_{aa} & A_{ab} \\ 0 & A_{bb} \end{bmatrix}. \tag{2.49}$$

Let $\beta := \sigma_{\min}(A_{aa})$ and $\gamma := \|A\| \geq \|A_{ab}\|$; also recall that $\sigma_{\min}(A_{bb}) \geq \alpha$. An elementary calculation then gives

$$\sigma_{\min}(A_{11})^2 + (\alpha + \beta + \gamma)\sigma_{\min}(A_{11}) - \alpha\beta \geq 0. \tag{2.50}$$

Therefore, we have a lower bound for $\sigma_{\min}(A_{11})$ which does not depend on the choice of $\mathcal{V}_1$ beyond the hypothesis that $\mathcal{V}_a \subset \mathcal{V}_1$. We can use perturbation theory to extend to the case when $\mathcal{V}_1$ forms a small (but nonzero) angle with $\mathcal{V}_a$.

For low-frequency forced vibration problems, the smoothest modes span a "bad" subspace that contains all the indefinite behavior. Such smooth functions can be identified by comparing different norms: for example, the less smooth a function $v \in H^1$ is, the farther apart will be $\|v\|_{H^1}$ and $\|v\|_{L^2}$. Therefore, we can ensure that Bubnov-Galerkin projections with a subspace $\mathcal{V}_1$ give an operator with a uniformly

bounded inverse by a condition like

$$\forall v \in \mathcal{V} - \{0\},\ \sup_{\hat{v} \in \hat{\mathcal{V}}} \frac{|||v - \hat{v}|||}{\|v\|} < \delta \tag{2.51}$$

where $|||\cdot|||$ is a stronger Sobolev norm than $\|\cdot\|$. For an example of such a result, see [38, §5.7]. Therefore, the finite element method for forced vibration problems converges when the spaces involved satisfy mild stability conditions in addition to the usual approximability conditions. For a more detailed treatment of these results, we refer to the monograph by Chatelin [43].

2.3.2 Eigenvalue localization

In this section, we informally describe a general estimate for localizing eigenvalues of $A(\lambda)$. Our approach is based in Schur complement bounds of the same flavor as those introduced by Lehmann and by Kahan [140, Chap 10]; similar uses of Schur complements appear in the literature on quantum mechanics, where they are associated with the names of Livsic and Feshbach [96]. For a summary of more standard estimates for Galerkin eigenpair approximations in finite element methods, we refer to the survey article [17]; a more general treatment is given in the book by Chatelin [43]. Chatelin also describes the behavior of Krylov subspace methods for matrix eigenvalue problems; for further descriptions, including error analysis, we turn to Saad [146], Stewart [157], and Parlett [140].

The stability of the Galerkin method for solving linear systems and the accuracy of the Galerkin method for approximating eigenpairs are complementary issues. For quasi-optimality results, we want $A_{11}(\lambda)$ to have a uniformly bounded inverse; for eigenvalue approximation, we want to understand the relationship between values of λ for which $A_{11}(\lambda)$ is singular and those for which $A(\lambda)$ is singular. For standard eigenvalue problems $A(\lambda) = A - \lambda I$, it is relatively difficult to compute interior eigenvalues for the same reason that it is relatively difficult to ensure stability in

Galerkin solution of linear systems with highly indefinite operators: unless restrictions are placed on the choice of projection spaces, there may be spurious singularities of $A_{11}(\lambda)$ that are not close to singularities of $A(\lambda)$. This issue also affects the behavior of Krylov subspace methods for model reduction, which we will address in the next subsection.

Suppose A depends continuously on λ, and let $\Lambda(A)$ denote the spectral set

$$\Lambda(A) := \{\lambda \in \mathbb{C} : A(\lambda) \text{ does not have a bounded inverse}\}. \tag{2.52}$$

Define the Schur complement $B(\lambda)$ by

$$B(\lambda) := (A(\lambda)^{-1})_{11}^{-1} = A_{11}(\lambda) - A_{12}(\lambda)A_{22}(\lambda)^{-1}A_{21}(\lambda); \tag{2.53}$$

then

$$\Lambda(A) \cup \Lambda(A_{22}) = \Lambda(B) \cup \Lambda(A_{22}). \tag{2.54}$$

The complement $B(\lambda)$ is the sum of the projected operator $A_{11}(\lambda)$, which we use as the basis for Galerkin approximations, and a term which incorporates all of the other parts of A. To show that the eigenvalues of $A_{11}(\lambda)$ provide good estimates of some of the eigenvalues of $A(\lambda)$, we need to bound the effects of the second term in the definition of $B(\lambda)$.

For any $\epsilon > 0$, we define the pseudospectral set $\Lambda_\epsilon(A_{11})$ [166] and a set Ω_ϵ where $\|B(\lambda) - A_{11}(\lambda)\| \leq \epsilon$:

$$\Lambda_\epsilon(A_{11}) := \bigcup_{\|E\| \leq \epsilon} \Lambda(A_{11} + E) \tag{2.55}$$

$$\Omega_\epsilon := \{\lambda \in \Lambda(A_{22})^c : \|A_{12}(\lambda)A_{22}(\lambda)^{-1}A_{21}(\lambda)\| < \epsilon\}. \tag{2.56}$$

By taking norms of (2.53) to bound the terms in (2.54), we have

$$\Lambda(A) \subset \Lambda_\epsilon(A_{11}) \cup \Omega_\epsilon^c. \tag{2.57}$$

That is, inside the resolved region Ω_ϵ, any eigenvalues of A must reside within $\Lambda_\epsilon(A_{11})$. Furthermore, if A has only a point spectrum, then for any connected component Δ

of $\Lambda_\epsilon(A_{11})$ contained strictly within Ω_ϵ, the continuity argument from Gershgorin's theorem [158, Section IV.2] implies that $\Lambda(A) \cap \Delta$ and $\Lambda(A_{11}) \cap \Delta$ contain the same number of eigenvalues, counting multiplicities.

The set Ω_ϵ describes the part of $\mathbb{C}$ for which the projection $A_{11}(\lambda)$ provides complete information about $\Lambda(A)$. The parameter ϵ tells us how good the eigenvalue estimates from $A_{11}(\lambda)$ really are inside of Ω_ϵ. The bound (2.57) therefore provides us with something like a microscope for inspecting $\Lambda(A)$. As ϵ is made smaller, the resolved region Ω_ϵ will shrink, but the pseudospectral set $\Lambda_\epsilon(A_{11})$ will provide more information about any eigenvalues of A that lie inside Ω_ϵ.

We wish to make two observations regarding (2.57) in the context of this dissertation. For simplicity, we will now restrict attention to the case of a linear eigenvalue problem, i.e. $A(\lambda) = A(0) - \lambda I$.

First, note that if $\|A_{12}\|\|A_{21}\|$ is small and the spectrum of A_{11} is "well separated" from $A_{22}(\lambda)$, then we will have $\Lambda_\epsilon(A_{11}) \subset \Omega_\epsilon$. In this case, each of the eigenvalues of A_{11} corresponds to an eigenvalue of A with some small perturbation, and it makes sense to refer to $\mathcal{V}_1$ as an approximation of an invariant subspace. That is, the two crucial ingredients to good approximation of an invariant subspace are a good projection space (so that there is a small residual) and good separation between the part of the spectrum associated with the invariant subspace and the rest of the spectrum. We return to these points in Chapter 5.

Second, note that while complete information about A_{22} is usually unavailable, in some circumstances it is possible to estimate bounds on A_{22}. For example, if we know a positivity bound such as the one discussed in the previous subsection, i.e. $\forall v \in \mathcal{V}_2, v^* A v \geq \alpha \|v\|^2$, then we can provide bounds such as

$$\left\{ \lambda \in \mathbb{C} : \mathrm{Re}(\lambda) < \alpha - \frac{\|A_{12}\|\|A_{21}\|}{\epsilon} \right\} \subset \Omega_\epsilon. \tag{2.58}$$

For finite element approximations of eigenvalues, estimates like this mean we can

accurately compute low frequency eigenvalues for which the modes are sufficiently well approximated by the finite element space, and which are sufficiently separated from any under-resolved modes. However, for a Krylov subspace approximation of eigenvalues in the interior of the spectrum, any coarse bounds on Ω_ϵ will generally contain $\Lambda_\epsilon(A_{11})$. This reflects a real difficulty, namely that unless $\mathcal{V}_1$ is close to an invariant subspace, the spectrum of A_{11} may contain "imposter" eigenvalues, which do not approximate any true eigenvalue of A and which are not necessarily well separated from the "good" eigenvalue estimates. As with the case of linear systems, making a careful choice of approximation spaces can mitigate the effects of spurious singularities in the projected system; this is the motivation behind the use of harmonic Ritz vectors in eigenvalue approximation [157, Section 4.4].

2.3.3 Krylov subspace model reduction

Krylov subspace projections are used in most model reduction algorithms for large problems, as described in many survey papers [18, 72, 12, 11, 152]. In these algorithms, one approximates a transfer function

$$H(\lambda) = l^* A(\lambda)^{-1} f \tag{2.59}$$

by a the transfer function of a projected system,

$$H_1(\lambda) = l_1^* A_{11}(\lambda)^{-1} f_1. \tag{2.60}$$

The spaces $\mathcal{W}_1$ and $\mathcal{V}_1$ are chosen so as to provide a good approximation in some local range for λ. For example, to generate a reduced model valid near some expansion point λ_0, the spaces $\mathcal{W}_1$ and $\mathcal{V}_1$ might be chosen to be invariant subspaces of $A(\lambda_0)$, or Krylov subspaces for $A(\lambda_0)^{-1}$. With an appropriate choice of subspaces, one can obtain a reduced transfer function H_1 which is a Padé approximation to the original

transfer function that matches some number of moments of H at λ_0; see the discussion of the Padé-via-Lanczos connection in the survey paper [18].

The same issues discussed in the previous subsections on Galerkin eigenvalue approximation and Galerkin solution of linear systems also figure here. If neither $\mathcal{V}_1$ nor $\mathcal{W}_1$ is an invariant subspace, then H_1 may have poles which are not poles of H. If these poles of H_1 are too close to the frequency range of interest, the accuracy of the transfer function may be affected. This fact makes it difficult to provide a priori error estimates for Krylov subspace procedures, except very close to the expansion point λ_0. This has not proven a serious impediment to the adoption of Galerkin model-reduction methods. However, there are alternatives to the Galerkin method in which an approximation is chosen to satisfy some optimization condition, and these methods do not to suffer such singularities; see for example the thesis of Li [116].

Even poles of H_1 which do not affect local accuracy of the approximation may affect how well global properties of H are mimicked by H_1. For example, even if H is stable (i.e. if all the poles of H are in the left half plane and the only poles on the imaginary axis are real), the reduced model H_1 need not be stable. For many systems, one can formulate hypotheses on $\mathcal{W}_1$ and $\mathcal{V}_1$ which are sufficient to guarantee that qualitative properties, such as the stability or passivity of H, are preserved in a reduced-order model. These hypotheses are designed to preserve physically meaningful structures in the model equations. For example, the Second-Order ARnoldi (SOAR) algorithm [19], which we use in one of the example calculations in Chapter 3, preserves the structure of a model governed by a system of second-order differential equations, as well as preserving the symmetry in the system coefficient matrices. For a description of a general framework for such structure-preserving algorithms, we refer to the recent paper of Li and Bai [117].

Chapter 3

Perfectly Matched Layers

3.1 Introduction

Modern communication systems rely on high-frequency electromechanical resonators to act as frequency references and filters. Though designers currently use quartz, ceramic, and surface-acoustic wave devices, surface-micromachined micro-electromechanical system resonators (MEMS resonators) in development offer an attractive alternative. Because they can be integrated into standard complementary metal-oxide semiconductor (CMOS) technology, MEMS resonators have the potential to use less area and power, and cost less money than existing commercial devices [134]. But to be viable, energy losses in these MEMS resonators must be minimized. The usual measure of this energy loss is the *quality factor* Q of a resonant peak, defined as

$$Q = 2\pi \left(\frac{\text{Stored energy}}{\text{Energy lost per period}} \right). \tag{3.1}$$

For an ideal linear single degree of freedom oscillator $Q = |\omega|/2\,\mathrm{Im}[\omega]$, where ω is the oscillator's complex-valued eigenvalue [148, p. 158]. Resonators in cell phone filters,

for example, require Q values greater than 1000 for good performance, and higher values are preferable [134, 5].

Depending on scale, geometry, and materials, the energy losses that lower Q may come from material damping, air damping, thermoelastic damping, or radiation of elastic waves from an anchor [41]. While losses in low-frequency resonators are dominated by air damping, for which increasingly accurate compact models are available [185, 21], high-frequency disk resonators have similar measured performance in vacuum or air [176]. Thermoelastic damping is a frequently-cited source of losses at high frequencies [67, 95, 3, 118, 153]. In most cases, the damping is estimated by fitting parameters in a model originally developed by Zener [187, 188, 190]; unfortunately, this parameter-fitting makes it difficult to tell what should be attributed to thermoelastic effects and what should be attributed to other sources of damping with similar functional form. Though anchor damping is a recognized source of losses [41], there are relatively few MEMS papers (see e.g. [150, 138, 139]) dealing with losses at the anchor.

Although it is not well studied, in several designs for high MHz or GHz frequency resonators, the dominant loss mechanism appears to be radiation of elastic energy through anchors. In these designs, the resonating device is much smaller than the silicon substrate on which it sits, and waves radiating from the anchor are so attenuated by the time they reflect from the sides of the microchip that the reflected waves are negligible. That is, the bulk of the chip can be modeled without loss as a semi-infinite half-space. To simulate the response of a semi-infinite domain, one usually employs boundary dampers, infinite elements, boundary integrals, or exact Dirichlet-to-Neumann (DtN) boundary conditions so that a domain of simulation can be finite and allow for the application of finite element or finite difference methods; see e.g. [192, Chapter 8],[70, 80, 16]. Each of these methods truncates the simulation domain with an artificial boundary at which outgoing waves are absorbed. For an elastic half

space, Green's function is not known in closed form, and so highly accurate global conditions, such as DtN conditions, cannot be used. Instead we model the semi-infinite domain using a *perfectly matched layer* (PML), which absorbs waves from any angle of incidence, but which does not require knowledge of Green's function [23].

Basu and Chopra [23] demonstrated the superior performance of their PML method for problems related to earthquake engineering. Here, we examine the utility of a PML for anchor loss computations in MEMS resonators. We begin with a brief review of PMLs for time-harmonic motion. This is followed by a discussion of the relevant finite element expressions. Our presentation, while similar to that of [23], leads to a simpler implementation. We analyze the effects of discretization on the PML behavior, and give describe how to choose the PML parameters to obtain good accuracy. Because of the large computational scale of resonator problems we also investigate the use of reduced-order models that preserve the complex symmetric structure of the PML equations.

We will illustrate the effectiveness of perfectly matched layers for a MEMS example in Chapter 7.

3.2 Perfectly matched layers

Except for scale, a microresonator atop a silicon chip is much like a structure on the earth's surface during an earthquake. While we are concerned with waves radiating away from a structure and the earthquake engineer is concerned with waves radiating toward a structure, in both cases the substrate is much larger than the structure, and it can be modeled as an elastic half-space (possibly heterogeneous). This infinite-domain approximation occurs in many physical models: acoustic waves radiating from a musical instrument, electromagnetic waves reflecting from aircraft,

elastic waves scattering from a crack in a solid, and water waves in an open harbor are only a few additional examples [167], [192, Chapter 8]. The essential characteristic of the infinite-domain solution is that only outgoing waves are allowed. To model infinite-domain problems on a computer, we need finite-size discretizations which enforce this radiation condition.

One way to enforce the radiation condition is to discretize an exact boundary equation satisfied by outgoing waves. For example, outside of a sphere containing any radiators and scatterers, waves can be written as a multipole expansion; in this expansion, the radiation condition just says that certain coefficients corresponding to incoming waves should be zero. A related global condition is the DtN map, which specifies how Dirichlet conditions and Neumann conditions must be related at a surface [167]. These boundary conditions are rigorously derived and highly accurate, but they usually require that the artificial boundary have a particular shape. They are also nonlocal in space: every boundary unknown is directly related to every other boundary unknown, and consequently the matrix of boundary terms is dense, and expensive to form and to solve. Furthermore, exact boundary conditions may be unavailable for problems in which no analytically tractable Green's function is known, as in our case.

A second approach is to build approximate boundary conditions based on the asymptotic behavior of outgoing waves. These approximate conditions are local and inexpensive, but only absorb waves over a small range of angles of incidence [167, 23]. Consequently, a large computational domain may be needed for accurate results. Further, they often have difficulty with surface waves and interface waves. Yet another approach is to add a nonphysical "sponge layer" to dissipate waves before they reach the artificial boundary. Waves passing through the sponge layer are damped on the way to the artificial boundary, and are further damped when they are reflected back, so that most of the signal entering the layer is absorbed. To be effective, though, the

layer must be designed so that there is no impedance mismatch to reflect waves back from the interface between the layer and the rest of the domain.

A *perfectly matched layer* (PML) is a refinement of a sponge layer. Bérenger invented the perfectly matched layer for problems in electromagnetic wave propagation [25], and it was later re-interpreted as a complex-valued change of coordinates which could be applied to any linear wave equation [52, 168, 162]. Not only do these layers rapidly attenuate waves, they also "perfectly match" the rest of the domain; that is, there are no spurious reflections at the interface due to perfect impedance matching. In [23], a perfectly matched layer for time-harmonic elastodynamics is described which – unlike previous elastodynamic PMLs such as those in [53] – can be implemented with finite elements in a standard displacement framework, with no non-standard global unknowns. We describe an alternate interpretation of the PML described in [23], and show how our interpretation further simplifies implementation in a finite element code.

3.2.1 A motivating example

A PML model of an infinite domain problem is composed of two parts: a sub-domain where the actual equation of interest is dealt with explicitly and, a sub-domain that produces the desired effect of a far-field radiation boundary condition. To set terminology and provide insight into the workings of PMLs, we review a simple 1-D example.

1-D elastic wave

Consider a longitudinal wave propagating in a homogeneous, semi-infinite rod with axial coordinate $x \in [0, \infty)$. If waves travel with speed c, the one-dimensional

wave equation that describes this system is

$$\frac{\partial^2 u}{\partial x^2} - \frac{1}{c^2}\frac{\partial^2 u}{\partial t^2} = 0 \tag{3.2}$$

where $u(x,t)$ is the displacement. Time-harmonic solutions $u(x,t) = \hat{u}(x)e^{i\omega t}$ are governed by a Helmholtz equation

$$\frac{d^2\hat{u}}{dx^2} + k^2\hat{u} = 0, \tag{3.3}$$

where $k = \omega/c$ is the wave number and $i = \sqrt{-1}$. Solutions to this problem have the form

$$\hat{u} = c_{\text{out}}e^{-ikx} + c_{\text{in}}e^{ikx} \tag{3.4}$$

where c_{out} is the magnitude of the outgoing wave traveling from the origin toward infinity, and c_{in} is the magnitude of the incoming wave traveling from infinity toward the origin. In general, we assume there is no source at infinity, so physically meaningful solutions to such problems have $c_{\text{in}} = 0$.

1-D elastic wave in a perfectly matched medium

We now consider the Helmholtz equation (3.3) under a change of coordinates. Let $\lambda : \mathbb{R} \to \mathbb{C}$ be a continuous function which is nowhere zero, and define a new coordinate

$$\tilde{x} = \int_0^x \lambda(s)\, ds. \tag{3.5}$$

By definition, $\tilde{x}$ and x are differentially related

$$\frac{d\tilde{x}}{dx} = \lambda(x) \qquad \frac{d}{d\tilde{x}} = \frac{1}{\lambda(x)}\frac{d}{dx}. \tag{3.6}$$

Now suppose that the stretched coordinate $\tilde{x}$ is used as the independent variable in equation (3.3). Then in terms of x, the equation is

$$\frac{1}{\lambda}\frac{d}{dx}\left(\frac{1}{\lambda}\frac{d\hat{u}}{dx}\right) + k^2\hat{u} = 0. \tag{3.7}$$

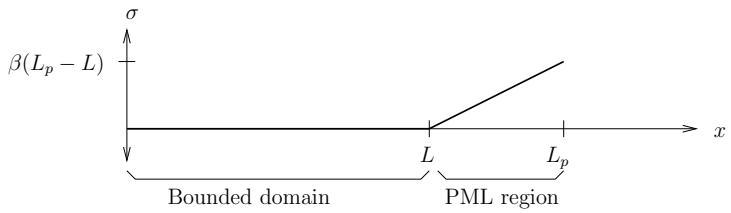

Figure 3.1. Piecewise linear attenuation function for a plane wave

where the derivative may be taken in a weak sense, since λ need not be C^1. Equation (3.7) describes wave propagation in a *perfectly matched medium* (PMM).

Suppose

$$\lambda(s) = 1 - i\sigma(s)/k; \tag{3.8}$$

then the solutions to the PMM equation (3.7) are

$$\hat{u} = c_{\text{out}} \exp\left(-\int_0^x \sigma(s)\, ds\right) \exp\left(-ikx\right) + c_{\text{in}} \exp\left(\int_0^x \sigma(s)\, ds\right) \exp\left(ikx\right). \tag{3.9}$$

So long as $\sigma = 0$, both the incoming and outgoing solutions to the PMM equation (3.7) agree with the solutions to the original Helmholtz equation (3.3). Where $\sigma > 0$, the wave decays in the direction of travel. Since the outgoing wave and the incoming wave travel in opposite directions, the outgoing wave amplitude decays with increasing x, while the incoming wave amplitude decays with decreasing x. For example, assume σ is defined to be zero on $[0, L]$ and $\sigma = \beta(s - L)$ on $[L, \infty)$. Then for $x > L$, the outgoing wave amplitude is $c_{\text{out}} \exp\left(-\beta(x - L)^2/2\right)$, and the incoming wave amplitude is $c_{\text{in}} \exp\left(\beta(x - L)^2/2\right)$.

Because waves decay so rapidly as they travel through the PMM region, we obtain a good approximation to the infinite-domain problem even if we force $\hat{u}(L_p) = 0$ for some finite $L_p > L$. This generates the concept of a *perfectly matched layer* (PML); i.e. a PML is a finite PMM attached to a region with regular wave behavior. For example, suppose we prescribe $\hat{u}(0) = 1$ and $\hat{u}(L_p) = 0$. For convenience, define

$\gamma = \beta(L_p - L)^2$; then the boundary conditions become

$$\begin{bmatrix} \hat{u}(0) \\ \hat{u}(L_p) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ e^{-(\gamma/2+ikL_p)} & e^{\gamma/2+ikL_p} \end{bmatrix} \begin{bmatrix} c_{\text{out}} \\ c_{\text{in}} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{3.10}$$

and therefore

$$c_{\text{out}} = \frac{1}{1 - e^{-\gamma-2ikL_p}} = 1 + O(e^{-\gamma}) \qquad c_{\text{in}} = \frac{-e^{-\gamma-2ikL_p}}{1 - e^{-\gamma-2ikL_p}} = -O(e^{-\gamma}). \tag{3.11}$$

Even for modest γ, the bounded-domain solution is a good approximation to the infinite domain solution. For $\gamma \approx 4.6$, only 1% of the outgoing wave is reflected. Increasing γ decreases the reflection in the continuous case; however, in the discrete equations obtained from finite difference or finite element approximations, we must be careful about how we increase γ. If β is too large, the waves entering the PML will decay rapidly, effectively creating a boundary layer; if the discretization is too coarse to resolve this decay, the numerical solution will be polluted by spurious reflections. We discuss this phenomenon and its implications further in Section 3.2.5.

3.2.2 Elastic perfectly matched layers

The multi-dimensional equations of motion for a time-harmonic elastodynamic medium with no body forces are

$$\omega^2 \rho u + \nabla \cdot \sigma = 0 \tag{3.12}$$

$$\sigma = \mathsf{C} : \epsilon \tag{3.13}$$

$$\epsilon(u) = \left(\frac{\partial u}{\partial x}\right)^s \tag{3.14}$$

where u is the displacement field, ϵ is the infinitesimal strain tensor, σ is the stress tensor, C is the material stiffness tensor, and ρ is the density. A simple isotropic elastic medium admits propagating disturbances moving at two characteristic velocities – compression waves (P waves) and shear waves (S waves). An anisotropic medium

admits further characteristic wave speeds, and inhomogeneities and interfaces add yet more wave types. However, as in the one-dimensional case, a complex-valued coordinate transformation can be used to attenuate each of these waves in the direction of travel without spurious reflections from artificial interfaces.

Multi-dimension PMM equations

Though it is possible to introduce the coordinate transformation into the local form of the equations [23], it is simpler to first recast the equations in weak form and then transform. The weak form of the time-harmonic elastodynamic equation is

$$\int_\Omega \epsilon(w) : \sigma(u)\, d\Omega - \omega^2 \int_\Omega \rho w \cdot u\, d\Omega = \int_\Gamma w \cdot t\, d\Gamma \tag{3.15}$$

where the domain is Ω, part of the boundary $\Gamma \subset \partial\Omega$ is subject to tractions t, and w is a weight function. As before, suppose $\tilde{x}$ is a transformed coordinate such that the Jacobian $\frac{\partial \tilde{x}}{\partial x} = \Lambda$ is continuously defined and everywhere nonsingular. Replacing x with $\tilde{x}$ everywhere in (3.15), we have

$$\int_{\tilde{\Omega}} \tilde{\epsilon}(w) : \tilde{\sigma}(u)\, d\tilde{\Omega} - \omega^2 \int_{\tilde{\Omega}} \rho w \cdot u\, d\tilde{\Omega} = \int_{\tilde{\Gamma}} w \cdot \tilde{\sigma}(u) \cdot \tilde{n} d\tilde{\Gamma} \tag{3.16}$$

We now map back to the x coordinate system:

$$\int_\Omega \tilde{\epsilon}(w) : \tilde{\sigma}(u)\ \det(\Lambda) d\Omega - \omega^2 \int_\Omega \rho w \cdot u\ \det(\Lambda) d\Omega = \int_\Gamma w \cdot \tilde{\sigma}(u) \cdot (\Lambda^{-T} n) \det(\Lambda)\, d\Gamma. \tag{3.17}$$

In the x coordinate system, the transformed strain and stress tensors are

$$\tilde{\epsilon}(u) = \left(\frac{\partial u}{\partial \tilde{x}}\right)^s = \left(\frac{\partial u}{\partial x}\Lambda^{-1}\right)^s \tag{3.18}$$

$$\tilde{\sigma}(u) = \mathsf{C} : \tilde{\epsilon}(u). \tag{3.19}$$

The local form of (3.17), which can be derived either from (3.17) or directly from transforming (3.12), is

$$\text{trace}\left(\frac{\partial \tilde{\sigma}(u)}{\partial x}\Lambda^{-1}\right) + \omega^2 \rho u = 0 \tag{3.20}$$

or, in indicial form,

$$\frac{\partial \tilde{\sigma}_{ij}}{\partial x_k} \left(\Lambda^{-1}\right)_{kj} + \omega^2 \rho u_i = 0. \tag{3.21}$$

3.2.3 Anisotropic medium interpretation

We now present a different way to look at the PML equations, in which the original form of the elasticity equations is maintained, but with different material coefficients. This leads to a succinct and intrinsically symmetric implementation for multi-dimensional elastic PMLs.

In indicial form, we write the strain associated with a displacement field u as

$$\epsilon_{ij}(u) = \frac{1}{2} \left(\delta_{ip}\delta_{jq} + \delta_{iq}\delta_{jp}\right) \frac{\partial u_p}{\partial x_q}. \tag{3.22}$$

The PML-transformed strain has the same form, except with one of the Kronecker δ functions replaced by Λ^{-1}:

$$\begin{aligned} \tilde{\epsilon}_{ij}(u) &= \frac{1}{2} \left(\delta_{ip}\delta_{jr} + \delta_{ir}\delta_{jp}\right) \frac{\partial u_p}{\partial x_q} \left(\Lambda^{-1}\right)_{qr} && (3.23) \\ &= \tilde{\mathsf{I}}_{ijpq} \frac{\partial u_p}{\partial x_q}, && (3.24) \end{aligned}$$

where $\tilde{\mathsf{I}}_{ijpq} := \frac{1}{2} \left(\delta_{ip} \left(\Lambda^{-1}\right)_{qj} + \left(\Lambda^{-1}\right)_{qi} \delta_{jp}\right)$. Now by substitution,

$$\begin{aligned} \tilde{\epsilon}_{ij}(w) \mathsf{C}_{ijkl} \tilde{\epsilon}_{kl}(u) &= \frac{\partial w_p}{\partial x_q} \tilde{\mathsf{I}}_{ijpq} \mathsf{C}_{ijkl} \tilde{\mathsf{I}}_{klrs} \frac{\partial u_r}{\partial x_s} && (3.25) \\ &= \frac{\partial w_p}{\partial x_q} \tilde{\mathsf{C}}_{pqrs} \frac{\partial u_r}{\partial x_s} && (3.26) \end{aligned}$$

where we define

$$\tilde{\mathsf{C}}_{pqrs} := \tilde{\mathsf{I}}_{ijpq} \mathsf{C}_{ijkl} \tilde{\mathsf{I}}_{klrs}\,. \tag{3.27}$$

Note that $\tilde{\mathsf{C}}_{pqrs}$ inherits the major and minor symmetries of C_{ijkl}. That is,

$$\mathsf{C}_{ijkl} = \mathsf{C}_{klij} \implies \tilde{\mathsf{C}}_{pqrs} = \tilde{\mathsf{C}}_{rspq} \tag{3.28}$$

$$\mathsf{C}_{ijkl} = \mathsf{C}_{jikl} \implies \tilde{\mathsf{C}}_{pqrs} = \tilde{\mathsf{C}}_{qprs}. \tag{3.29}$$

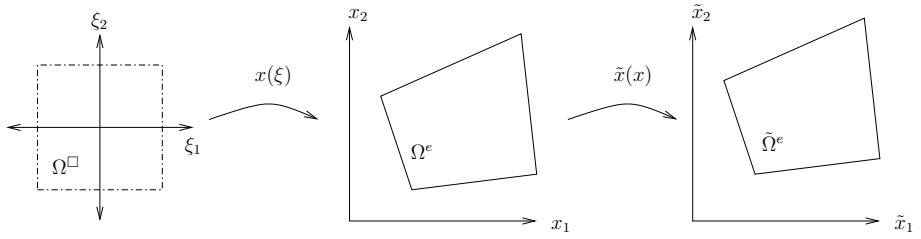

Figure 3.2. Concatenated isoparametric mapping and PML coordinate mapping

Because of the minor symmetries ($\tilde{\mathsf{C}}_{pqrs} = \tilde{\mathsf{C}}_{qprs}$ and $\tilde{\mathsf{C}}_{pqrs} = \tilde{\mathsf{C}}_{pqsr}$), we can rewrite (3.26) as

$$\tilde{\epsilon}_{pq}(w)\mathsf{C}_{pqrs}\tilde{\epsilon}_{rs}(u) = \frac{\partial w_p}{\partial x_q}\tilde{\mathsf{C}}_{pqrs}\frac{\partial u_r}{\partial x_s} = \epsilon_{pq}(w)\tilde{\mathsf{C}}_{pqrs}\epsilon_{rs}(u). \tag{3.30}$$

If we substitute (3.30) into the weak form of the PML equation (3.17), and assume that there is no loading on the transformed part of the boundary, we have

$$\int_\Omega \epsilon(w) : \tilde{\mathsf{C}}\epsilon(u)\ \det(\Lambda)d\Omega - \omega^2 \int_\Omega \rho w \cdot u\ \det(\Lambda)d\Omega = \int_\Gamma w \cdot t\, d\Gamma. \tag{3.31}$$

Now define

$$\mathsf{C}^{\mathrm{PML}} = \tilde{\mathsf{C}}\det(\Lambda) \tag{3.32}$$

$$\rho^{\mathrm{PML}} = \rho\det(\Lambda). \tag{3.33}$$

so that (3.31) becomes

$$\int_\Omega \epsilon(w) : \mathsf{C}^{\mathrm{PML}}\epsilon(u)\, d\Omega - \omega^2 \int_\Omega \rho^{\mathrm{PML}} w \cdot u\, d\Omega = \int_\Gamma w \cdot t\, d\Gamma. \tag{3.34}$$

The form of (3.34) is identical to the form of the standard elasticity equation (3.15), but with inhomogeneous, anisotropic, complex-valued material properties.

3.2.4 Finite element implementation

To derive the weak form of the PML equations in the x coordinate system, Equation (3.17), we performed a change of variables in the integrals of Equation (3.16).

Because isoparametric finite elements already use mapped integration, we can combine the change of variables associated with the PML mapping with the change of variables associated with the isoparametric coordinate transformation.

Consider the element in Figure 3.2. Suppose we choose shape functions N_I, so that we have interpolations within elements of the form $u = \sum_I N_I u_I$ and $w = \sum_I N_I w_I$. Then the nodal submatrices for the element stiffness and mass are given by

$$k_{IJ}^e = \int_{\Omega^\square} \tilde{B}_I^T D \tilde{B}_J \tilde{J} d\Omega^\square \tag{3.35}$$

$$m_{IJ}^e = \left(\int_{\Omega^\square} \rho N_I^T N_J \tilde{J} d\Omega^\square \right) \mathbf{1} \tag{3.36}$$

where $\mathbf{1}$ is the second order identity tensor and the nodal matrices $\tilde{B}_I$ come from transforming coordinates in the standard B-matrix formulation [191, Chapter 4], D is the standard matrix of material parameters, and $\tilde{J}$ is the Jacobian of the composition of the PML mapping with the isoparametric mapping:

$$\tilde{J} = \det\left(\frac{\partial x}{\partial \xi}\right) \det\left(\Lambda\right). \tag{3.37}$$

In practice, we evaluate the integrals numerically by Gaussian quadrature in the parent domain. Whether the quadrature is done analytically or numerically, the form of the integrands in (3.35) and (3.36) guarantees that the mass and stiffness matrices will be complex symmetric.

Remarks:

1. This interpretation of the PML in terms of an additional coordinate transformation works with plane stress, plane strain, axisymmetric, or three-dimensional problems. In the axisymmetric case, however, the factor of r that appears in the integrands should *not* be transformed into the PML coordinate systems, since that factor of r comes from the Jacobian of the mapping to the (r, z) coordinates, and not from the mapping to the $(\tilde{r}, \tilde{z})$ coordinates.

2. For many problems, a reasonable choice of coordinate transformations is to independently stretch each coordinate x_i, so that Λ is a diagonal matrix; i.e. $\Lambda = \mathrm{diag}(\lambda_i)$. If we further choose stretching functions so that Λ can be described by low-order polynomials, then it makes sense to also use isoparametric interpolation to compute the values of the stretching function. That is, given values for λ_i at each node, we compute $\Lambda = \mathrm{diag}(\lambda_i)$ by interpolation at the Gauss points where it is evaluated.

3. By writing the PML equations in this form we can easily institute an economy of programming where every element in a mesh is a "PML element." Regular elements are formed using the coordinate transformation $\Lambda = \mathbf{1}$ and true PML elements by $\Lambda = \mathrm{diag}(\lambda_i)$. Thus the creation of PML elements only requires a minor modification of the traditional element mapped integration routines.

3.2.5 Effects of discretization and angle of incidence

We now consider the effects of discretizing the PML. Our model problem will be a two-dimensional Helmholtz equation on $[0, L_p] \times \mathbb{R}$, where a PML transformation is applied to the x coordinate for $x \in [L, L_p]$:

$$\frac{1}{\lambda}\frac{\partial}{\partial x}\left(\frac{1}{\lambda}\frac{\partial u}{\partial x}\right) + \frac{\partial^2 u}{\partial y^2} + k^2 u = 0 \tag{3.38}$$

$$u(0, y) = \exp(ik_y y) \tag{3.39}$$

$$u(L_p, y) = 0. \tag{3.40}$$

This equation admits plane-wave type solutions of the form $u(x, y) = v(x)\exp(ik_y y)$ where v satisfies the one-dimensional PML equation

$$\frac{1}{\lambda}\frac{d}{dx}\left(\frac{1}{\lambda}\frac{dv}{dx}\right) + k_x^2 v = 0 \tag{3.41}$$

$$v(0) = 1 \tag{3.42}$$

$$v(L_p) = 0. \tag{3.43}$$

We analyzed this one-dimensional problem in Section 3.2.1. Recall that in the untransformed part of the domain, the solution $v(x)$ is a linear combination of the free-space left-traveling and right-traveling waves:

$$v(x) = c_{\text{in}} \exp(ik_x x) + c_{\text{out}} \exp(-ik_x x) \text{ for } x \in [0, L]. \tag{3.44}$$

The continuous reflection coefficient is defined as $r_{\text{continuous}} := |c_{\text{in}}/c_{\text{out}}|$. If $\lambda(x) = 1 - i\sigma(x)/k$, then the reflection coefficient has the form $\exp(-\gamma k_x/k)$, where γ is a function of the PML length and the choice of parameters. Thus plane waves traveling nearly perpendicular to the PML interface are more strongly absorbed by the PML than are waves traveling at a shallow angle.

We analyze the discrete PML in much the same way we analyzed the continuous problem. Starting with a finite element discretization of the our model problem, we use transform away the y coordinate to obtain a one-dimensional discrete system. This one-dimensional system is finite, so we can solve it numerically. In the ordinary part of the domain where the PML is not in effect, we write the solutions as a linear combination of discrete left-traveling and right-traveling waes; the ratio of the magnitude of these components defines a discrete reflection coefficient.

Our discrete model problem is illustrated in Figure 3.3. For the purpose of concreteness, we will consider a mesh of square biquadratic elements of uniform size h. Let (x_l, y_l) be the positions of the nodes in $[0, h) \times [0, h)$, i.e.

$$\begin{aligned} (x_1, y_1) &= (0, 0) \quad & (x_2, y_2) &= \left(\tfrac{h}{2}, 0\right) \\ (x_3, y_3) &= \left(\tfrac{h}{2}, \tfrac{h}{2}\right) \quad & (x_4, y_4) &= \left(0, \tfrac{h}{2}\right) \end{aligned}$$

Every node in the mesh can be written uniquely as $(x_l + ph, y_l + qh)$ for some integers p and q. Let U^l_{pq} be the field value at node $(x_l + ph, y_l + qh)$ and define $U_{pq} = (U^1_{pq}, U^2_{pq}, U^3_{pq}, U^4_{pq})^T \in \mathbb{C}^4$. Grouping together field values in this way, we write the

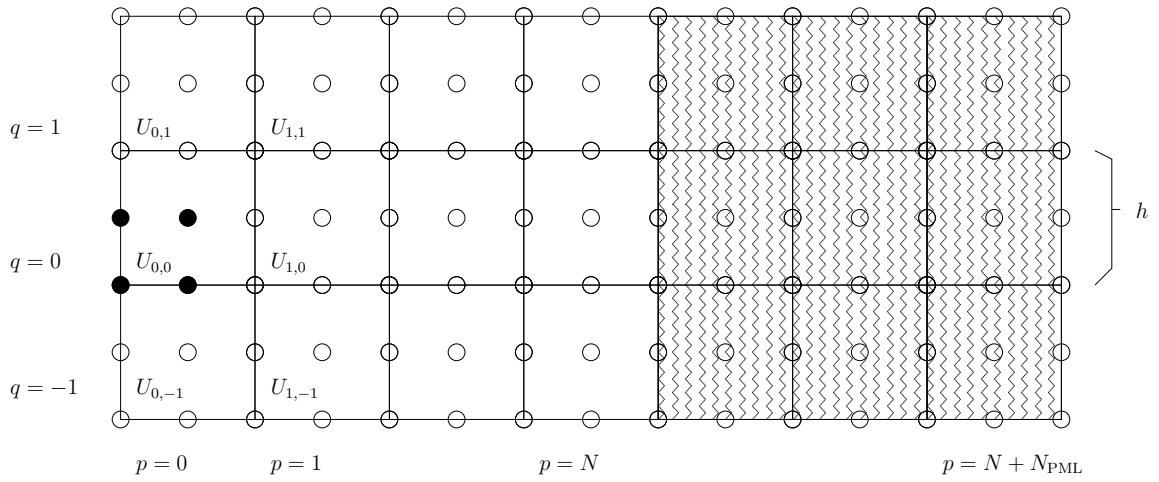

Figure 3.3. Discrete 2D plane wave test problem.

discretized Helmholtz equation in terms of a system of difference equations

$$\sum_{p,q} H_{rspq} U_{pq} = 0 \tag{3.45}$$

where $H_{rspq} \in \mathbb{C}^{4\times 4}$ represents the interaction between U_{pq} and U_{rs}. Note that $H_{rspq} = 0$ for $|r-p| > 1$ or $|s-q| > 1$. Also, because of translational invariance, H_{rspq} really only depends on $s-q$:

$$H_{rspq} = C_{rp,s-q}. \tag{3.46}$$

We now seek solutions of the form $U_{pq} = V_p \exp(ik_y qh)$. The new variable V satisfies

$$\sum_{p} \hat{H}_{rp} V_p = 0 \tag{3.47}$$

where

$$\hat{H}_{rp} := \exp(-ik_y h) C_{rp,-1} + C_{rp,0} + \exp(ik_y h) C_{rp,1}. \tag{3.48}$$

Equation 3.48 is a finite-size system of linear equations which may be solved numerically.

In the untransformed interior of the domain ($0 < p < N$), we have a constant coefficient difference equation

$$B^T V_{p-1} + A V_p + B V_{p+1} = 0 \tag{3.49}$$

where $A = \hat{H}_{pp}$ and $B = \hat{H}_{p\,p+1} = \hat{H}^T_{p\,p-1}$. Therefore for $0 < p < N$, we can write

$$V_p = \sum_m c_m \xi_m^p W_m \tag{3.50}$$

where (ξ_m, W_m) are solutions to the quadratic eigenvalue problem

$$(B^T + \xi A + \xi^2 B)W = 0. \tag{3.51}$$

For biquadratic elements, there are eight solutions to the eigenvalue problem (3.51). Two eigenvalues are zero, and two eigenvalues are infinite; these eigenvalues correspond to modes which only appear at boundaries of the discrete system. Two of the eigenvalues are purely real, and correspond to evanescent waves with a wavelength in the y direction of $h/2$. The remaining two eigenvalues are a complex conjugate pair on the unit circle. We call these eigenvalues $\xi_1 \approx \exp(-ik_x h)$ and $\xi_2 = \xi_1^H \approx \exp(ik_x h)$; they correspond to right-traveling and left-traveling discrete waves. We now define the discrete reflection coefficient to be $r_{\text{discrete}} := |c_1/c_2|$.

Because ξ_1 is an algebraic function of $k_x h$, it can only approximate the transcendental function $\exp(-ik_x h)$. Thus, there is always *numerical dispersion*; that is, for a fixed h the discrete wave speed depends on the wave number. Similarly, if the mesh size h is not uniform, there will be *numerical reflections* at interfaces where the mesh density – and consequently the discrete wave speed – changes. These spurious effects come from the error implicit in the discretization, and they vanish in the limit as $kh \to 0$. While the mesh size in our model problem is uniform in the original coordinate x, the elements are *not* uniformly sized with respect to the transformed coordinate $\tilde{x}$; it is unsurprising, then, that there is a mismatch in the discrete wave behavior at the PML interface that causes reflections.

We therefore view the discrete reflection as the sum of two parts: an interface reflection due to discretization, and a far-end reflection due to the finite termination of the PML. We can easily estimate the magnitude of both effects. To estimate the

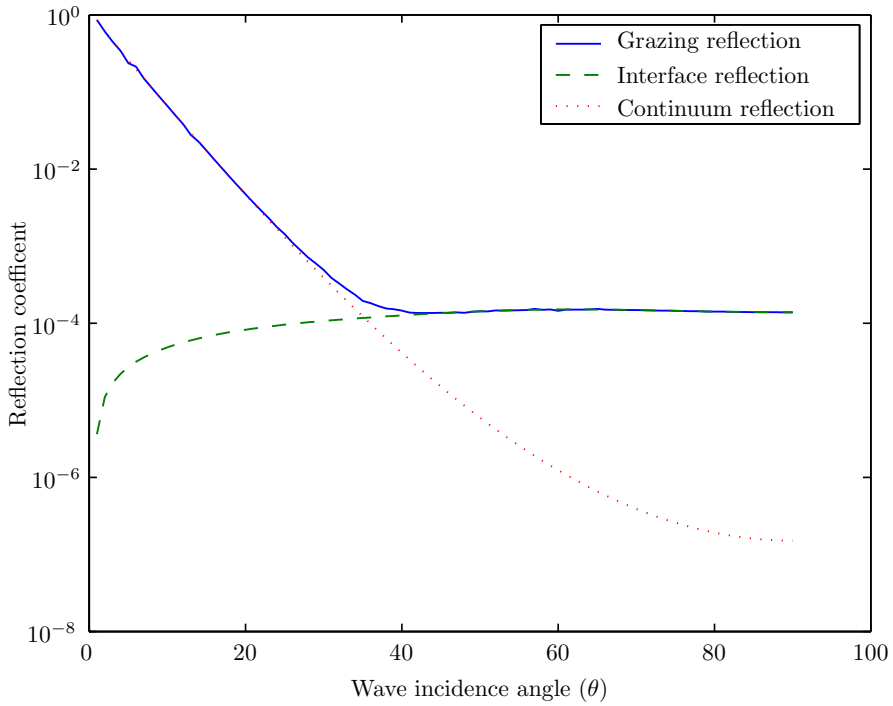

Figure 3.4. Discrete reflection coefficient r_{discrete} as a function of varying angle. The discrete reflection coefficient (solid line) is closely approximated by the sum of the continuum reflection coefficient $r_{\text{continuum}}$ (dotted line) and the reflection coefficient for one-dimensional waves of length k_x directly entering a long PML (dashed line).

far-end reflection, we use the continuum reflection coefficient $r_{\text{continuum}}$. To estimate the interface reflection, we use a PML sufficiently long that the supposed continuum reflection should be small, on the same order as the roundoff threshold. Our life is further simplified by the realization that the interface reflection depends only weakly on k_y, assuming that $k_y h$ is not too large; as with the far-end reflection, only the k_x component of the wave vector matters. That is, the reflection coefficient for a two-dimensional finite element mesh of the model problem (3.38) is nearly the same as the reflection coefficient for a one-dimensional finite element mesh of the model problem (3.41).

We illustrate the point in Figure 3.4. For a mesh of biquadratic elements, we send waves with $kh = 2\pi/10$ into a five-element PML with the parabolic stretch profile $\lambda(x) = 1 - i0.3h^{-2}(L - L_p)^2$ for $x > L$. The waves arrive at different angles, from 1 to

90 degrees; for each angle, we compute the discrete reflection coefficient r_{discrete} (solid line). At the same time, we compute the continuum reflection coefficient (dotted line) and an estimated interface reflection (dashed line). The discrete reflection coefficient is computed by launching discrete plane waves with $k_y h = 0$ and given k_x into a PML long enough that the continuum reflection would be 10^{-15}. For this problem, the estimate of r_{discrete} given by the sum of these two coefficients is never off by more than 25%.

We also note two other relevant features of Figure 3.4. The continuum reflection coefficient is a strictly monotone function of k_x: the larger k_x is, the smaller $r_{\text{continuum}}$ is. In contrast, the smaller k_x is, the smaller the interface reflection; with more elements per wavelength in the x direction, the continuum wave behavior is better resolved. Depending on the exact choice of PML parameters, the interface reflection may not be completely monotone. Nevertheless, in general r_{discrete} is dominated by interface reflection when k_x is large and far-end reflection when k_x is small.

To be more general, consider PML transformations of the form

$$\lambda(x) = \begin{cases} 1 - i\beta|x - L|^p, & x > L \\ 1 & x \leq L. \end{cases} \tag{3.52}$$

For a reasonably resolved discretization, we model $r_{\text{discrete}} \approx r_{\text{model}}$, where

$$r_{\text{model}}(k_x, \beta, L_p - L, h) := r_{\text{continuum}}(\hat{k}_x, \beta, L_p - L) + r_{\text{interface}}(\hat{k}_x h, \beta h^{-p}). \tag{3.53}$$

Here $\hat{k}_x = i \log(\xi_1)/h$ is the discrete wave speed,

$$r_{\text{continuum}} = \exp\left(-\frac{2\beta}{p+1}(L_p - L)^{p+1}\hat{k}_x\right) \tag{3.54}$$

is the reflection coefficient in the continuum case, and $r_{\text{interface}}$ is the discrete reflection for a PML with length chosen so that $r_{\text{continuum}}$ is equal to the machine unit roundoff threshold:

$$L_p - L = h\left\lceil h^{-1}\left(-\frac{p+1}{2\beta\hat{k}_x}\log\epsilon_{\text{machine}}\right)^{1/(p+1)}\right\rceil. \tag{3.55}$$

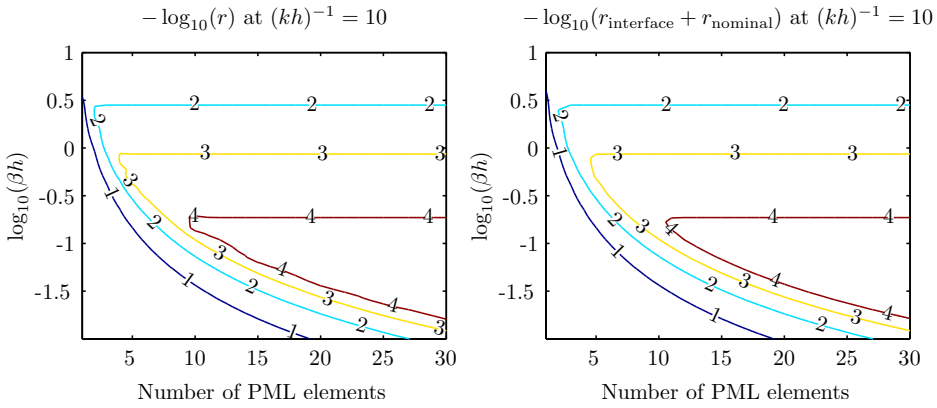

Figure 3.5. Actual (left) and estimated (right) numerical reflection for varying PML parameters at ten quadratic elements per wave.

To test this model, we computed the actual and predicted r_{discrete} for all combinations of

1. Bilinear, biquadratic, and bicubic elements
2. Linear and parabolic damping profiles
3. Angles of attack between 1 and 90 degrees, in steps of one degree
4. Mesh densities between 8 and 20 elements per wavelength, in steps of one element per wavelength)
5. Nominal reflection of $10^{-2}, 10^{-3}, \ldots 10^{-12}$ for plane waves traveling in the x direction
6. Between 1 and 10 elements through the PML

Over this range of parameters, we found that r_{model} was never more than twice r_{discrete}. In 97% of our test cases, r_{model} was within a factor of two of r_{discrete}. In the cases when r_{model} was substantially larger than r_{discrete}, we typically found that $r_{\text{continuum}}$ and $r_{\text{interface}}$ were comparable, and so the two sources or reflection canceled.

We illustrate the behavior of the discrete PML with two simple experiments with plane waves launched directly into a PML with a linear damping profile. Figure 3.5

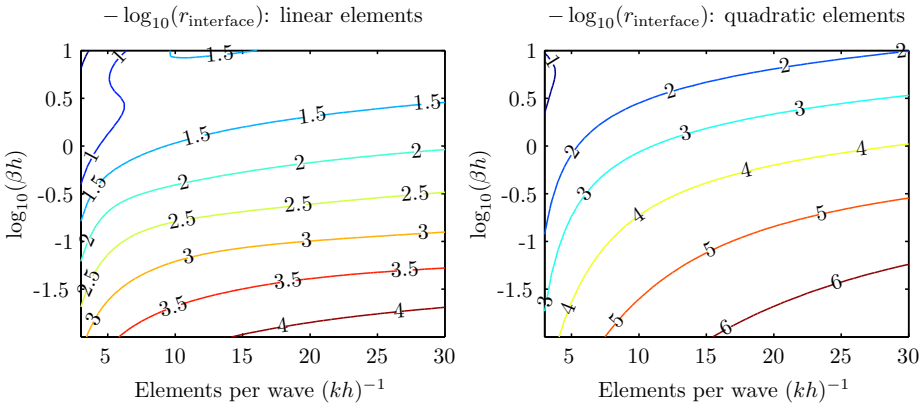

Figure 3.6. Amount of spurious reflection from a discrete PML interface for varying mesh densities and PML parameters. Both linear (left) and quadratic (right) elements are shown.

shows the estimated and actual reflection coefficients for a fixed value of kh as the PML parameter β and the PML length are varied. Notice that when the PML reaches a critical length, the interface reflection begins to dominate, and further lengthening the PML leads to no improvement in the numerical performance. Figure 3.6, we show the effects of interface reflections in the PML for linear and quadratic elements. We see from these plots that fine discretizations (either from smaller values of kh or from higher-order elements) and low values of β lead to decreased interface reflections. We also note that quadratic elements perform substantially better than linear elements.

This model for the behavior of the discrete PML suggests the following simple heuristic for choosing the PML parameters given a polynomial profile and a reflection tolerance r_{tol}. First, consider the maximum wave number in the system $k_{\max}$, and choose β so that the interface reflection for waves of length $k_{\max}$ is $r_{\text{tol}}/2$. Second, choose the PML length so that the continuum reflection is $r_{\text{tol}}/2$ for waves with the minimal propogating wave number $k_{\min}$ entering at the shallowest allowed angle $\theta_{\min}$:

$$L_p - L = h \left\lceil h^{-1} \left(-\frac{p+1}{2\beta k_{\min} \sin(\theta_{\min})} \log(r_{\text{tol}}/2) \right)^{1/(p+1)} \right\rceil . \tag{3.56}$$

Remarks:

1. The idea of measuring the behavior of discrete plane waves entering a PML goes back at least to Collino and Monk [51]. Though they worked with a simpler low-order finite difference scheme, dispersion analysis of higher-order elements is an old, standard technique [128], and in that sense our calculation of the discrete reflection coefficients is a straightforward combination of previous results. Our primary contribution is the decomposition of the discrete reflection coefficient into far-end and interface reflections, each of which can be estimated in a simple fashion.

2. Dispersion analysis is typically done with constant-coefficient differential equations or difference equations. However, the discrete PML equations do not have constant coefficients. While analyzing the behavior of waves propogating through a PML with a constant stretch function can provide useful intuition about the effects of discretization [89, 51], in practice the stretch function is never chosen to be a constant. We therefore prefer to avoid the term *dispersion* when referring to waves propagating through the PML, and instead simply refer to the numerical *reflection* caused by mismatches in the discrete wave behavior in the PML and the discrete wave behavior in the untransformed domain.

3.3 Quality factors and forced motion computations

In the MEMS problems of interest to us, we wish to compute quality factors and to compute forced motion responses. The governing equations after spatial discretization of the weak form are given by:

$$K_{\text{dyn}}(\omega)u = F\,, \tag{3.57}$$

where $K_{\text{dyn}}(\omega) := K(\omega) - \omega^2 M(\omega)$. As we noted at the end of Section 3.2.2, the PML coordinate transformations suggested in [23] are dependent on the frequency, so that the attenuation through the PML layer will be independent of the forcing frequency. This makes the system matrices dependent upon the drive frequency. Also note that the system matrices are complex symmetric. The points taken together create a somewhat involved problem. However, a number of basic observations can be used to greatly simplify the situation.

3.3.1 Quality factors via an eigencomputation

In a properly designed high quality MEMS resonator the drive pattern and frequency are always chosen to excite some resonant mode. Because of this, it suffices for many cases to simply compute the complex-valued eigenvalue of the system closest to the real-valued drive frequency. From this complex-valued eigenvalue, Q is formally defined according to Equation (3.1) as the ratio of the stored energy to the energy loss per radian, which in terms of a single mode damped oscillator can be expressed as

$$Q = \frac{|\omega|}{2\,\text{Im}(\omega)} \tag{3.58}$$

where ω is the computed eigenvalue [148].

Because the system matrices depend upon ω, the eigenvalues ω will correspond to solutions of the nonlinear eigenvalue problem $\det(K_{\text{dyn}}(\omega)) = 0$. However, when the frequency range of interest is not too wide, the parameters of the coordinate transformation may be chosen once to give acceptable attenuation over the desired range, so that the approximate dynamic stiffness for ω near a fixed reference frequency ω_0 is

$$K^0_{\text{dyn}}(\omega) := K(\omega_0) - \omega^2 M(\omega_0). \tag{3.59}$$

Finding roots such that $\det(K^0_{\rm dyn}(\omega)) = 0$ is a linear (generalized) eigenvalue problem, which we can approach using standard tools.

To find the damped eigenvalues near some specified (real-valued) reference frequency ω_0, we use a shift-and-invert Arnoldi procedure, described in standard references on numerical linear algebra [81, Chapter 9], [59, Chapter 7]. This procedure computes an orthonormal basis V for the Krylov subspace

$$\mathcal{K}_n(K_{\rm dyn}(\omega_0), u_0) = \text{span}\{u_0, K^0_{\rm dyn}(\omega_0)^{-1}u_0, \ldots, K^0_{\rm dyn}(\omega_0)^{-(n-1)}u_0\}; \tag{3.60}$$

the eigenvalues are then approximated by the eigenvalues of the much smaller problem $V^H K^0_{\rm dyn}(\omega)V$, where $(\cdot)^H$ indicates complex conjugate transpose. For computing a few isolated eigenvalues near ω_0, the main cost of the shift-and-invert Arnoldi procedure is to compute the factorization needed to apply $K^0_{\rm dyn}(\omega_0)^{-1}$. In our numerical experiments, we use UMFPACK [56] to factor the shifted matrix, and we use `eigs`, MATLAB's interface to the implicitly-restarted Arnoldi code ARPACK [114], to compute the desired eigenvalues.

3.3.2 Efficient forced motion computations

Though damped mode eigencomputations are illuminating, they do not give a complete picture of the frequency response behavior. In practice, we are interested in systems in which there is a single periodically-forced input and a single output determined by some sensed displacement. Suppose F is a time-harmonic load pattern vector, and the output is a linear function of displacement $P^T u$. Then we are really interested in computing the transfer function

$$H(\omega) = P^T K_{\rm dyn}(\omega)^{-1} F \tag{3.61}$$

which we approximate for a range of frequencies near ω_0 by

$$H^0(\omega) = P^T K^0_{\rm dyn}(\omega)^{-1} F. \tag{3.62}$$

Even if $K^0_{\text{dyn}}(\omega)$ is nearly singular at some frequency, the response amplitude $|H^0(\omega)|$ may not peak, since F may be nearly orthogonal to the forcing that drives the mode, or P may be orthogonal to the modal displacement pattern. Therefore, one normally examines $H^0(\omega)$ directly using a Bode plot.

Since the dimension N of $K^0_{\text{dyn}}(\omega)$ will be large, it is expensive to evaluate $H^0(\omega)$ directly. Instead, we construct a *reduced-order model* of dimension $n \ll N$, which we use to approximate $H^0(\omega)$. Krylov-subspace projections are often used to build reduced models of large systems [18, 11]. If we build an orthogonal basis V for a small Krylov subspace using shift-and-invert Arnoldi, then we can approximate $H^0(\omega)$ near the shift ω_0 by

$$\hat{H}^0(\omega) := (V^H P)^H (V^H K^0_{\text{dyn}}(\omega) V)^{-1} (V^H F) \approx H^0(\omega). \tag{3.63}$$

Though $\hat{H}^0(\omega)$ is often a good approximation to $H^0(\omega)$, the projected system matrix $V^H K^0_{\text{dyn}}(\omega) V$ does not preserve the complex symmetric structure of the original discretization. We can construct a symmetry-preserving reduced-order model by choosing an orthonormal projection basis W such that

$$\text{span}(W) = \text{span}([\text{Re}(V), \text{Im}(V)]). \tag{3.64}$$

Because the span of W contains the span of V, a reduced model based on W will be at least as accurate as the standard Arnoldi-based reduced model. Also, because W is a real-valued basis, projection onto the space spanned by W corresponds to a Bubnov-Galerkin discretization of the PML equation with shape functions $N_I^{\text{reduced}} = \sum_J W_{IJ} N_J$. While these facts alone might induce us to use W rather than V as a projection basis [33], we expect projection onto W to yield much better accuracy than standard Arnoldi projection, as we now describe.

If (K, M) is a Hermitian pencil and M is positive definite, then the pencil will have an orthonormal eigensystem, so that if v is a column eigenvector, v^H will be

a corresponding row eigenvector. In the study of such eigenproblems, the Rayleigh quotient

$$\rho(v) = \frac{v^H K v}{v^H M v} \tag{3.65}$$

plays a special role. When v is an eigenvector, $\rho(v)$ is a corresponding eigenvalue; and further, since ρ is stationary when and only when the argument is an eigenvector, the Rayleigh quotient produces second-order accurate eigenvalue estimates from eigenvector estimates which are only accurate to first order. For general pencils, a column eigenvector v and the corresponding row eigenvector w^H need have no such simple relationship, and so the Rayleigh quotient only provides first-order accurate eigenvalue estimates. The appropriate generalization of the Rayleigh quotient to the non-Hermitian case is $(w^H K v)/(w^H M v)$, a ratio which again yields second order accuracy (so long as the degenerate case $w^H M v \neq 0$ is avoided). When K and M are complex symmetric, we know the left and right eigenvectors are simply (non-conjugated) transposes of each other, and so we re-write the second-order accurate quotient estimate as

$$\theta(v) = \frac{v^T K v}{v^T M v}. \tag{3.66}$$

This *modified Rayleigh quotient* was used in [13] as part of a Jacobi-Davidson strategy for solving complex symmetric eigenvalue problems from PML discretizations of problems in electromagnetics.

The usefulness of having both left *and* right eigenvectors explains why model reduction methods based on nonsymmetric Lanczos iteration often approximate better than Arnoldi methods: a nonsymmetric Lanczos iteration simultaneously builds a basis for a right Krylov subspace, which typically contains good approximations for column eigenvectors; and a left Krylov subspace, which typically contains good approximations for row eigenvectors. For complex symmetric matrices, however, left and right subspaces are simply conjugates of each other, and the definition of span(W)

given above is equivalent to

$$\text{span}(W) = \text{span}([V, \text{conj}\, V]). \tag{3.67}$$

That is, if V is an Arnoldi basis for a right Krylov subspace, then both the right Krylov subspace and the corresponding left Krylov subspace are subspaces of span(W). As explained in the previous paragraph, then, the position of any eigenvalues (poles) which are estimated by vectors in V will be determined to second-order accuracy by projection onto W, where projection onto V would typically attain only first-order accuracy. For similar reasons, if P and F are proportional to each other, the estimated transfer function obtained from projecting onto W will match H^0 in $2n$ moments, rather than the n moments typical of a standard Arnoldi projection [18].

3.3.3 Conclusions

In this chapter, we have described, developed, and enhanced tools suitable for the simulation of quality factors in very high frequency MEMS resonators. The simulation of this regime of physical behavior, to date, has been largely ignored due to the analytical and numerical difficulty of estimating anchor losses which dominate such systems. The primary numerical advances made are as follows:

1. We have described an alternate interpretation of a PML for time-harmonic elasticity which was introduced in [23] and in doing so have shown how to construct a particularly simple finite element implementation for all relevant classes of analysis.

2. Through the use of one-dimensional analysis, we have elucidated the effect of discretization on the perfect matching property in order to derive heuristics for choosing the PML parameters.

3. By exploiting the complex symmetry inherent in the PML equations, we have also described how to improve the accuracy of standard methods for computing free vibrations and for building reduced models for forced frequency-response analysis.

Chapter 4

Thermoelastic Damping

4.1 Introduction

Mechanical vibrations may be damped through extrinsic losses, or transfer of energy to the external environment; and through intrinsic losses, or through transfer of energy to internal degrees of freedom. The radiation of elastic waves from an anchor is an example of an extrinsic loss; an example of intrinsic loss is the material damping that occurs due to motions of dislocations and impurities within a crystal structure. Another intrinsic loss mechanism comes from *thermoelastic* effects. Most materials expand when the temperature increases; thermodynamic consistency demands that this cannot be a distinct cause and effect, but that changes in the volume should lead to changes in temperature. Therefore pressure waves passing through a medium create spatial variations in temperature, and the relaxation of those spatial variations through heat diffusion leads to attenuation of the waves. This energy loss mechanism is called thermoelastic damping (TED). For a brief review of the role of TED, we refer to Section 1.3.3, which we partly recount here.

The importance of thermoelastic damping in solids was recognized by Zener in

the 1930s [187, 188, 190, 143]. Early designers of macroscopic mechanical resonators knew about TED [122]; and as early as 1990, Roszhart showed that Zener's model of TED in beams matched the measured quality factor for vacuum-packed MEMS cantilevers [145]. In subsequent investigations of damping in MEMS, TED has been repeatedly identified as a significant, or even dominant, contribution [41, 67, 95]. These investigations have typically relied on Zener's approximate solution for flexural waves in beams, or on refinements to Zener's approximation derived under similar kinematic assumptions about the device motion [118, 135, 153]. Gorman and Duwel have used finite-element simulations in FEMLAB to numerically solve the coupled equations of thermoelasticity [67, 83], but because of the prominent role played by flexural mechanisms in many MEMS designs, variants of Zener's approximation have dominated the literature.

In this chapter, we review the equations of thermoelasticity and their nondimensionalization. We then describe the weak form which is the basis for a finite element discretization, and show how to approximate the eigenvalues of the discretized problem using a perturbation technique. Our perturbation method generalizes Zener's approach, as we describe, and allows us to compute TED for problems with more complicated geometries than those Zener considered. We have implemented our method in `HiQLab`, and we conclude with some example test calculations.

4.2 Equations of thermoelasticity

The equations of linear thermoelasticity in the absence of body forces or heat sources are [42, 136]

$$\rho\ddot{\mathbf{u}} = \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}^T \tag{4.1}$$

$$\boldsymbol{\sigma} = \mathsf{C}\boldsymbol{\varepsilon} - \boldsymbol{\beta}\theta \tag{4.2}$$

$$\rho c_v\dot{\theta} = \boldsymbol{\nabla}\cdot(\kappa\boldsymbol{\nabla}\theta) - T_0\boldsymbol{\beta}:\dot{\boldsymbol{\varepsilon}} \tag{4.3}$$

where $u(x,t)$ is the displacement field, $T_0+\theta(x,t)$ is the temperature field, and

$$\boldsymbol{\kappa} = \text{thermal conductivity tensor} \tag{4.4}$$

$$\boldsymbol{\beta} = \text{thermal stress constitutive tensor} \tag{4.5}$$

$$T_0 = \text{reference temperature} \tag{4.6}$$

$$\mathsf{C} = \text{isothermal elasticity tensor} \tag{4.7}$$

$$c_v = \text{specific heat at constant volume} \tag{4.8}$$

In the case of an isotropic or a cubic solid, the thermal conductivity and thermal stress tensors are scalar multiples of an identity tensor:

$$\boldsymbol{\kappa} = \kappa\mathbf{1} \tag{4.9}$$

$$\boldsymbol{\beta} = \beta\mathbf{1} = \alpha_T\mathsf{C}:\mathbf{1}. \tag{4.10}$$

For isotropic problems, we may relate the thermal stress coefficient β to the ordinary thermal coefficient of expansion α_T by the relation

$$\beta = (3\lambda+2\mu)\alpha_T = \frac{E\alpha_T}{1-2\nu}, \tag{4.11}$$

where λ and μ are the Lamé parameters and E and ν are Young's modulus and Poisson's ratio.

4.3 Nondimensionalization

We now non-dimensionalize the thermoelasticity equations (4.1)-(4.3) as in [42] and [136]. For simplicity, we consider the case of a homogeneous isotropic or cubic material, for which

$$\boldsymbol{\kappa} = \kappa \mathbf{1} \tag{4.12}$$

$$\boldsymbol{\beta} = \beta \mathbf{1}. \tag{4.13}$$

We also assume that we can write the elasticity tensor as

$$\mathsf{C} = E\hat{\mathsf{C}} \tag{4.14}$$

where $\hat{\mathsf{C}}$ is dimensionless. We suppose E is Young's modulus in the isotropic case, and some other characteristic scale in the anisotropic case. Under these assumptions, we may write the thermoelastic equations as

$$\ddot{\mathbf{u}} = \boldsymbol{\nabla} \cdot \left(\frac{E}{\rho} \hat{\mathsf{C}} \epsilon - \frac{\beta}{\rho} \theta \mathbf{1} \right) \tag{4.15}$$

$$\dot{\theta} = \frac{\kappa}{\rho c_v} \nabla^2 \theta - \frac{\beta T_0}{\rho c_v} \operatorname{tr}(\dot{\boldsymbol{\varepsilon}}) \tag{4.16}$$

The constants in (4.15) and (4.16) have the following dimensions, where $[M]$, $[L]$, $[T]$, and $[K]$ denote units of mass, length, time, and temperature:

$$\left[\frac{E}{\rho} \right] = [L^2][T^{-2}] \tag{4.17}$$

$$\left[\frac{\beta}{\rho} \right] = [L^2][T^{-2}][K^{-1}] \tag{4.18}$$

$$\left[\frac{\kappa}{\rho c_v} \right] = [L^2][T^{-1}] \tag{4.19}$$

$$\left[\frac{\beta T_0}{\rho c_v} \right] = [K]. \tag{4.20}$$

In addition, we suppose a characteristic frequency ω_* with units

$$[\omega_*] = [T^{-1}]. \tag{4.21}$$

From these constants, we form the following characteristic scales for time, length, and temperature:

$$\tau_* = \omega_*^{-1} \tag{4.22}$$

$$l_* = \tau_* \sqrt{\frac{E}{\rho}} \tag{4.23}$$

$$\theta_* = \frac{T_0 \beta}{\rho c_v} \tag{4.24}$$

After expressing (4.15) and (4.16) relative to these scales, we have the dimensionless equations

$$\ddot{\mathbf{u}} = \boldsymbol{\nabla} \cdot (\mathsf{C}\boldsymbol{\varepsilon} - \xi\theta\mathbf{1}) \tag{4.25}$$

$$\dot{\theta} = \eta\nabla^2\theta - \mathrm{tr}(\dot{\boldsymbol{\varepsilon}}), \tag{4.26}$$

where

$$\xi := \frac{\beta^2 T_0}{\rho c_v E} \tag{4.27}$$

$$\eta := \frac{\omega_*}{\tilde{\omega}} \tag{4.28}$$

$$\tilde{\omega} = \frac{E c_v}{\kappa} \tag{4.29}$$

For polysilicon, typical values of ξ and $\tilde{\omega}$ are about 2×10^{-4} and 3.8×10^{12} Hz (see Figure 4.1). Thus even at gigahertz frequencies, the reduced frequency η is only about 10^{-4}.

When ξ is small, we expect the thermal fields to only weakly influence the mechanical problem. Therefore, we can use ξ as a regular perturbation parameter, as we will describe shortly. The parameter η is also small for the problems we care about. But η scales the highest-order spatial derivative in the thermal equation, and so that equation has the structure of a singular perturbation problem. Therefore when η is small, we expect boundary layers in the thermal problem.

The `HiQLab` system includes automatic support for rescaling problems, as described in Section 6.5.3. In particular, we provide routines to automatically compute

characteristic time and temperature scales for thermoelastic problems from a length scale and from properties in a material database.

4.4 Weak form and discretization

Let Ω be a bounded domain on which we wish to solve the coupled thermoelastic equations, and let $\delta\mathbf{u} : \Omega \to \mathbb{R}^3$ and $\delta\theta : \Omega \to \mathbb{R}$ be H^1 test functions on Ω. Then we multiply (4.25) and (4.26) by $\delta\mathbf{u}$ and $\delta\theta$ and integrate to get the weak forms

$$\int_\Omega \delta\mathbf{u} \cdot \ddot{\mathbf{u}}\, d\Omega + \int_\Omega \delta\boldsymbol{\varepsilon} : \hat{\mathsf{C}}\boldsymbol{\varepsilon}\, d\Omega = \xi \int_\Omega \mathrm{tr}(\delta\boldsymbol{\varepsilon})\theta\, d\Omega + \int_\Gamma \delta\mathbf{u} \cdot \mathbf{t}\, d\Gamma \tag{4.30}$$

$$\int_\Omega \delta\theta\, \dot{\theta}\, d\Omega + \eta \int_\Omega \boldsymbol{\nabla}\delta\theta \cdot \boldsymbol{\nabla}\theta\, d\Omega = -\int_\Omega \delta\theta\ \mathrm{tr}(\dot{\boldsymbol{\varepsilon}})\, d\Omega + \int_\Gamma \delta\theta\, h\, d\Gamma \tag{4.31}$$

where $\mathbf{t}$ is the surface traction and h is the surface heat flux.

We use a Bubnov-Galerkin finite element discretization of the weak equations, to obtain a discretization of the weak equations:

$$M_{uu}\ddot{u} + K_{uu}u = \xi K_{u\theta}\theta + F_u \tag{4.32}$$

$$C_{\theta\theta}\dot{\theta} + \eta K_{\theta\theta}\theta = -C_{\theta u}\dot{u} + F_\theta \tag{4.33}$$

Note that $K_{u\theta} = C_{\theta u}^T$. We may write this coupled system of second-order mechanical equations and first-order thermal equations in the second order form

$$\begin{bmatrix} M_{uu} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{u} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ C_{\theta u} & C_{\theta\theta} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} K_{uu} & -\xi K_{u\theta} \\ 0 & \eta K_{\theta\theta} \end{bmatrix} \begin{bmatrix} u \\ \theta \end{bmatrix} \tag{4.34}$$

or we may introduce the velocity field $v = \dot{u}$ to obtain the first-order form

$$\begin{bmatrix} I & 0 & 0 \\ 0 & M_{uu} & 0 \\ 0 & 0 & C_{\theta\theta} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & I & 0 \\ K_{uu} & 0 & -\xi K_{u\theta} \\ 0 & C_{\theta u} & \eta K_{\theta\theta} \end{bmatrix} \begin{bmatrix} u \\ v \\ \theta \end{bmatrix} = \begin{bmatrix} 0 \\ F_u \\ F_\theta \end{bmatrix}. \tag{4.35}$$

4.5 Perturbative eigenvalue approximation

We now wish to compute the quality factors for poles of the coupled system (4.32)-(4.33). That is, we wish to find complex frequencies ω and vectors u and θ such that

$$(-\omega^2 M_{uu} + K_{uu})u = \xi K_{u\theta}\theta \tag{4.36}$$

$$(i\omega C_{\theta\theta} + \eta K_{\theta\theta})\theta = -i\omega C_{\theta u} u. \tag{4.37}$$

In the standard approach to this problem, we would convert to first-order form:

$$\left(i\omega \begin{bmatrix} I & 0 & 0 \\ 0 & M_{uu} & 0 \\ 0 & 0 & C_{\theta\theta} \end{bmatrix} + \begin{bmatrix} 0 & I & 0 \\ K_{uu} & 0 & -\xi K_{u\theta} \\ 0 & C_{\theta u} & \eta K_{\theta\theta} \end{bmatrix} \right) \begin{bmatrix} u \\ v \\ \theta \end{bmatrix} = 0. \tag{4.38}$$

Equation (4.38) is a generalized linear eigenvalue problem, which we can solve using standard sparse eigenvalue solvers. However, we know that ξ is typically small, and so we will consider an alternative method based on a linearization of (4.36) and (4.37) about the state $\xi = 0$.

For $\xi = 0$, the resonant frequencies and modal vectors are the solution to a purely mechanical eigenvalue problem, and the corresponding temperature vector can be found by solving a linear system:

$$(-\omega_0^2 M_{uu} + K_{uu})u_0 = 0 \tag{4.39}$$

$$(i\omega_0 C_{\theta\theta} + \eta K_{\theta\theta})\theta_0 = -i\omega_0 C_{\theta u} u_0. \tag{4.40}$$

To first order, the change in ω^2 for small ξ is given by

$$-\delta(\omega^2) M_{uu} u_0 + (-\omega_0^2 M_{uu} + K_{uu})\delta u = \xi K_{u\theta}\theta_0 \tag{4.41}$$

If we multiply through by u_0^T, the second term on the left hand side vanishes, and we find

$$\delta(\omega^2) = \frac{u_0^T \xi K_{u\theta}\theta_0}{u_0^T M u_0}. \tag{4.42}$$

Therefore, we have a modified Rayleigh quotient estimate

$$\omega^2 = \frac{u_0^T \left(K_{uu} u_0 - \xi K_{u\theta} \theta_0\right)}{u_0^T M_{uu} u_0} + O(\xi^2). \tag{4.43}$$

For large Q, $|\omega| \approx |\operatorname{Re}(\omega)|$, and we may write

$$\left|\frac{\operatorname{Im}(\omega^2)}{\omega^2}\right| = \left|\frac{2\operatorname{Im}(\omega)}{\omega}\right| \left|\frac{\operatorname{Re}\omega}{\omega}\right| \approx \left|\frac{2\operatorname{Im}(\omega)}{\omega}\right| = Q^{-1}. \tag{4.44}$$

Now estimate $|\omega|$ by $|\omega_0|$ and $\operatorname{Im}(\omega)$ from the modified Rayleigh quotient to find

$$Q^{-1} \approx \left|\xi \operatorname{Im}\left(\frac{u_0^T K_{u\theta} \theta_0}{u_0^T K_{uu} u_0}\right)\right| = \left|\xi \operatorname{Im}\left(\frac{u_0^T K_{u\theta} \left(C_{\theta\theta} - i\frac{\eta}{\omega_0} K_{\theta\theta}\right)^{-1} C_{\theta u} u_0}{u_0^T K_{uu} u_0}\right)\right|. \tag{4.45}$$

In our derivation of (4.45), the only use we made of the nondimensionalization was in arguing that ξ is small. Whether or not the problem is nondimensionalized, we may use the same algorithm to compute the effect of thermoelastic damping in the usual case when the thermal variations only weakly influence the displacement fields:

1. Compute a mode u_0 with frequency ω_0 for the purely mechanical problem.
2. Compute the approximate time-harmonic temperature field θ_0 based on the mechanical solution.
3. Compute the time-harmonic forcing vector $\hat{f}$ induced by the thermal effects.
4. The imaginary part of $u_0 \cdot \hat{f}$ represents a time-averaged rate of energy dissipation, and so the ratio of $\operatorname{Im}(u_0 \cdot \hat{f})$ to the modal stiffness $u_0^T K_{uu} u_0$ is an estimate for Q^{-1}.

This approximation procedure is essentially the same as the procedure described by Zener, as we describe in the next section.

The perturbation method requires less computation than using Arnoldi or some related iteration to directly compute the eigenvalues of a linearized form like (4.35).

If we use a space of dimension N_{mech} for the mechanical degrees of freedom and a space of dimension N_{thermal} for the thermal degrees of freedom, then the perturbation method involves a real, symmetric, purely mechanical mode calculation of size N_{mech}; the solution of one complex symmetric linear system of size N_{thermal} to compute a temperature field; and a few matrix multiplications and dot products to compute the perturbed frequency estimate. In contrast, finding the eigenvalues via linearization like (4.35) requires the solution of a nonsymmetric eigenvalue problem of size $2N_{\text{mech}} + N_{\text{thermal}}$.

4.6 Relation to Zener's approach

Zener's "method of orthonormal thermodynamic potentials," as described in his monograph [189], is a special case of the perturbation expansion applied above. Zener begins by computing a purely mechanical solution, and then computes an approximation to the thermal field using a Bubnov-Galerkin method with a basis of orthonormal modes for the heat equation. He then computes the rate of energy dissipation using a formula like (4.42). Zener applies his method to find relaxation times for transverse and longitudinal vibrations of beams and rods with circular or rectangular cross-sections. Along with TED, he also treats damping mechanisms where stress depends on chemical concentration, magnetically-induced eddy currents, and other quantities governed by diffusion. In the MEMS literature, however, Zener's method is most widely identified with a formula for the quality factor of a transversely vibrating beam (see e.g. [39, Section 2.3]).

Based on our perturbation approach, we recover Zener's calculation of TED for transverse vibration of beams as follows. Suppose we have a narrow beam of width w, vibrating in flexure at a frequency of ω_0. If x is the axial direction and y the transverse direction, we impose the Euler-Bernoulli assumption that the only important stress

component is σ_{xx}, which is proportional to y. We also assume that the thermal solution takes the form

$$\theta = a_\theta \phi(y) \tag{4.46}$$

$$\phi(y) = \sqrt{\frac{2}{w}} \sin\left(\frac{\pi y}{w}\right). \tag{4.47}$$

In Galerkin terms, we choose one shape function (y) to represent the strain field, and one sinusoidal shape function ($\phi(y)$) to represent the thermal field. Therefore, for this specific case, we have simple scalars rather than finite element matrices,

$$K_{uu} = \int_{-w/2}^{w/2} y^2 \, dy = \frac{w^3}{12} \tag{4.48}$$

$$K_{u\theta} = C_{\theta u} = \int_{-w/2}^{w/2} y\phi(y) \, dy = \sqrt{\frac{8w^3}{\pi^4}} \tag{4.49}$$

$$K_{\theta\theta} = \int_{-w/2}^{w/2} \left(\frac{d\phi}{dy}\right)^2 dy = \left(\frac{\pi}{w}\right)^2 \tag{4.50}$$

$$C_{\theta\theta} = \int_{-w/2}^{w/2} \phi(y)^2 \, dy = 1. \tag{4.51}$$

We now observe that

$$\frac{K_{u\theta}^2}{K_{uu}} = 0.986... \approx 1, \tag{4.52}$$

so that the formula (4.45) becomes approximately

$$Q^{-1} \approx \left| \hat{\xi} \, \mathrm{Im} \left\{ \left(C_{\theta\theta} - \frac{i\eta}{\omega_0} K_{\theta\theta} \right)^{-1} \right\} \right| = \hat{\xi} \frac{\eta}{\omega_0} \frac{\pi^2}{w^2} \left[1 + \left(\frac{\eta}{\omega_0} \frac{\pi^2}{w^2} \right)^2 \right]^{-1} \tag{4.53}$$

where the coupling coefficient

$$\hat{\xi} := \frac{(\alpha_T E)^2 T_0}{\rho c_v E} \tag{4.54}$$

is analogous to (4.27), but with $\alpha_T E$ instead of β for consistency with Euler-Bernoulli beam theory. In [189], Zener actually expands the temperature field in a series and then notes that the approximation (4.52) holds, so that all but the first term may be neglected.

Symbol	Value	SI units	Description
ρ	2300	kilogram meter^{-3}	Mass density
E	165×10^9	Pascal	Young's modulus
ν	0.3	dimensionless	Poisson's ratio
α_T	2.6×10^{-6}	Kelvin^{-1}	Thermal coefficient of expansion
c_v	712	Joule Kelvin^{-1} meter^{-3}	Specific heat at constant volume (STP)
κ_T	30	Watt Kelvin^{-1} meter^{-1}	Thermal conductivity
T_0	293.15	Kelvin	Standard temperature
ξ	1.25×10^{-3}	dimensionless	thermomechanical coupling coefficient
$\tilde{\xi}$	2.0×10^{-4}	dimensionless	Zener's thermomechanical coupling coefficient
$\tilde{\omega}$	4.0×10^{12}	second^{-1}	Thermal characteristic frequency

Figure 4.1. Parameters for polysilicon (material `silicon2` in the `HiQLab` material database).

The main advantage of our generalization over Zener's method of orthogonal thermodynamic potentials is that our method extends easily to problems with complicated geometries. In general, it is difficult to analytically compute modes of the mechanical problem or of the thermal problem. By allowing more general approximation bases, we extend Zener's method to any problem that can be accurately discretized with finite element methods. A secondary advantage to our derivation is that we make explicit the use of ξ as a perturbation parameter, where Zener assumes first-order decoupling between the thermal and mechanical fields implicitly.

4.7 Comparisons for a beam calculation

So far, we have described three methods for computing the quality factor of a vibrating beam:

1. Discretize the thermoelastic PDEs and find eigenvalues from a linearization such as (4.38);

2. Discretize the thermoelastic PDEs and find eigenvalues by a perturbation formula such as (4.45); or

3. Use Zener's approximation formula (4.52) or some other specialized perturbation formula.

We now compare these methods for beams of varying aspect ratio.

We consider two-dimensional (plane-stress) models of cantilevered polysilicon beams 2 microns deep by 10, 20, 40, 60, 80, and 100 microns long. The beams are subject to fixed temperature and displacement boundary conditions at one end, and are free elsewhere. The material parameters used in the simulations are given in Figure 4.1. We use a finite element discretization of square bicubic elements, one micron on a side. All the eigenvalue problems are solved using shift-invert Arnoldi iteration from ARPACK [114] via MATLAB's `eigs` command with default tolerances and settings. We use an analytical estimate of the beam's first fundamental for the shift. We used UMFPACK [56] to compute the sparse LU factorization used to apply the operator for the Arnoldi iteration.

4.7.1 Effect of nondimensionalization

To find approximate eigenpairs, ARPACK searches a sequence of Krylov subspaces, beginning with an initial space generated from a random start vector. Approximate eigenpairs are extracted from these spaces by a Galerkin procedure, and when the residual norm for an eigenpair is small enough, it is considered converged (see Section 2.3). The accuracy of the converged eigenvalues depends both on the sensitivity of the eigenvalue and on the residual tolerance. If the residual tolerance is

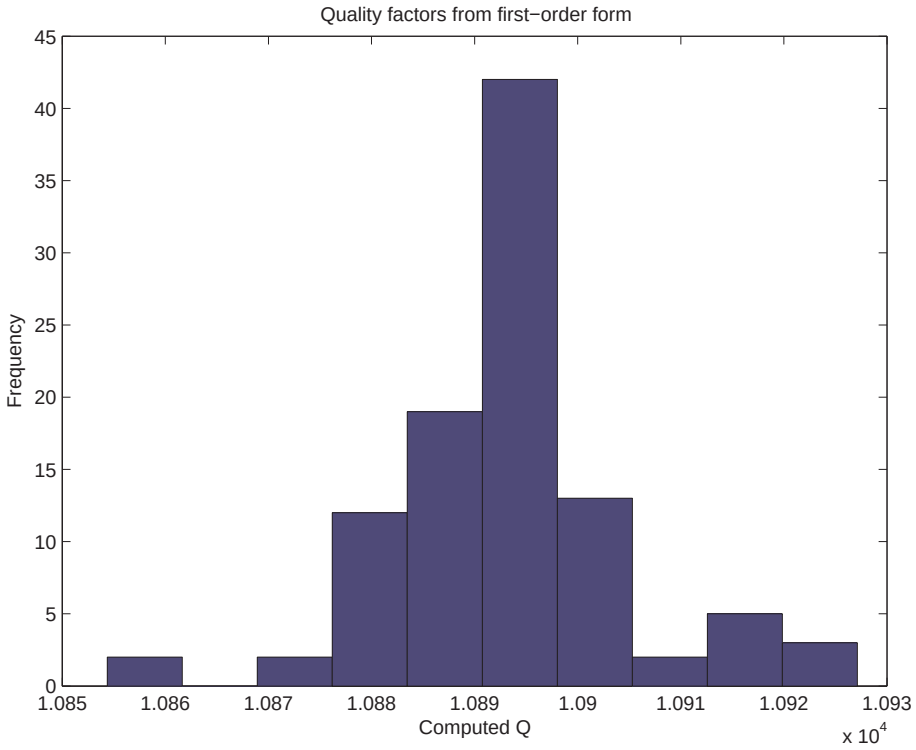

Figure 4.2. Distribution of computed quality factors for a single 20 micron polysilicon beam model when using shift-invert Arnoldi on a first-order form in SI units. The input matrices to the Arnoldi eigensolver were the same in all cases; the variation is due purely to the different choices of random starting vectors when forming the Krylov subspace.

loose, or if the eigenvalue is very sensitive, then the eigenvalues may be inaccurate. One problem with computing eigenvalues from the first-order form (4.38) is that the residual has components corresponding to displacements, velocities, and temperatures; if we are careless in our choice of units for these components, the eigensolver may exit too quickly. Thus, if we compute eigenvalues from the first-order form, it is important to first re-scale the problem, as we have done in Section 4.3. Because the perturbation method treats the mechanical and thermal fields in different steps, it is less sensitive to the choice of scales.

To illustrate the situation, we consider the behavior of the eigensolver for the 20 micron beam. We form the first-order equations in SI units and in the first-order form (4.38), and then solve each problem a hundred times. Because of the different choices of starting vectors, each run produces slightly different answers. When the problem is not scaled, these differences are large enough to lead to variations in the third digit of the computed quality factor, as we show in Figure 4.2. In contrast, the quality factors computed from the dimensionless first-order form varied in the eigthth digit.

In a similar experiment for our perturbation method, the quality factors computed from the perturbation method varied in the thirteenth digit whether the problem was posed in SI units or nondimensionalized. Note that we do not claim that the computed quality factors are accurate to thirteen places; we only claim that they are not artificially sensitive due to scaling issues.

4.7.2 Comparison of numerical results

Figure 4.3 shows the relative errors between the quality factors computed from the first-order form, from our perturbation method, and from Zener's formula. These errors are computed relative to the mean quality factor from one hundred runs of

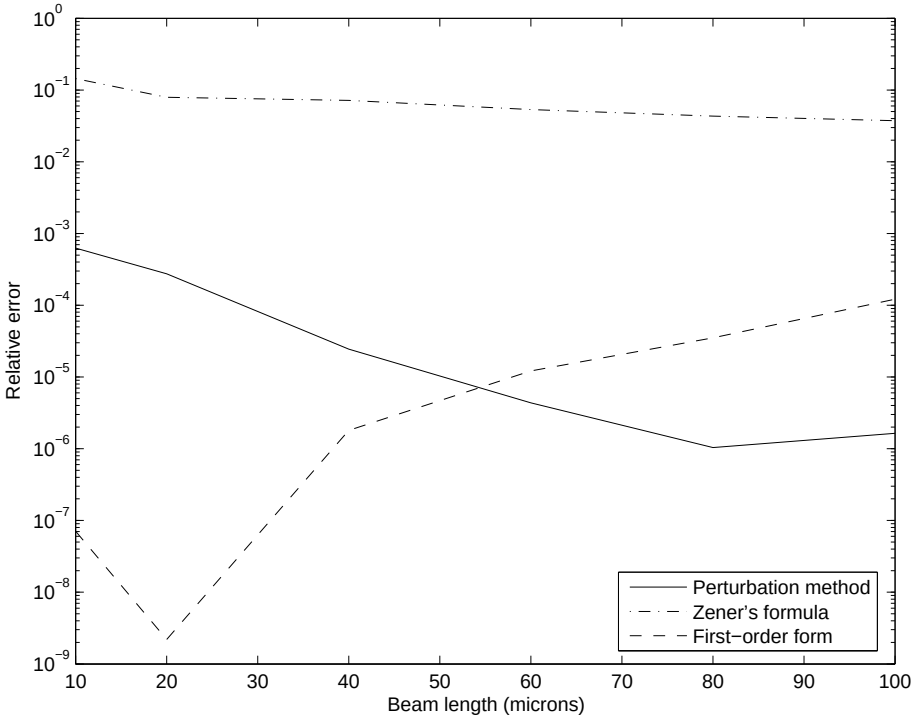

Figure 4.3. Relative errors in computed Q values for different beam lengths, using the first-order form, using our perturbation method, and using Zener's formula.

ARPACK for the scaled first-order form. We also recorded the minimum and maximum quality factors computed from the first-order form over the hundred trials; we report the range of values, scaled by the mean, as an error measure for the first-order form. For all tests, we saw agreement between the first-order form calculation and the perturbation method, with at least three digits of agreement. For the case of the 60, 80, and 100 micron tests, we note that the quality factors computed from our perturbation method lie strictly within the range of quality factors computed from the first-order form.

Zener's formula gives fair agreement with the other calculations, and the agreement gets better as the beams get longer. Zener's formula only disagreed with the other formulas by more than 10% in the case of the 10 micron beam; for this case, the aspect ratio of the beam length to width is only 5:1, and so we are pressing the limits of what we can expect from the Euler-Bernoulli beam theory on which Zener's formula is based.

4.7.3 Performance

In Figure 4.4, we compare the time to compute thermoelastic damping using the first-order form and using our perturbation method. All experiments were run on a PowerBook with a 1.67 GHz G4 processor. For the hundred micron beam, we used a mesh of 2800 free nodes; the purely mechanical eigenvalue problem therefore has 5600 unknowns, while the first-order form has 14000 unknowns. As before, the reported results are averaged over a hundred trials. For this type of problem, our perturbation method is about twice as fast as the calculation from the first-order form.

Somewhat surprisingly, scaling the first-order form not only improves the accuracy of the eigenvalue calculation; it also improves the speed. For the hundred micron beam example, computing the quality factor from the unscaled first-order form took three

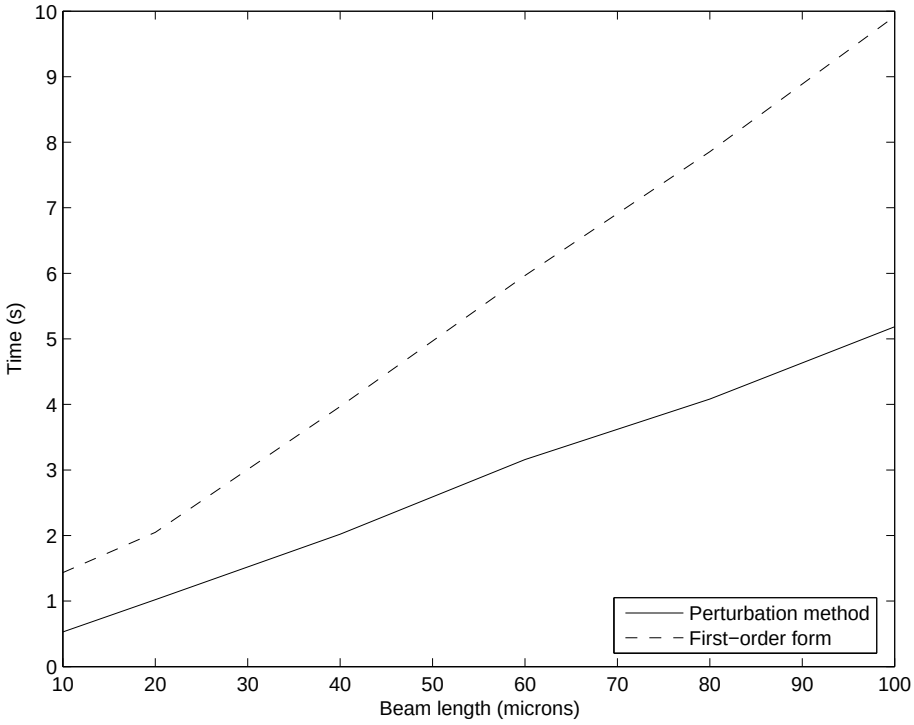

Figure 4.4. Average time to compute quality factors for beam models of different lengths, using the first-order form and the perturbation method.

minutes; this should be compared to the five seconds and ten seconds, respectively, required to compute the quality factor using the perturbation method and using the scaled first-order form. The performance difference comes from the choice of pivots in the sparse LU factorization; because of the very different scales, UMFPACK chooses pivots for stability rather than for minimal fill. Consequently, the L factor in the unscaled case has about 11 million nonzeros, while the L factor for the scaled problem has about 380 thousand nonzeros.

4.8 Conclusion

In this chapter, we have described methods for computing thermoelastic damping, including Zener's formula for TED in flexing beams, a new perturbation method which generalizes Zener's formula, and a standard method based on the eigenvalues computed from a first-order form of the differential equations. Unlike Zener's formula, our method applies to general geometries; we require only that the thermomechanical coupling be weak (as characterized by a the small size of a particular dimensionless number). This requirement is met for typical MEMS materials.

Our perturbation method is both faster and less sensitive to scaling issues than the standard calculation based on a first-order form, and the relative difference in the results of the two methods is less than 10^{-3} for the beam tests reported in this chapter. For this reason, the default TED calculation method in `HiQLab` is based on our perturbation method, though the first-order form is provided as an option. For calculations based on the first-order form, it is important that the equations be appropriately scaled; failure to choose good scales affects the solution accuracy, and can also make the solution procedure much slower. `HiQLab` provides support to automatically rescale thermoelastic problems according to the formulas described in Section 4.3.

Chapter 5

Continuation of Invariant Subspaces

5.1 Introduction

Parameter-dependent Jacobian matrices provide important information about dynamical systems

$$\frac{du}{dt} = f(u, \alpha), \text{ where } u \in \mathbb{R}^n, \ \alpha \in \mathbb{R}, \ f(u, \alpha) \in \mathbb{R}^n. \tag{5.1}$$

For example, to analyze stability at branches $(u(s), \alpha(s))$ of steady states

$$f(u, \alpha) = 0, \tag{5.2}$$

we look at the linearization $A(s) = D_u f(u(s), \alpha(s))$. If the system comes from a spatial discretization of a partial differential equation, then $A(s)$ will typically be large and sparse. In this case, an invariant subspace $\mathcal{R}(s)$ corresponding to a few eigenvalues near the imaginary axis provides information about stability and bifurcations.

Recently, we developed with collaborators the *CIS algorithm* for the continuation of invariant subspaces of a parameter-dependent matrix [60, 63, 73, 74, 32]. In this

Script capitals ($\mathcal{Z}$)	Subspaces of $\mathcal{R}^m$
San-serif capitals (S)	Operators on matrix spaces (e.g. Sylvester operators)
Standard roman capitals (Z)	Matrices and bases
$\text{Grass}(n,m)$	The Grassmann manifold of m-dimensional subspaces of $\mathbb{R}^n$
$\text{Stief}(n,m)$	The Stiefel manifold of orthogonal bases of elements of $\text{Grass}(n,m)$
$O(n)$	Orthogonal matrices in $\mathbb{R}^{n\times n}$
$f(u,\alpha)$	Right-hand side in a dynamical system $\frac{du}{dt} = f(u,\alpha)$
$(u(s),\alpha(s))$	A branch of equilibria of $\frac{du}{dt} = f(u,\alpha)$
$A(s)$	A parameter-dependent matrix ($A:\mathbb{R}\to\mathbb{R}^{n\times n}$). Typically, $A(s) = D_u f(u(s),\alpha(s))$.
$A[s_0,s_1] = \frac{A(s_0)-A(s_1)}{s_0-s_1}$	Newton divided difference of A
$\mathcal{R}(s)$	A continuous maximal invariant subspace of $A(s)$
$Q(s) = \begin{bmatrix} Q_1(s) & Q_2(s)\end{bmatrix}$	A continuous basis for $\mathbb{R}$ such that $\mathcal{R}(s) = \text{span}(Q_1(s))$
$T(s) = \begin{bmatrix} T_{11}(s) & T_{12}(s) \\ 0 & T_{21}(s)\end{bmatrix}$	A continuous block Schur factor: $A(s)Q(s) = Q(s)T(s)$
$Y(s) = Q_2(s_0)^T Q_1(s)$	Riccati equation unknown
$\bar{Q}_1(s) = Q(s_0)\begin{bmatrix} I \\ Y(s)\end{bmatrix}$	Alternately normalized basis for $\mathcal{R}(s)$
$\hat{T}(s) = \begin{bmatrix} \hat{T}_{11} & \hat{T}_{12} \\ \hat{E}_{11} & \hat{T}_{21}\end{bmatrix}$	Approximate Schur form at s near s_0: $\hat{T}(s) = Q(s_0)^T A(s) Q(s_0)$
$\Lambda(s) = \{\lambda_i\}_{i=1}^n$	The spectrum of $A(s)$
$\Lambda_1(s) = \{\lambda_i\}_{i=1}^m$	The spectrum of $A(s)\|_{\mathcal{R}(s)}$
$\Lambda_2(s) = \{\lambda_i\}_{i=m+1}^n$	The spectrum of $A(s)\|_{\mathcal{R}(s)^\perp}$
$P(s)$	A (skew) eigenprojector associated with $\mathcal{R}(s)$
$\mathsf{S}Y = YA_{11} - A_{22}Y$	Sylvester operator associated with A ($A_{ij} = Q_i^T A Q_j$)
$\text{sep}(B,C)$	The smallest singular value of the Sylvester map $X \to BX - XC$
$\mathcal{V}$	A projection space for Galerkin approximation
V	An orthonormal basis for $\mathcal{V}$

Figure 5.1. Table of notation

report, we extend the CIS algorithm to make it more suitable to numerical bifurcation analysis. Our goal is to extend numerical bifurcation techniques developed for small systems to larger systems. We also wish to ensure that bifurcations are detected reliably; this goal becomes especially relevant for non-normal matrices, where a small perturbation of a matrix may result in a large change to its eigenvalues [165, 169]. To this end, we make the following contributions to the development of the method: we derive new sufficient conditions for the existence of a continuously-defined invariant subspace; we introduce logic to adapt or reinitialize the subspace during continuation so that it is always well-defined and always includes information relevant to bifurcation analysis; we extend the algorithm to use Galerkin projection methods when n is large and direct methods are expensive; and we integrate our method into the MATCONT bifurcation analysis tool [61].

The CIS algorithm consists of a predictor based on first derivative information, and a corrector based on iterative refinement of an approximate invariant subspace (see [156], [58] and references therein). The algorithm evaluates a smoothly varying orthonormal basis for $\mathcal{R}(s)$ at sample points $s_0 < s_1 < \ldots < s_{N-1} < s_N$. This basis approximately minimizes arclength over all orthonormal bases for $\mathcal{R}(s)$, in a sense we will make precise in Section 5.2.1. The step size is adapted so that $h_i = s_i - s_{i-1}$ decreases when $\mathcal{R}(s)$ changes fast and increases when $\mathcal{R}(s)$ changes slowly. When the eigenvalues corresponding to $\mathcal{R}(s)$ come too near the rest of the spectrum, the continuation procedure breaks down. In this case, the size of the continued subspace is adapted, and continuation proceeds with a larger or smaller subspace.

The rest of the paper is organized as follows. After discussing related work in the remainder of this section, we turn to the theory of existence and uniqueness of continuously-defined invariant subspaces and prove our new result on sufficient conditions for existence in Section 5.2. In Section 5.3, we describe the CIS algorithm and our new algorithms for initializing and updating the invariant subspace during the

continuation process. In Section 5.4, we describe how to modify the CIS algorithm to use projection methods; and in Section 5.5, we illustrate the usefulness of the modified algorithm in bifurcation analysis through the solution of a model problem in MATCONT. We conclude and present our plans for future work in Section 5.6.

5.1.1 Related work

The local behavior of eigendecompositions and other matrix factorizations when viewed as matrix functions is of long-standing interest, and is treated in detail in the book by Stewart and Sun [158], as well as in the authoritative tome of Kato [106]. The local behavior of invariant subspaces can be analyzed by representing the subspaces near some reference subspace in terms of an orthogonal departure from that reference space; such analysis leads directly to an algebraic Riccati equation. In [58], this Riccati equation was used as the basis for a unified analyis of several algorithms for refining approximate invariant subspaces; and in more recent work [37], new algorithms for invariant subspace approximation are proposed which combine a Galerkin approximate solution to an algebraic Riccati equation with the subspace construction ideas of the Jacobi-Davidson algorithm. In [69], Edelman and his colleagues proposed a more global approach to the analysis of linear algebra algorithms based on Grassmann manifolds and Stiefel manifolds (manifolds of subspaces and of orthonormal subspace bases, respectively); this approach has inspired several new methods for invariant subspace refinement, four of which are summarized and analyzed in [4].

No algorithm can produce globally continuous eigendecompositions, even for the set of diagonalizable matrices. However, one can smoothly define an invariant subspace basis along a path through matrix space, assuming the path crosses no singularities that would render the subspace discontinuous. In [62], a variety of continuous eigendecompositions for one-parameter matrix functions are described, including

continuous Schur and block Schur decompositions. In a paper by Govaerts, Guckenheimer, and Khibnik [86] which motivated our work on invariant subspace continuation, a low-dimensional invariant subspace of the Jacobian matrix, corresponding to the eigenvalues with largest real parts, was computed at each point along a continuation path and used to detect Hopf bifurcations via the bialternate matrix product. The authors concluded that subspace reduction can be combined with complicated bifurcation computations and should be tried for large problems.

The CIS algorithm was presented and analyzed in [60] and further studied in [63], [73], with additional practical developments in [74] and [32]. The algorithm of [63] constructs a smooth block 2-by-2 Schur decomposition; in [65], the approach is extended to the case of more blocks, and a new method is proposed to compute a smooth similarity reduction to block bidiagonal form. In [64], the approach described in [60] for using subspace continuation to compute connecting orbits between equilibria was extended to compute connecting orbits between periodic orbits. To continue low-dimensional invariant subspaces of sparse matrices, the authors of [27] use a bordered Bartels-Stewart algorithm to solve each corrector iteration; in [36], this approach is combined with ideas from [63, 73]. Though [27] and [36] deal with methods for sparse matrices, they differ from our current work in that they use different predictors and correctors, and they do not analyze and update the subspace during continuation to ensure it retains all information relevant to bifurcations.

Numerical continuation for large nonlinear systems arising from ODEs and discretized PDEs is an active area of research, and the idea of subspace projection is common in many methods being developed. The continuation algorithms are typically based on Krylov subspaces, or on recursive projection methods which use a time integrator instead of a Jacobian multiplication as a black box to identify the low-dimensional invariant subspace where interesting dynamics take place; see e.g. [14, 151, 84, 44, 85, 66, 78, 40, 50], and references there.

5.2 Continuous invariant subspaces

Let $A \in C^k([0,1],\ \mathbb{R}^{n\times n})$ be a k-times continuously differentiable parameter-dependent matrix. We can write the spectrum $\Lambda(s)$ of $A(s)$ as n continuous functions $\lambda_1(s), \ldots, \lambda_n(s)$ [106]. At parameter values where $\lambda_i(s)$ is a multiple eigenvalue, $\lambda_i(s)$ may not be differentiable, and it may be impossible to define a continuous right eigenvector. However, $\lambda_i(s)$ is a C^k function with a C^k right eigenvector as long as $\lambda_i(s)$ has algebraic multiplicity 1. More generally, define

$$\begin{aligned} \Lambda_1(s) &:= \{\lambda_i(s)\}_{i=1}^{m} \\ \Lambda_2(s) &:= \{\lambda_i(s)\}_{i=m+1}^{n} \\ \Lambda(s) &:= \Lambda_1(s) \cup \Lambda_2(s) \end{aligned} \quad . \tag{5.3}$$

While $\Lambda_1(s)$ and $\Lambda_2(s)$ remain disjoint, there is a well-defined maximal right invariant subspace $\mathcal{R}(s)$ corresponding to $\Lambda_1(s)$, and $\mathcal{R}(s)$ is C^k. There are several ways to prove this fact; each provides a useful perspective.

In what follows, we will primarily use the Frobenius matrix norm: $\|A\|_F = \sqrt{\mathrm{tr}(A^T A)}$. We also assume that complex conjugate pairs are not split between Λ_1 and Λ_2.

5.2.1 The geometry of subspaces

We begin with a brief review of the geometry of subspaces and orthonormal bases (see [69] for a more complete treatment). The *Stiefel manifold* $\mathrm{Stief}(n,m)$ is the set of matrices with orthonormal columns:

$$\mathrm{Stief}(n,m) := \{Z \in \mathbb{R}^{n\times m} : Z^T Z = I\} \tag{5.4}$$

where $m \leq n$. We can also write

$$\mathrm{Stief}(n,m) = \{QI_{n,m} : Q \in O(n), I_{n,m} = \text{leading } m \text{ columns of } I_n\} \tag{5.5}$$

Well-known examples of Stiefel manifolds are the unit sphere (for $m = 1$) and the orthogonal group $O(n)$ (for $m = n$).

The *Grassmann manifold* $\text{Grass}(n, m)$ is the set of all m-dimensional subspaces of $\mathbb{R}^n$. We represent elements of $\text{Grass}(n, m)$ by equivalence classes of members of $\text{Stief}(n, m)$ spanning the same space. That is,

$$\text{Grass}(n, m) = \text{Stief}(n, m)/\left[Z \sim ZU,\, U \in O(m)\right]. \tag{5.6}$$

Grassmann manifolds are more difficult to picture than Stiefel manifolds are, since they are not naturally subsets of a Euclidean space of matrices.

The tangent directions at $Q_0 \in O(n)$ are translations of the skew symmetric matrices. For any $Q \in O(n)$ near Q_0,

$$Q = Q_0 + Q_0 H + \text{higher order terms, where } H = -H^T. \tag{5.7}$$

The tangents to $\text{Stief}(n, m)$ have a related structure. If $Z_0 = Q_0 I_{n,m} \in \text{Stief}(n, m)$ for $Q_0 \in O(n)$, then for any nearby $Z \in \text{Stief}(n, m)$,

$$Z = Z_0 + Q_0 H I_{n,m} + \text{higher order terms, where } H = -H^T \tag{5.8}$$

In block form, these tangent directions look like $Q_0 \begin{bmatrix} H_{11} \\ H_{21} \end{bmatrix}$, where $H_{11} \in \mathbb{R}^{m \times m}$ is skew-symmetric and $H_{21} \in \mathbb{R}^{(n-m) \times m}$ is arbitrary.

The tangent space at $Z_0 \in \text{Stief}(n, m)$ is a direct sum of two orthogonal spaces: the vertical space and the horizontal space (Figure 5.2). The *vertical space* is

$$\{\Delta Z \in \mathbb{R}^{n \times m} : \Delta Z = Z_0 H_{11} \text{ and } H_{11} \in \mathbb{R}^{m \times m} \text{ is skew}\}, \tag{5.9}$$

and the *horizontal space* is

$$\{\Delta Z \in \mathbb{R}^{n \times m} : Z_0^T \, \Delta Z = 0\}. \tag{5.10}$$

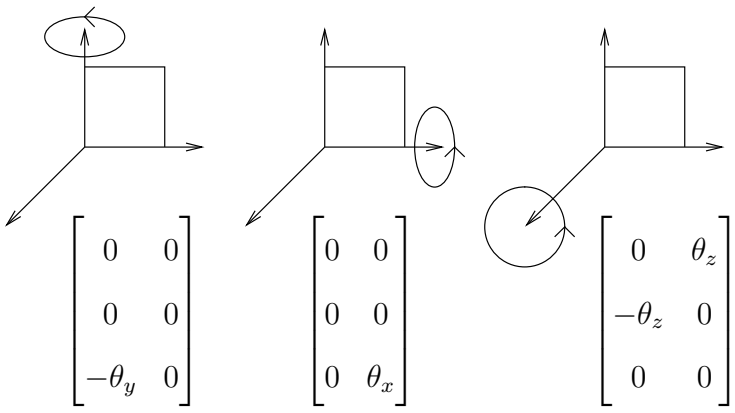

Figure 5.2. Two horizontal tangents (left) and one vertical tangent (right) at $[e_1, e_2] \in \text{Stief}(3, 2)$

The set of matrices in $\text{Stief}(n, m)$ spanning the same space as Z_0 is $\{Z \in \text{Stief}(n, m) : Z = Z_0 U,\ U \in O(m)\}$. The vertical directions are exactly the tangents to this set. So vertical motion "spins" vectors without changing the subspace, while horizontal motion changes the subspace spanned.

We define the differentiable structure of $\text{Grass}(n, m)$ in terms of the structure of $\text{Stief}(n, m)$: a path $\mathcal{Z}(s)$ in $\text{Grass}(n, m)$ is C^k if there is a C^k basis $Z : [0, 1] \to \text{Stief}(n, m)$ such that $\mathcal{Z}(s) = \text{span}(Z(s))$. This basis is not unique; however, given a basis $Z_0 \in \text{Stief}(n, m)$ for $\mathcal{Z}(0)$, there is a unique C^k basis starting from Z_0 which moves only horizontally. We describe the basis in the following lemma.

Lemma 1. *Let $\mathcal{Z} : [0, 1] \to \text{Grass}(n, m)$ be a C^k parameter-dependent space ($k > 0$). Then for any $Z_0 \in \text{Stief}(n, m)$ such that $\mathcal{Z}(0) = \text{span}(Z_0)$, there is a unique C^k basis $Z : [0, 1] \to \text{Stief}(n, m)$ for $\mathcal{Z}(s)$ such that $Z(0) = Z_0$ and*

$$Z(s)^T Z'(s) = 0. \tag{5.11}$$

This basis minimizes the Euclidean arclength

$$l(Z) = \int_0^1 \|Z'(s)\|_F \, ds \tag{5.12}$$

over all C^k orthonormal bases for $\mathcal{Z}(s)$.

Proof. Let $\hat{Z} : [0,1] \to \text{Stief}(n,m)$ be one C^k orthonormal basis for $\mathcal{Z}$. Any other C^k orthonormal basis for $\mathcal{Z}$ can be written $Z = \hat{Z}U$ for some C^k function $U : [0,1] \to O(m)$. By the Pythagorean theorem,

$$\|(\hat{Z}U)'\|_F^2 = \|(I - \hat{Z}\hat{Z}^T)(\hat{Z}U)'\|_F^2 + \|\hat{Z}\hat{Z}^T(\hat{Z}U)'\|_F^2 \tag{5.13}$$

where the first term corresponds to horizontal motion, and the second term to vertical motion. Since the Frobenius norm is invariant under unitary transformations, we can show the first term depends only on $\mathcal{Z}$, and not on the particular choice of basis:

$$\begin{aligned} \|(I - \hat{Z}\hat{Z}^T)(\hat{Z}U)'\|_F &= \|(I - \hat{Z}\hat{Z}^T)(\hat{Z}'U + \hat{Z}U')\|_F & (5.14)\\ &= \|(I - \hat{Z}\hat{Z}^T)\hat{Z}'U\|_F & (5.15)\\ &= \|(I - \hat{Z}\hat{Z}^T)\hat{Z}'\|_F. & (5.16) \end{aligned}$$

By again using unitary invariance of the norm, we rewrite the second term as

$$\|\hat{Z}\hat{Z}^T(\hat{Z}U)'\|_F = \|\hat{Z}^T(\hat{Z}U)'\|_F = \|(\hat{Z}U)^T(\hat{Z}U)'\|_F. \tag{5.17}$$

Therefore, the minimum attainable arclength should occur when

$$0 = (\hat{Z}U)^T(\hat{Z}U)' = U^T(\hat{Z}^T\hat{Z}'U + U') \tag{5.18}$$

or equivalently,

$$U' = -\hat{Z}\hat{Z}'U. \tag{5.19}$$

By the standard theory for linear ODEs, there is a unique U which satisfies (5.19) together with the initial condition $\hat{Z}(0)U(0) = Z_0$. Therefore, there is a unique orthonormal basis $Z = \hat{Z}U$ which satisfies (5.11) and $Z(0) = Z_0$. Furthermore, Z has minimal arclength.

□

For computation, we can approximate the equation $Z(s)^T Z'(s) = 0$ by the condition

$$Z(s_i)^T(\bar{Z}(s_{i+1}) - Z(s_i)) = 0. \tag{5.20}$$

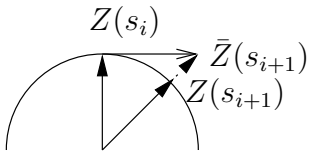

Figure 5.3. Discrete approximation to $Z(s)^T Z'(s) = 0$ for $m = 1$

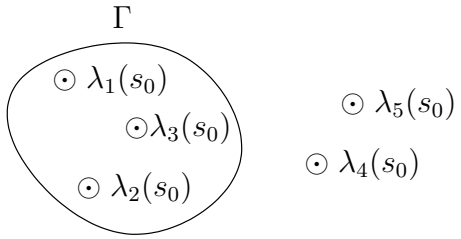

Figure 5.4. Contour Γ in $\mathbb{C}$ enclosing $\Lambda_1 \subset \Lambda$

As long as no vectors in $\mathcal{Z}(s_i)$ are normal to $\mathcal{Z}(s_{i+1})$, such a $\bar{Z}(s_{i+1}) \in \mathbb{R}^{n\times m}$ exists. Then we let the computed $Z(s_{i+1})$ be the element of $\mathrm{Stief}(n, m)$ nearest $\bar{Z}(s_{i+1})$ (Figure 5.3). The problem of finding the nearest element of $\mathrm{Stief}(n, m)$ to a given full-rank matrix in $\mathbb{R}^{n\times m}$ is called the *orthogonal Procrustes problem* [81, p. 582], and we will return to it later.

5.2.2 Complex-analytic characterization

In [106], Kato characterizes continuity of invariant subspaces in terms of the associated eigenprojections. If Γ is a union of disjoint positively-oriented simple closed contours in $\mathbb{C}$ with $\Lambda_1(s_0)$ inside Γ and $\Lambda_2(s_0)$ outside Γ (see Figure 5.4), then

$$P(s) := -\frac{1}{2\pi i}\int_\Gamma (A(s) - \xi I)^{-1}\, d\xi \tag{5.21}$$

is well-defined for any s near s_0. The matrix $P(s)$ is a projection with range $\mathcal{R}(s)$.

Suppose $X_0 \in \mathbb{R}^{n\times m}$ is a basis for $\mathcal{R}(s_0)$. Then we use $P(s)$ to locally produce a

continuous basis $X(s)$ for $\mathcal{R}(s)$:

$$X(s) := P(s)X_0. \tag{5.22}$$

Because $X(s_0) = X_0$ is full rank, and the full rank matrices form an open subset of $\mathbb{R}^{n\times m}$, by continuity $X(s)$ will have full rank for all s sufficiently near s_0.

5.2.3 Differential equation characterization

We can also prove the existence of a C^k invariant subspace by writing a differential equation for a Schur factorization. This is the approach used in [62], [60], and [63]; we summarize their result in the following theorem.

Theorem 2. *([62, 60, 63]) Suppose $\Lambda_1(s)$ and $\Lambda_2(s)$ are disjoint for all $s \in [0,1]$. Then there is an orthogonal matrix Q and block upper triangular matrix T, each with C^k dependence on s, so that*

$$\begin{aligned} A(s) &= Q(s)T(s)Q(s)^T & (5.23)\\ &= \begin{bmatrix} Q_1(s) & Q_2(s) \end{bmatrix} \begin{bmatrix} T_{11}(s) & T_{12}(s) \\ 0 & T_{22}(s) \end{bmatrix} \begin{bmatrix} Q_1(s) & Q_2(s) \end{bmatrix}^T. & (5.24) \end{aligned}$$

where $Q_1(s) \in \mathbb{R}^{n\times m}$ is a basis for the subspace $\mathcal{R}(s)$ corresponding to $\Lambda_1(s)$, and $Q_2(s) \in \mathbb{R}^{n\times(n-m)}$ is a basis for $\mathcal{R}(s)^\perp$.

Proof. The proof is written in detail in the cited references, so we only sketch the main ideas here. We differentiate the relation $A = QTQ^T$ to get

$$A' = Q'TQ^T + QTQ'^T + QT'Q^T. \tag{5.25}$$

Because Q is orthogonal, $H = Q^TQ'$ must be skew; by multiplying by Q^T and Q on the left and right, respectively, we have

$$Q^TA'Q = HT - TH + T'. \tag{5.26}$$

Since $T_{21} = T'_{21} = 0$, H_{21} satisfies

$$Q_2^T A' Q_1 = H_{21} T_{11} - T_{22} H_{21} \tag{5.27}$$

The spectra of T_{11} and T_{22} (Λ_1 and Λ_2 respectively) remain disjoint by hypothesis, so there is a unique solution H_{21} for equation (5.27). We specify that $H_{11}(s) = Q_1(s)^T Q_1'(s) = 0$ and $H_{22}(s) = Q_2(s)^T Q_2'(s) = 0$ to get a unique solution for equation (5.25) given an initial factorization $A(0) = Q(0)T(0)Q(0)^T$. Because we have constrained Q_1 and Q_2 to move only horizontally, Q_1 is a minimal arclength basis for $\mathcal{R}$ and Q_2 is a minimal arclength basis for $\mathcal{R}^\perp$. □

5.2.4 Algebraic characterization

A Riccati equation

Suppose $A \in C^k([0,1], \mathbb{R}^{n\times n})$ and at some $s_0 \in [0,1]$, $\Lambda_1(s_0)$ and $\Lambda_2(s_0)$ are disjoint. Then by the results in previous sections, there is a (non-unique) continuous block Schur decomposition for s near s_0, which at s_0 is

$$A(s_0) = \begin{bmatrix} Q_1(s_0) & Q_2(s_0) \end{bmatrix} \begin{bmatrix} T_{11}(s_0) & T_{12}(s_0) \\ 0 & T_{22}(s_0) \end{bmatrix} \begin{bmatrix} Q_1(s_0) & Q_2(s_0) \end{bmatrix}^T \tag{5.28}$$

where the the spectrum of $T_{ii}(s_0)$ is $\Lambda_i(s_0)$. Sufficiently near s_0, continuity demands that no nonzero vector in $\mathcal{R}(s)$ be orthogonal to $\mathcal{R}(s_0)$, so we may write

$$\mathcal{R}(s) = \operatorname{span}\left(Q(s_0) \begin{bmatrix} I \\ Y(s) \end{bmatrix} \right) \tag{5.29}$$

for some continuous Y with $Y(s_0) = 0$. The function $Y(s)$ must satisfy an algebraic Riccati equation, which we describe in the following lemma.

Lemma 3. *([60, 63]) Let $A \in C^k([0,1], \mathbb{R}^{n\times n})$ have a block Schur decomposition at*

s_0 as in (5.28), where the diagonal blocks of $T(s_0)$ have disjoint spectra. Define

$$\widehat{T}(s) = \begin{bmatrix} \widehat{T}_{11}(s) & \widehat{T}_{12}(s) \\ E_{21}(s) & \widehat{T}_{22}(s) \end{bmatrix} := Q(s_0)^T A(s) Q(s_0). \tag{5.30}$$

Then for s near s_0, there is a unique, continuous, minimum-norm solution $Y(s) \in \mathbb{R}^{(n-m)\times m}$ to the Riccati equation

$$F(Y) := \widehat{T}_{22}(s)Y - Y\widehat{T}_{11}(s) + E_{21}(s) - Y\widehat{T}_{12}(s)Y = 0 \tag{5.31}$$

and there is a continuous block Schur decomposition

$$A(s) = Q(s)T(s)Q(s)^T \tag{5.32}$$

where

$$Q(s) = \bar{Q}(s)\left(\bar{Q}(s)^T\bar{Q}(s)\right)^{-1/2} \tag{5.33}$$

$$\bar{Q}(s) = Q(s_0)\begin{bmatrix} I & -Y(s)^T \\ Y(s) & I \end{bmatrix}. \tag{5.34}$$

This theorem is stated in [60] and [63], and extends results proved by Demmel [58], Stewart [156], and Stewart and Sun [158, section V.2]. For completeness, we repeat the proof here.

Proof. We want the matrix $\bar{Q}_1(s)$, which is exactly the matrix used in (5.29), to be a basis for $\mathcal{R}(s)$. To span an invariant subspace, $\bar{Q}_1(s)$ must satisfy the equation

$$A(s)\bar{Q}_1(s) = \bar{Q}_1(s)\bar{T}_{11}(s) \tag{5.35}$$

for some matrix $\bar{T}_{11}(s)$. As we saw in Section 5.2.1, (5.35) has a continuous set of solutions. To specify a unique solution, we add a normalizing equation:

$$Q_1(s_0)^T\bar{Q}_1(s) = I, \tag{5.36}$$

which implies

$$\bar{Q}_1(s) = Q(s_0) \begin{bmatrix} I \\ Y(s) \end{bmatrix}. \tag{5.37}$$

In order to have $\bar{Q}_1(s_0) = Q_1(s_0)$, we require $Y(s_0) = 0$.

If we multiply (5.35) on the left by $Q(s_0)^T$ and substitute (5.37) for $\bar{Q}_1(s)$ we have

$$Q(s_0)^T A(s) Q(s_0) \begin{bmatrix} I \\ Y(s) \end{bmatrix} = \begin{bmatrix} I \\ Y(s) \end{bmatrix} \bar{T}_{11}(s). \tag{5.38}$$

Now we rewrite $Q(s_0)^T A(s) Q(s_0)$ using (5.30):

$$\begin{bmatrix} \widehat{T}_{11}(s) & \widehat{T}_{12}(s) \\ E_{21}(s) & \widehat{T}_{22}(s) \end{bmatrix} \begin{bmatrix} I \\ Y(s) \end{bmatrix} = \begin{bmatrix} I \\ Y(s) \end{bmatrix} \bar{T}_{11}(s). \tag{5.39}$$

The first row of (5.39) gives us an expression for $\bar{T}_{11}(s)$:

$$\bar{T}_{11}(s) = \widehat{T}_{11}(s) + \widehat{T}_{12}(s) Y(s). \tag{5.40}$$

We substitute into the second row to get

$$E_{21}(s) + \widehat{T}_{22}(s) Y(s) = Y(s) \left(\widehat{T}_{11}(s) + \widehat{T}_{12}(s) Y(s) \right) \tag{5.41}$$

and rearrange terms to get (5.31).

Note that the computed $Q(s)$ is an orthogonal matrix, and is designed so that the leading columns span $\mathcal{R}(s)$. □

A constructive existence proof

Lemma 3 says that near s_0 we can write $\mathcal{R}(s)$ in terms of a continuous solution to an algebraic Riccati equation, but it says nothing about the size of the neighborhood or the magnitude of the Riccati solution. To get more detailed information about $Y(s)$, we extend a theorem due to Stewart [156], [158, section V.2].

Theorem 4. *Define* $\Omega := C([a,b], \mathbb{R}^{(n-m)\times m})$. *For* $Y \in \Omega$, *we will suppress the argument* s *to write* $\|Y\|$ *for the function* $s \mapsto \|Y(s)\|$. *The norm* $\|\cdot\|$ *may be any consistent norm. We use* $|||Y||| = \max_{s\in[a,b]} \|Y(s)\|$ *to denote the norm on* Ω.

Let $Y_0 \in \Omega$ *be given. Define a Sylvester operator* $\mathsf{S} : \Omega \to \Omega$ *and a bilinear function* $\phi : \Omega \times \Omega \to \Omega$ *by*

$$\begin{aligned} \mathsf{S}Z &:= Z(\hat{T}_{11} + \hat{T}_{12}Y_0) - (\hat{T}_{22} - Y_0\hat{T}_{12})Z \qquad (5.42)\\ \phi(X,Y) &:= \mathsf{S}^{-1}(X\hat{T}_{12}Y). \qquad (5.43) \end{aligned}$$

Suppose S *is invertible on* $[a,b]$. *Then we can define continuous functions* $\alpha, \beta : [a,b] \to \mathbb{R}$ *by*

$$\begin{aligned} \alpha &:= \|\mathsf{S}^{-1}(F(Y_0))\| \qquad (5.44)\\ \beta &:= \max_{\|X\|=\|Y\|=1} \|\phi(X,Y)\|. \qquad (5.45) \end{aligned}$$

Suppose also that $4\alpha\beta < 1$ *on* $[a,b]$, *and define*

$$\xi_* := \frac{2\alpha}{1+\sqrt{1-4\alpha\beta}}. \qquad (5.46)$$

Then there is a unique continuous solution Y_* *to the Riccati equation (5.31) such that* $\|Y_* - Y_0\| \le \xi_*$.

Proof. Let $Z := Y - Y_0$. Then we rewrite (5.31) as

$$0 = F(Y_0 + Z) = F(Y_0) - \mathsf{S}(Z + \phi(Z,Z)), \qquad (5.47)$$

which we can rearrange to get

$$Z = \mathsf{S}^{-1}\left(F(Y_0)\right) - \phi(Z,Z). \qquad (5.48)$$

So the map $\psi : \Omega \to \Omega$ given by

$$\psi(Z) := \mathsf{S}^{-1}(F(Y_0)) - \phi(Z,Z) \qquad (5.49)$$

has fixed points where (5.31) has solutions. Now define $\Omega_0 = \{Z \in \Omega : \|Z\| \leq \xi_*\}$. We will show that $\psi(\Omega_0) \subseteq \Omega_0$ and ψ is contractive on Ω_0, so by the contraction mapping theorem the iteration $Z_{i+1} = \psi(Z_i)$ will converge to a unique fixed point $Z_* \in \Omega_0$ starting from any $Z_1 \in \Omega_0$.

1. $\psi(\Omega_0) \subseteq \Omega_0$:

   By the definition of α and β,

   $$\|\psi(Z)\| \leq \alpha + \beta\|Z\|^2$$

   Define $\tau(\xi) = \alpha + \beta\xi^2$. The quadratic equation $\xi = \tau(\xi)$ has two real solutions when $4\alpha\beta < 1$; the smaller solution is

   $$\xi_* = \frac{1 - \sqrt{1 - 4\alpha\beta}}{2\beta} = \frac{2\alpha}{1 + \sqrt{1 - 4\alpha\beta}}$$

   Because $\beta \geq 0$, τ is monotonically nondecreasing for positive arguments. So for $0 \leq \|Z\| \leq \xi_*$,

   $$0 \leq \|\psi(Z)\| \leq \tau(\|Z\|) \leq \tau(\xi_*) = \xi_*$$

   So $\psi(\Omega_0) \subset \Omega_0$. Therefore all the iterates Z_i remain in Ω_0.

2. ψ is contractive on Ω_0:

   For any $X, Y \in \Omega_0$,

   $$\begin{aligned}
   \|\psi(X) - \psi(Y)\| &= \|\phi(X, X) - \phi(Y, Y)\| \\
   &= \|\phi(X, X - Y) + \phi(X - Y, Y)\| \\
   &\leq \beta\left(\|X\|\|X - Y\| + \|X - Y\|\|Y\|\right) \\
   &\leq 2\beta\xi_*\|X - Y\| \\
   &= \frac{4\alpha\beta}{1 + \sqrt{1 - 4\alpha\beta}}\|X - Y\| \\
   &< 4\alpha\beta\|X - Y\|
   \end{aligned}$$

Let $\gamma := \max_{s\in[a,b]} 4\alpha\beta$; by hypothesis, $\gamma < 1$. Then we have

$$|||\psi(X) - \psi(Y)||| < \gamma\,|||X - Y|||\,.$$

Therefore, ψ has a unique fixed point Z_* in Ω_0; and there is a unique continuous solution $Y_* = Y_0 + Z_*$ to the Riccati equation (5.31) such that $\|Y_* - Y_0\| \leq \xi_*$.

□

The *separation* of matrices B and C is the smallest singular value of the Sylvester operator $\mathsf{A}(X) = BX - XC$:

$$\operatorname{sep}(B, C) := \sigma_{\min}(\mathsf{A}) = \min_{\|X\|_F=1} \|BX - XC\|_F = \frac{1}{\|\mathsf{A}^{-1}\|_2}. \tag{5.50}$$

$\operatorname{sep}(B, C)$ is zero when B and C have a common eigenvalue, and it is small if a small perturbation makes them share an eigenvalue. If B and C are normal, $\operatorname{sep}(B, C)$ is the distance between their spectra, but in general $\operatorname{sep}(B, C)$ may be much smaller, since a small change to a non-normal matrix can cause a relatively large change to the spectrum [165, 169]. By manipulating norm inequalities and using the notion of matrix separation, we can bound the quantities α and β defined in Theorem 4.

Lemma 5. *Let α and β be defined as in Theorem 4. Then the following inequalities hold pointwise one $s \in [a, b]$:*

$$\|\mathsf{S}\|_2^{-1}\|F(Y_0)\|_F \;\leq\; \alpha \;\leq\; \|\mathsf{S}^{-1}\|_2\|F(Y_0)\|_F \tag{5.51}$$

and

$$\frac{1}{\sqrt{(n-m)m}}\|\mathsf{S}^{-1}\|_2\|\widehat{T}_{12}\|_2 \;\leq\; \beta \;\leq\; \|\mathsf{S}^{-1}\|_2\|\widehat{T}_{12}\|_2, \tag{5.52}$$

where

$$\|\mathsf{S}^{-1}\|_2 = \frac{1}{\operatorname{sep}(\widehat{T}_{11} + \widehat{T}_{12}Y_0, \widehat{T}_{22} - Y_0\widehat{T}_{12})}. \tag{5.53}$$

Proof. Equation (5.53) is simply the definition of sep, while (5.51) follows from the basic properties of an operator two-norm. To see the upper bound in (5.52), observe that

$$\|\phi(X,Y)\|_F = \|X\widehat{T}_{12}Y\|_F \leq \|\widehat{T}_{12}\|_2\|X\|_F\|Y\|_F.$$

To see the lower bound in (5.52), let $X = e_i u^T$ and $Y = ve_j^T$, where u and v are left and right singular vectors for $\sigma_{\max}(\widehat{T}_{12})$ and i and j are chosen to maximize $\|\mathsf{S}^{-1}(e_i e_j^T)\|$. Then $\|X\|_F = \|Y\|_F = 1$, and

$$\begin{aligned}
\beta &\geq \|\mathsf{S}^{-1}((e_i u^T)\widehat{T}_{12}(ve_j)\|_F && (5.54)\\
&= \|\widehat{T}_{12}\|_2\|\mathsf{S}^{-1}(e_i e_j^T)\|_F && (5.55)\\
&\geq \frac{1}{\sqrt{(n-m)m}}\|\mathsf{S}^{-1}\|_2\|\widehat{T}_{12}\|_2 && (5.56)
\end{aligned}$$

We can see the last inequality by viewing $\mathsf{S}^{-1}(e_i e_j)^T$ as the column of greatest norm from S^{-1} when S^{-1} is viewed in Kronecker product form as an $(n-m)m$-by-$(n-m)m$ matrix. □

The bounds (5.51) and (5.52) together with Theorem 4 yield the following theorem.

Theorem 6. *([60, 63]) Let $Y_0 : [a,b] \to \mathbb{R}^{(n-m)\times m}$ be continuous, and define*

$$\kappa(\widehat{T}) := \frac{\|\widehat{T}_{12}\|_2\ \|F(Y_0)\|_F}{\mathrm{sep}^2(\widehat{T}_{11} + \widehat{T}_{12}Y_0, \widehat{T}_{22} - Y_0\widehat{T}_{12})} \tag{5.57}$$

In any neighborhood containing s_0 in which $\kappa(\widehat{T}) < 1/4$, the Riccati equation (5.31) has a unique continuous solution $Y_*(s)$ such that

$$\|Y_*\|_F < \frac{2\|F(Y_0)\|_F}{\mathrm{sep}(\widehat{T}_{11} + \widehat{T}_{12}Y_0, \widehat{T}_{22} - Y_0\widehat{T}_{12})}. \tag{5.58}$$

In any neighborhood containing s_0 where $\kappa(\widehat{T}) < 1/12$, Newton's method at a fixed s on (5.31) will converge quadratically to Y_* starting from Y_0.

Proof. To prove the existence statement, substitute (5.51) and (5.52) into Theorem 4.

In [58], Demmel proved the quadratic convergence of Newton's iteration for $\kappa(\widehat{T}) < 1/12$. To extend to the case of a parameter-dependent matrix, we simply apply Demmel's theorem pointwise.

□

5.2.5 Connecting subspaces

Suppose we are given bases for invariant subspaces of $A(s)$ at $s = 0$ and $s = h$. How can we check that the two end points are connected by a continuously defined invariant subspace basis on $[0, h]$? This question has practical significance for our continuation algorithm, since we would like to avoid mistaken branch-jumping behavior when two subspaces come close to each other, and we would like to detect when a continued invariant subspace ceases to be continuously defined.

Theorem 6 partially answers the question of how to check for a continuous connecting invariant subspace. But to apply the theorem, we need to bound $\kappa(\widehat{T})$ on the interval $[0, h]$. In the remainder of this section, we describe how to construct bounds which incorporate information from both $s = 0$ and $s = h$ using interpolation. Our ultimate goal is Theorem 12, but first we need some technical lemmas.

We first turn to the problem of bounding $\|B^{-1}\|_2$, where $B \in C^1([0, h], \mathbb{R}^{p\times p})$ is some parameterized operator on a Euclidean space. Since S is also a linear operator on a Euclidean space ($\mathbb{R}^{(n-m)\times m}$ with the Frobenius inner product), all our results apply directly to S as well. We begin by reviewing a simple result about matrix interpolation.

Lemma 7. *Suppose $B \in C^1([0, h], \mathbb{R}^{p\times p})$ and B' is Lipschitz with constant M. Then*

$$B(s) = B(0) + B[0,h]s + B[0,h,s]s(s-h) \tag{5.59}$$

where $B[0,h]$ *and* $B[0,h,s]$ *are first and second Newton divided differences and*

$$\begin{aligned} \|B[0,h]\|_2 &\leq \max_{\xi\in[0,h]} \|B'(\xi)\|_2 \\ \|B[0,h,s]\|_2 &\leq M. \end{aligned}$$

Proof. For any $u, v \in \mathbb{R}^p$ and any distinct $a, b \in [0,h]$, $a < b$, the mean value theorem applied to the scalar function $u^T B(s) v$ implies

$$u^T B[a,b] v = u^T B(\xi) v \tag{5.60}$$

for some $\xi \in [a,b]$. Therefore, $\|B[a,b]\|_2 \leq \max_{\xi\in[a,b]} \|B(\xi)\|_2$.

Now we compute

$$\begin{aligned} u^T B[0,h,s] v &= (u^T B[0,s]v - u^T B[h,s]v)/h && (5.61)\\ &= (u^T B'(\xi_1)v - u^T B'(\xi_2)v)/h && (5.62)\\ &\leq \frac{\|B'(\xi_1) - B'(\xi_2)\|_2}{h} \|u\|_2\|v\|_2 && (5.63)\\ &\leq M\|u\|_2\|v\|_2. && (5.64) \end{aligned}$$

So $\|B[0,h,s]\|_2 \leq M$.

□

We can now show a very simple bound on the minimal singular value of B.

Lemma 8. *Suppose* $B \in C^1([0,h], \mathbb{R}^{p\times p})$ *and* B' *is Lipschitz with constant* M*. Then*

$$\sigma_{\min}(B(s)) \geq \sigma_{\min}(B(0)) - \|B[0,h]\|_2 s - Ms(h-s) \tag{5.65}$$

Proof. By the previous lemma,

$$\|B(s) - B(0)\|_2 = \|B[0,h]s + B[0,h,s]s(s-h)\| \leq \|B[0,h]\|_2 s + Ms(s-h).$$

To complete the proof, recall (e.g. from [81]) that

$$|\sigma_{\min}(B(s)) - \sigma_{\min}(B(0))| \leq \|B(s) - B(0)\|_2.$$

□

Lemma 8 uses only the norm of $B(s) - B(0)$; we can refine the bound by using the direction as well as the magnitude.

Lemma 9. *Suppose $B \in C^1([0,h], \mathbb{R}^{p\times p})$ and B' is Lipschitz with constant M. Then*

$$\sigma_{\min}(B(s)) \geq \sigma_{\min}(B(0))(1 - \|B(0)^{-1}B[0,h]\|_2 s) - Ms(h-s) \tag{5.66}$$

Proof. Let $E(s) = B[0,h]s$. If $\|B(0)^{-1}E(s)\|_2 \geq 1$, then the lemma is trivial. Otherwise, $I + B(0)^{-1}E(s)$ is invertible, and

$$\begin{aligned}(B(0) + E(s))^{-1} &= \left(I + B(0)^{-1}E(s)\right)^{-1} B(0)^{-1} &(5.67)\\ &= \sum_{k=0}^{\infty} \left(-B(0)^{-1}E(s)\right)^k B(0)^{-1} &(5.68)\end{aligned}$$

so

$$\left\|(B(0) + E(s))^{-1}\right\|_2 \leq \frac{\|B(0)^{-1}\|_2}{1 - \|B(0)^{-1}E(s)\|_2}. \tag{5.69}$$

Taking inverses on both sides, we have

$$\sigma_{\min}(B(0) + E(s)) \geq \sigma_{\min}(B(0))(1 - \|B(0)^{-1}E(s)\|). \tag{5.70}$$

Therefore

$$\begin{aligned}\sigma_{\min}(B(s)) &= \sigma_{\min}(B(0) + B[0,s]s + B[0,h,s]s(s-h)) &(5.71)\\ &\geq \sigma_{\min}(B(0) + B[0,s]s) - Ms(h-s) &(5.72)\\ &\geq \sigma_{\min}(B(0))(1 - \|B(0)^{-1}B[0,h]\|_2 s) - Ms(h-s) &(5.73)\end{aligned}$$

□

We now turn to the problem of bounding $\|F(Y_0)\|_F$ in 5.31 for a specific choice of Y_0. Suppose $\begin{bmatrix} I \\ hZ \end{bmatrix}$ is a basis for a given invariant subspace of $\widehat{T}(h)$ (see 5.30); then we linearly interpolate $Y_0(s) = sZ$, so that the residual $F(Y_0)$ is zero at both $s = 0$ and $s = h$.

Lemma 10. *Suppose $\widehat{T} \in C^1$ and $\widehat{T}'$ has Lipschitz constant M. Also suppose $\begin{bmatrix} I \\ hZ \end{bmatrix}$ spans an invariant subspace of $\widehat{T}(h)$, and define*

$$G(s) := \widehat{T}_{22}[0,h]Z - Z\widehat{T}_{11}[0,h] - Z\left(\widehat{T}_{12}(0) + (s+h)\widehat{T}_{12}[0,h]\right)Z. \tag{5.74}$$

Then for $Y_0(s) = sZ$, and for any $s \in [0,h]$,

$$\|F(Y_0)\|_F \leq \frac{h^2}{2}\left\{\max(\|G(0)\|_F, \|G(h)\|_F) + \sqrt{m}M(1 + h\|Z\|_2)^2\right\} \tag{5.75}$$

Proof. We write $F(Y_0(s))$ as the product

$$F(Y_0(s)) = \begin{bmatrix} -Y_0(s) & I \end{bmatrix} \widehat{T}(s) \begin{bmatrix} I \\ Y_0(s) \end{bmatrix}. = \begin{bmatrix} -sZ & I \end{bmatrix} \widehat{T}(s) \begin{bmatrix} I \\ sZ \end{bmatrix}. \tag{5.76}$$

Using the Newton form of the interpolant,

$$\widehat{T}(s) = \widehat{T}(0) + \widehat{T}[0,h]s + \widehat{T}[0,h,s]s(s-h); \tag{5.77}$$

we can therefore write $F(Y_0(s))$ as

$$\begin{aligned} F(Y_0(s)) &= F_1(Y_0(s)) + F_2(Y_0(s)) && (5.78)\\ F_1(Y_0(s)) &= \begin{bmatrix} -sZ & I \end{bmatrix}\left(\widehat{T}(0) + \widehat{T}[0,h]s\right)\begin{bmatrix} I \\ sZ \end{bmatrix} && (5.79)\\ F_2(Y_0(s)) &= \begin{bmatrix} -sZ & I \end{bmatrix}\left(\widehat{T}[0,h,s]s(s-h)\right)\begin{bmatrix} I \\ sZ \end{bmatrix}. && (5.80) \end{aligned}$$

We now bound the norms of $F_1(Y_0(s))$ and $F_2(Y_0(s))$ independently.

To bound $F_1(Y_0(s))$, we expand and collect terms at each order in s:

$$\begin{aligned} F_1(Y_0(s)) &= E_{21}(0) \qquad (5.81)\\ &\quad +s\left(\widehat{T}_{22}(0)Z - Z\widehat{T}_{11}(0) + E_{21}[0,h]\right)\\ &\quad +s^2\left(\widehat{T}_{22}[0,h]Z - Z\widehat{T}_{11}[0,h] - Z\widehat{T}_{12}(0)Z\right)\\ &\quad +s^3\left(-Z\widehat{T}_{12}[0,h]Z\right) \qquad (5.82)\end{aligned}$$

Since $F(Y_0(s))|_{s=0} = 0$, we know $E_{21}(0) = 0$. Similarly, since $F(Y_0(s))|_{s=h} = 0$, we know

$$\begin{aligned} &\widehat{T}_{22}(0)Z - Z\widehat{T}_{11}(0) + E_{21}[0,h]\\ &= -h\left(\widehat{T}_{22}[0,h]Z - Z\widehat{T}_{11}[0,h] - Z\widehat{T}_{12}(0)Z\right)\\ &\quad -h^2\left(-Z\widehat{T}_{12}[0,h]Z\right). \qquad (5.83)\end{aligned}$$

Substituting (5.83) into (5.82), we have

$$\begin{aligned} F_1(Y_0(s)) &= (s^2 - sh)\left(\widehat{T}_{22}[0,h]Z - Z\widehat{T}_{11}[0,h] - Z\widehat{T}_{12}(0)Z\right) +\\ &\quad (s^3 - sh^2)\left(-Z\widehat{T}_{12}[0,h]Z\right). \qquad (5.84)\end{aligned}$$

Factoring out $s(s-h)$ from both terms, we have

$$F_1(Y_0(s)) = s(s-h)G(s). \qquad (5.85)$$

Note that $G(s)$ is linear, so by convexity of norms,

$$\|G(s)\|_F \le \max\left(\|G(0)\|_F, \|G(h)\|_F\right) \text{ for } s \in [0,h]. \qquad (5.86)$$

Therefore

$$\|F_1(Y_0(s))\|_F \le \frac{h^2}{2}\max\left(\|G(0)\|_F, \|G(h)\|_F\right) \text{ for } s \in [0,h]. \qquad (5.87)$$

We use a cruder bound for $F_2(Y_0(s))$. Since $F_2(Y_0(s)) \in \mathbb{R}^{(n-m)\times m}$, $\|F_2(Y_0(s))\|_F \le \sqrt{m}\|F_2(Y_0(s))\|_2$. Both $\begin{bmatrix} -sZ & I \end{bmatrix}$ and $\begin{bmatrix} I \\ hZ \end{bmatrix}$ are bounded in

2-norm by $1 + h\|Z\|_2$; and by 7, $\|\widehat{T}[0,h,s]\| \leq M$. Therefore

$$\begin{aligned}\|F_2(Y_0(s))\|_2 &\leq \left\|\begin{bmatrix}-sZ & I\end{bmatrix}\right\|_2 \left\|\widehat{T}[0,h,s]\right\|_2 \left\|\begin{bmatrix}I\\ sZ\end{bmatrix}\right\|_2 s(s-h) &(5.88)\\ &\leq \frac{h^2}{2}M(1+h\|Z\|_2)^2. &(5.89)\end{aligned}$$

Substituting the above bounds into $\|F(Y_0(s))\|_F \leq \|F_1(Y_0(s))\|_F + \|F_2(Y_0(s))\|_F$ concludes the proof.

□

Now we bound $\|\widehat{T}_{12}(s)\|_2$ on $[0,h]$.

Lemma 11. *Suppose $\widehat{T} \in C^1$ and $\widehat{T}'$ has Lipschitz constant M. Then for $s \in [0,h]$,*

$$\|\widehat{T}_{12}(s)\|_2 \leq \max\left(\|\widehat{T}_{12}(0)\|_2, \|\widehat{T}_{12}(h)\|_2\right) + \frac{1}{2}Ms(h-s) \tag{5.90}$$

Proof. By Lemma 7,

$$\begin{aligned}\|T_{12}(s)\|_2 &= \|T_{12}(0) + T_{12}[0,h]s + T_{12}[0,h,s]s(s-h)\|_2 &(5.91)\\ &\leq \|T_{12}(0) + T_{12}[0,h]s\|_2 + Ms(h-s), &(5.92)\end{aligned}$$

and because norms are convex functions,

$$\|T_{12}(0) + T_{12}[0,h]s\|_2 \leq \max\left(\|T_{12}(0)\|_2, \|T_{12}(h)\|_2\right). \tag{5.93}$$

□

Putting together the preceding bounds, we have the following theorem.

Theorem 12. *Suppose $\widehat{T}(s)$ is C^2 and $\widehat{T}'$ is Lipschitz with constant M. Suppose $\begin{bmatrix}I\\0\end{bmatrix}$ and $\begin{bmatrix}I\\hZ\end{bmatrix}$ span invariant subspaces at 0 and h respectively. Let S be defined as in (5.42). Then if*

$$\sigma_{\min}(\mathsf{S}(0))(1 - h\|\mathsf{S}(0)^{-1}\mathsf{S}[0,h]\|_2) - \frac{1}{2}Mh^2 > 0 \tag{5.94}$$

the operator S *is invertible for all* $s \in [0,h]$*. Further, the constants* α *and* β *defined in (5.44) and (5.45) are bounded for all* $s \in [0,h]$ *by*

$$\begin{aligned}
\alpha &\leq \frac{h^2}{2}\frac{\max\left(\|G(0)\|_F, \|G(h)\|_F\right) + \sqrt{m}M(1+h\|Z\|_2)^2}{\sigma_{\min}(\mathsf{S}(0))(1-h\|\mathsf{S}(0)^{-1}\mathsf{S}[0,h]\|_2) - \frac{1}{2}Mh^2} && (5.95)\\
&= \frac{h^2}{2}\frac{\max\left(\|G(0)\|_F, \|G(h)\|_F\right) + \sqrt{m}M}{\sigma_{\min}(\mathsf{S}(0))} + O(h^3) && (5.96)\\
\beta &\leq \frac{\max\left(\|\widehat{T}_{12}(0)\|_2, \|\widehat{T}_{12}(h)\|_2\right) + \frac{1}{2}Mh^2}{\sigma_{\min}(\mathsf{S}(0))(1-h\|\mathsf{S}(0)^{-1}\mathsf{S}[0,h]\|_2) - \frac{1}{2}Mh^2} && (5.97)\\
&= \frac{\max\left(\|\widehat{T}_{12}(0)\|_2, \|\widehat{T}_{12}(h)\|_2\right)}{\sigma_{\min}(\mathsf{S}(0))} + O(h) && (5.98)
\end{aligned}$$

where

$$G(s) = \widehat{T}_{22}[0,h]Z - Z\widehat{T}_{11}[0,h] - Z\left(\widehat{T}_{12}(0) + (s+h)\widehat{T}_{12}[0,h]\right)Z.$$

Therefore, by Theorem 4, if the resulting upper bound on $4\alpha\beta$ *is bounded below one, there is a continuous connecting invariant subspace between* $\begin{bmatrix} I \\ 0 \end{bmatrix}$ *at* $s = 0$ *and* $\begin{bmatrix} I \\ hZ \end{bmatrix}$ *at* $s = h$*.*

Dropping higher-order terms, we have

$$\begin{aligned}
\alpha &\leq \frac{h^2}{2}\frac{\max\left(\|G(0)\|_F, \|G(h)\|_F\right) + \sqrt{m}M}{\sigma_{\min}(\mathsf{S}(0))} + O(h^3) && (5.99)\\
\beta &\leq \frac{\max\left(\|\widehat{T}_{12}(0)\|_2, \|\widehat{T}_{12}(h)\|_2\right)}{\sigma_{\min}(\mathsf{S}(0))} + O(h) && (5.100)
\end{aligned}$$

Besides $\mathrm{sep}(\widehat{T}_{11}(0), \widehat{T}_{22}(0)) = \sigma_{\min}(\mathsf{S}(0))$ and $\|\mathsf{S}(0)^{-1}\mathsf{S}[0,h]\|_2$, the quantities in the bounds of the above theorem are cheap and simple to compute.

5.3 The CIS algorithm: direct methods

We now describe the CIS algorithm in the case when we can use direct solvers. Much of this work is described in [60], [63], [73], and [74]. Here, we emphasize parts

of the computation that we perform differently, or which are particularly relevant to the sparse case.

At the highest level, our algorithm is as follows:

1. Choose an initial invariant subspace.
2. Compute a continuation step.
3. Normalize the solution.
4. Adapt the space and step size to improve convergence and resolve features of interest.

We can continue either $Q_1(s)$ and $T_{11}(s)$ or the full $Q(s)$ and $T(s)$ matrices. Currently, our dense code computes the full Schur factors at each step. When we continue only the first part of the decomposition, as we do in the sparse case, we also compute a few extra eigenvalues from $\Lambda_2(s)$. We use these eigenvalues to decide whether the algorithm should be reinitialized with a different partitioning of the spectrum.

5.3.1 Initialization

To initialize the algorithm at s_0, we compute a Schur decomposition of $A(s_0)$ and use standard LAPACK routines [6] to sort the decomposition so selected eigenvalues appear in $T_{11}(s_0)$. For bifurcation problems, we assume that only a small part of the spectrum is unstable; therefore, we include all the unstable eigenvalues as well as a few stable eigenvalues nearest the imaginary axis in our m-dimensional subspace (see Figure 5.5).

We require that $\Lambda_1(s_0)$ contains any unstable eigenvalues and some specified number of stable eigenvalues; but we may include additional eigenvalues in order to sim-

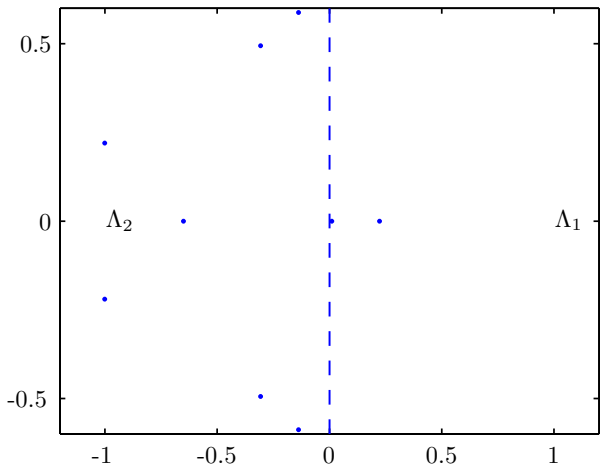

Figure 5.5. Selected eigenvalues during initialization

plify the subsequent continuation process. For example, we include an extra eigenvalue in order to avoid splitting a complex conjugate pair of eigenvalues between $\Lambda_1(s_0)$ and $\Lambda_2(s_0)$. More generally, we would like to choose $\Lambda_1(s_0)$ so that the gap between the real parts of the leftmost eigenvalue in $\Lambda_1(s_0)$ and the rightmost eigenvalue in $\Lambda_2(s_0)$ are greater than some threshold. In this way, we hope to keep track of all eigenvalues that might cross the imaginary axis.

In the dense case, the same LAPACK routine used to sort the Schur form also estimates the sensitivity of the selected subspace, and so we may choose a larger subspace if the smallest feasible subspace is very sensitive. Though the cost of the computations at a single point increases as we increase the size of our subspace, continuing a less sensitive subspace will allow us to take larger steps.

We summarize the initialization procedure in Algorithm 1.

5.3.2 Choosing a subspace

We have considered three strategies for computing $\mathcal{R}(s_1)$ starting from $\mathcal{R}(s_0)$:

Algorithm 1 Choose an initial subspace

Input: $A(s_0)$,
$n_{\min}$, $n_{\max}$, {bounds on subspace size}
$n_{\text{stableref}}$, {number of stable reference eigenvalues}
ϵ_{gap}, {minimum gap between $\Lambda_1(s_0)$ and $\Lambda_2(s_0)$}

Output: $Q_1(s_0)$ and $T_{11}(s_0)$

Compute a Schur decomposition $A(s_0) = QTQ^T$

$t :=$ real parts of converged eigenvalues sorted in descending order

Find smallest m so that $\begin{cases} n_{\min} \le m \le n_{\max} \\ m \ge (\#\text{ unstable eigenvalues}) + n_{\text{stableref}} \\ t(m) - t(m+1) > \epsilon_{\text{gap}} \end{cases}$

if no such m exists **then**

error "Spectrum too tightly clustered"

else

Sort subspace for rightmost m eigenvalues to the front of Q, T

Return $Q_1 = Q(:, 1:m)$, $T_{11} = T(1:m, 1:m)$

end if

- As in the construction of Theorem 6, apply a predictor and then use a Newton corrector.
- Choose a subspace which minimizes the distance between eigenvalues in the computed $\Lambda(s_1)$ and eigenvalues in $\Lambda(s_0)$.
- Choose a subspace by finding the m eigenvectors of $A(s_1)$ which most nearly lie in $\mathcal{R}(s_0)$, or which most nearly lie in a predicted subspace.

We currently use an approximate Euler predictor and a Newton corrector. We use the convergence of the corrector to govern our step size: if it converges slowly or fails to converge, we reduce the step size, or reinitialize the continuation process with a larger or smaller subspace. If the corrector converges quickly, we increase the step size.

Subspace predictors

We build an Euler predictor for $\mathcal{R}(s_1)$ by differentiating the Schur factorization as in (5.25) and substituting finite difference approximations for Q' and T'. Alternatively, we could differentiate the Riccati equation (5.31) and substitute a finite difference approximation for Y'. Either way, this gives us the equation

$$T_{22}(s_0)Y_0(s_1) - Y_0(s_1)T_{11}(s_0) = -(s_1 - s_0)E'_{21}(s_1) \tag{5.101}$$

If derivatives of A are unavailable, we can substitute a finite difference approximation for $E'_{21}(s)$ to get the approximate Euler predictor equation

$$T_{22}(s_0)Y_0(s_1) - Y_0(s_1)T_{11}(s_0) = -E_{21}(s_1). \tag{5.102}$$

We can also build a secant predictor; but to do so, we must consider how consecutive steps are normalized. In a single predictor-corrector step, we normalize the basis

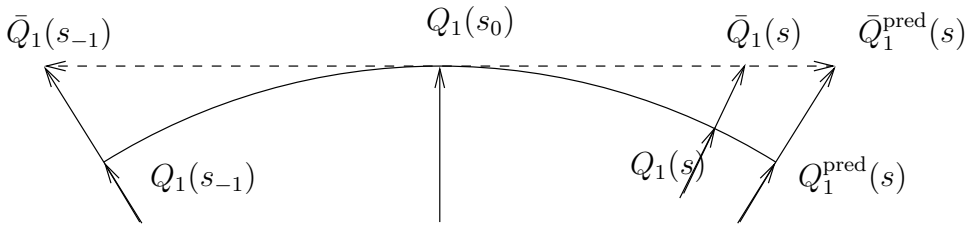

Figure 5.6. Choosing a consistent normalization for secant prediction

for a space $\mathcal{X}$ by requiring that $Q(s_0)^T X = I$; however, this normalization changes with each step. If $\mathcal{R}(s_{-1})$ is the invariant subspace from a previous continuation step, we must choose a basis $\bar{Q}_1(s_{-1})$ for $\mathcal{R}(s_{-1})$ which is consistent with the current normalization (see Figure 5.6). Because $\bar{Q}_1(s_{-1})$ spans the same space as $Q_1(s_{-1})$, there must be some invertible $B(s_{-1}) \in \mathbb{R}^{m\times m}$ such that

$$\bar{Q}_1(s_{-1}) = Q_1(s_{-1})B(s_{-1}), \tag{5.103}$$

and the normalizing condition is

$$I = Q_1(s_0)^T \bar{Q}_1(s_{-1}) = Q_1(s_0)^T Q_1(s_{-1})B(s_{-1}). \tag{5.104}$$

Therefore

$$B(s_{-1}) = \left(Q_1(s_0)^T Q_1(s_{-1})\right)^{-1} \tag{5.105}$$

$$\bar{Q}_1(s_{-1}) = Q_1(s_{-1})\left(Q_1(s_0)^T Q_1(s_{-1})\right)^{-1}. \tag{5.106}$$

By linear extrapolation, the secant predictor for $\bar{Q}_1(s_1)$ is

$$\bar{Q}_1^{\text{pred}}(s_1) = Q_1(s_0) + \frac{s_1 - s_0}{s_0 - s_{-1}}\left(Q_1(s_0) - \bar{Q}_1(s_{-1})\right) \tag{5.107}$$

The Riccati unknown has the form $Y(s) = Q_2(s_0)^T \bar{Q}_1(s)$ with $Y(s_0) = 0$, so we can rewrite the predictor (5.107) as

$$Y_0(s_1) = -\frac{s_1 - s_0}{s_0 - s_{-1}} Y(s_{-1}), \tag{5.108}$$

where

$$Y(s_{-1}) = Q_2(s_0)^T \bar{Q}_1(s_{-1}). \tag{5.109}$$

We similarly write higher-order polynomial predictors by choosing a consistent normalization for several steps and using polynomial extrapolation.

Direct Newton corrector iterations

One way to find $\bar{Q}_1(s)$ is to simultaneously solve residual equations for the the eigensystem and the normalization:

$$R = \begin{bmatrix} A(s)\bar{Q}_1(s_1) - \bar{Q}_1(s_1)\bar{T}_{11}(s_1) \\ Q_1(s_0)^T \bar{Q}_1(s_1) - I \end{bmatrix} = 0 \tag{5.110}$$

We can compute a Newton step for (5.110) using a bordered Bartels-Stewart algorithm [27]. Alternately, we can eliminate $\bar{T}_{11}(s_1)$ and perform Newton iteration on the Riccati equation (5.31). A Newton step for the Riccati equation can be solved using an ordinary Bartels-Stewart algorithm [81, p. 367].

Newton iterations on the reduced and unreduced systems are equivalent in exact arithmetic, assuming that the initial iterate in the unreduced case satisfies the normalization condition $Q_1(s_0)^T \bar{Q}_1^{\text{pred}}(s_1) = I$. However, while reducing (5.110) to a Riccati equation reduces the problem size by a modest amount, the reduced system will usually be dense, even if (5.110) is sparse. For small problems, we use dense methods, and the loss of sparsity matters little; for large problems, we sidestep the issue by using projection methods, as described in Chapter 5.4. For medium-sized problems, it may be better to use sparse direct solvers to take Newton steps on the unreduced system of equations.

5.3.3 Normalizing the solution

After we compute a basis $\bar{Q}_1(s_1)$ for $\mathcal{R}(s_1)$, we normalize to find another basis $Q_1(s_1)$ which is as near as possible to $Q_1(s_0)$. This normalization approximates the minimal arclength condition described in Section 5.2.1. We describe several ways to write the normalization in the following lemma.

Lemma 13. *Let $\bar{Q}_1(s_1)$ be a basis for $\mathcal{R}(s_1)$ with $Q_1(s_0)^T\bar{Q}_1(s_1) = I$. Let $\bar{Q}_1(s_1) = U\Sigma V^T$ be a singular value decomposition with $U \in \mathbb{R}^{n\times m}$ and $\Sigma, V \in \mathbb{R}^{m\times m}$, and let $Y(s_1) = Q_2(s_0)\bar{Q}_1(s_1)$. Then the orthonormal basis $Q_1(s_1) \in \mathrm{Stief}(n,m)$ for $\mathcal{R}(s_1)$ which minimizes $\|Q_1(s_1) - Q_1(s_0)\|_F$ can be written in the following ways:*

$$Q_1(s_1) = UV^T \tag{5.111}$$

$$Q_1(s_1) = \bar{Q}_1(s_1)\left(\bar{Q}_1(s_1)^T\bar{Q}_1(s_1)\right)^{-1/2} \tag{5.112}$$

$$Q_1(s_1) = Q_1(s_0)\begin{bmatrix} I \\ Y(s_1) \end{bmatrix}\left(I + Y(s_1)^TY(s_1)\right)^{-1/2}. \tag{5.113}$$

Proof. If $\bar{Q}_1(s_1) = U\Sigma V^T$, then one orthonormal basis for $\mathcal{R}(s_1)$ is UV^T. We can write any other orthonormal basis for $\mathcal{R}(s_1)$ as UV^TW for some orthogonal matrix $W \in O(m)$.

Now we solve an orthogonal Procrustes problem ([81, p. 582]) to find W corresponding to the orthonormal basis nearest Q_0. Choose W to minimize

$$\|Q_1(s_0) - UV^TW\|_F^2. \tag{5.114}$$

Because the Frobenius norm is invariant under unitary transformations, we have

$$\begin{aligned}
&\left\|Q_1(s_0) - UV^TW\right\|_F^2 \\
&= \left\| Q(s_0) \left(\begin{bmatrix} I_m \\ 0 \end{bmatrix} - \begin{bmatrix} Q_1(s_0)^T UV^TW \\ Q_2(s_0)^T UV^TW \end{bmatrix} \right) \right\|_F^2 \\
&= \left\| \begin{bmatrix} I_m - Q_1(s_0)^T UV^TW \\ -Q_2(s_0)^T UV^TW \end{bmatrix} \right\|_F^2
\end{aligned}$$

and by the Pythagorean theorem,

$$\begin{aligned}
\left\|Q_1(s_0) - UV^TW\right\|_F^2 &= \left\|I_m - Q_1(s_0)^T UV^TW\right\|_F^2 + \\
&\quad \left\|-Q_2(s_0)^T UV^TW\right\|_F^2 .
\end{aligned}$$

The second term of the sum does not depend on W, since W is orthogonal. Therefore, we minimize $\left\|Q_1(s_0) - UV^TW\right\|_F^2$ by minimizing

$$\left\|I_m - Q_1(s_0)^T UV^TW\right\|_F^2 \tag{5.115}$$

By hypothesis,

$$I = Q_1(s_0)^T \bar{Q}_1(s_0) = Q_1(s_0)^T U \Sigma V^T. \tag{5.116}$$

If we substitute (5.116) into (5.115) and use the unitary invariance of the Frobenius norm yet again, we have

$$\begin{aligned}
\left\|I_m - Q_1(s_0)^T UV^TW\right\|_F^2 &= \left\|Q_1(s_0)^T U \left(\Sigma - V^TWV\right) V^T\right\|_F^2 \\
&= \left\|\Sigma - V^TWV\right\|_F^2
\end{aligned}$$

The matrix V^TWV is orthogonal, and the closest orthogonal matrix to the positive diagonal matrix Σ is the identity. Therefore, (5.115) is minimized when $V^TWV = I$. Thus $\|Q_1(s_0) - UV^TW\|_F^2$ is minimized for $W = I$, and so $Q_1(s_1) = UV^T$. This proves (5.111).

To show (5.112), we write

$$\begin{aligned}
\bar{Q}_1(s_1)\left(\bar{Q}_1(s_1)^T\bar{Q}_1(s_1)\right)^{-1/2} &= U\Sigma V^T\left(V\Sigma U^T U\Sigma V^T\right)^{-1/2} \\
&= U\Sigma V^T\left(V\Sigma^2 V^T\right)^{-1/2} \\
&= U\Sigma V^T V\left(\Sigma^2\right)^{-1/2} V^T \\
&= UV^T \\
&= Q_1(s_1)
\end{aligned}$$

If we write

$$\bar{Q}_1(s_1) = Q(s_0)\begin{bmatrix} I \\ Y(s_1) \end{bmatrix}, \tag{5.117}$$

then

$$\begin{aligned}
\bar{Q}_1(s_1)^T\bar{Q}_1(s) &= \begin{bmatrix} I \\ Y(s_1) \end{bmatrix}^T Q(s_0)^T Q(s_0)\begin{bmatrix} I \\ Y(s_1) \end{bmatrix} \\
&= \left(I + Y(s_1)^T Y(s_1)\right)
\end{aligned} \tag{5.118}$$

Now substitute (5.117) and (5.118) into (5.112) to get (5.113).

□

5.3.4 Subspace analysis and adaptation

Bifurcations and overlaps

When the CIS algorithm is initialized, the set $\Lambda_1(s_0)$ contains all the unstable eigenvalues of $A(s_0)$ and a few of the stable eigenvalues nearest the imaginary axis. The set $\Lambda_2(s_0)$ lies strictly left of $\Lambda_1(s_0)$ in the complex plane. During continuation, eigenvalues from $\Lambda_1(s)$ may cross the imaginary axis (a bifurcation), or $\Lambda_2(s)$ may cease to lie strictly to the left of $\Lambda_1(s)$ (an overlap). These situations are illustrated

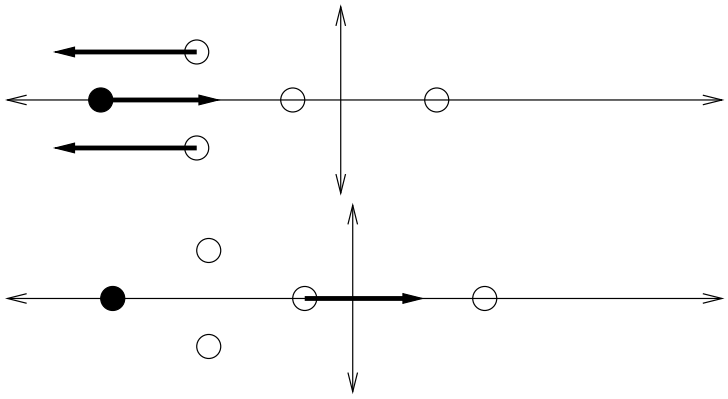

Figure 5.7. Examples of overlap and bifurcation. In the top example (overlap), one of the eigenvalues from $\Lambda_1(s)$ (open circles) changes position with one of the eigenvalues of $\Lambda_2(s)$. In the bottom example, an eigenvalue crosses over the imaginary axis (a bifurcation), so that $\Lambda_1(s)$ contains fewer stable eigenvalues.

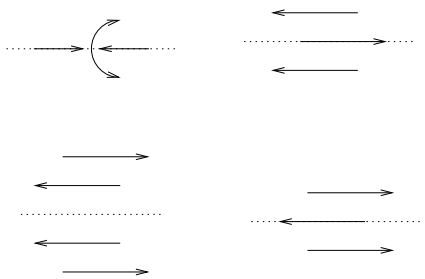

Figure 5.8. Generic overlap situations. On the left, two real eigenvalues collide and produce a complex pair (top), and the real parts of two complex conjugate eigenvalue pairs change order (bottom). On the right, a complex conjugate pair and a real eigenvalue change places in two ways.

in Figure 5.7. When bifurcation or overlap occurs, we reinitialize the continuation procedure.

A *generic* overlap or bifurcation is one which persists when the function $A(s)$ is perturbed. For steady-state continuation problems, the only generic bifurcations are fold bifurcations, in which an isolated real eigenvalue crosses the imaginary axis; and Hopf bifurcations, in which an isolated complex conjugate pair of eigenvalues crosses the imaginary axis. There are four generic types of overlap (see Figure 5.8). In three cases, a single real eigenvalue or complex conjugate pair from $\Lambda_2(s)$ moves

right of some element of $\Lambda_1(s)$. In the fourth case, a single eigenvalue from $\Lambda_2(s)$ collides with an eigenvalue from $\Lambda_1(s)$ to form a complex conjugate pair. $Q_1(s)$ corresponding to $\Lambda_1(s)$ will cease to be continuously defined, and we expect that the Newton iteration will not converge. Complex conjugate eigenvalues in the spectrum may also generically collide and become real eigenvalues, but because we do not allow complex conjugate pairs to be split between $\Lambda_1(s)$ and $\Lambda_2(s)$, this behavior does not result in an overlap.

Step size and subspace adaptation

Standard bifurcation analysis algorithms [85] involve computing functions of $A(s)$. We adapt these methods to large problems by computing the same functions of the much smaller $T_{11}(s)$. Therefore, we try to ensure that only eigenvalues from $\Lambda_1(s)$ can cross the imaginary axis, so that $T_{11}(s)$ will provide all the relevant information about bifurcations. To prevent eigenvalues from $\Lambda_2(s)$ from crossing the imaginary axis, we adapt the step size and the size of the $\Lambda_1(s)$ so that overlaps and bifurcations are not allowed in the same step. We summarize the step size and subspace adaptation logic in Algorithm 2.

When an overlap occurs because two real eigenvalues collide to form a conjugate pair, the Newton iteration will fail to converge. To detect other types of overlap at s, we compute the overlap set:

$$\{(\lambda_i(s), \lambda_j(s)) \in \Lambda_1(s) \times \Lambda_2(s) : \mathrm{Re}(\lambda_i(s)) < \mathrm{Re}(\lambda_j(s))\}\,.$$

If this set is non-empty, then an overlap has occurred. To decide whether multiple overlaps have occurred, we count the number of $(\lambda_i(s), \lambda_j(s))$ pairs in the overlap set. To avoid double-counting overlaps involving complex conjugate pairs, we only count the pairs such that $\mathrm{Im}(\lambda_i(s)) \leq 0$ and $\mathrm{Im}(\lambda_j(s)) \leq 0$.

Only one overlap is allowed in a step. If we detect multiple overlaps, we retry with a smaller step size until only one overlap is left. If we reach the minimum step size and still have multiple overlaps, we reinitialize the continuation process at s_i so that the overlap set from the failed step belongs entirely to $\Lambda_1(s_i)$ or entirely to $\Lambda_2(s_i)$.

We detect bifurcations by counting the unstable eigenvalues. If the total number of unstable eigenvalues at s_{i+1} differs from the total number of unstable eigenvalues at s_i, then a bifurcation occurred during the step. If this total number changed by more than one real eigenvalue or one complex conjugate eigenpair, we assume that multiple bifurcations have occurred, and we try to resolve them by decreasing the step size. If we cannot resolve the behavior with the minimum step size, then the algorithm fails with a diagnostic message. Unless we fail or a bifurcation and an overlap both occur during the step, we assume that $\Lambda_1(s)$ contains all information about bifurcations.

If an overlap or bifurcation occurs in an accepted step from s_i to s_{i+1}, we will reinitialize the computation at s_{i+1} before attempting another step. This way, the new spectral sets will not overlap, and the new $\Lambda_1(s_{i+1})$ will include no more or fewer eigenvalues than necessary after a bifurcation.

5.4 The CIS algorithm: projection methods

We now turn to the case when the dimension n of $A(s)$ is large and we are interested in a space $\mathcal{R}(s)$ of dimension $m \ll n$. In this case, direct methods are expensive; however, if we can multiply by $A(s)$ quickly, we can use projection methods.

5.4.1 Choosing a projection space

In the direct case, we consider two spectral sets: $\Lambda_1(s)$, which contains the unstable eigenvalues and a few of the rightmost stable eigenvalues; and $\Lambda_2(s)$, which contains

Algorithm 2 Continue and adapt invariant subspace of $A(s)$

Input:	$A(s)$	{matrix-valued function}
	s_0,	{starting parameter}
	h_{initial}, $h_{\min}$, $h_{\max}$	{starting step size, step size bounds}
Output:	$Q(s)$ and $T(s)$	

Compute initial point $Q(s_0)$, $T(s_0)$ using Algorithm 1.

$s := s_0$, $h := h_{\text{initial}}$

while not done **do**

Compute a candidate step and candidate step size $\hat{h}$

Test for bifurcation and overlap

if subspace did not converge **then**

Reinitialize at s using Algorithm 1 and reset step size to h_{initial}

else if multiple overlap, multiple bifurcation, or overlap and bifurcation **then**

if $h > h_{\min}$ **then**

Decrease h

else if multiple bifurcation **then**

error "Could not resolve nongeneric bifurcation"

else

Reinitialize at s using Algorithm 1

end if

else

Record the decomposition and diagnostic information

$s := s + h$, $h := \min(h_{\max}, \hat{h})$

if bifurcation or overlap occurred in accepted step **then**

Reinitialize at s using Algorithm 1 and reset step size to h_{initial}

end if

end if

end while

the remaining eigenvalues. In the projection case, we consider three spectral sets: $\Lambda_1(s)$, a set of m elements which contains the unstable eigenvalues and a few of the rightmost stable eigenvalues; $\Lambda_2(s)$, a set of $p-m$ elements which contains a few of the rightmost eigenvalues not in $\Lambda_1(s)$; and $\Lambda_3(s)$, a set of $n-p$ elements which contains the remainder of the spectrum. Our basic strategy in the projected CIS algorithm is to build a projection space $\mathcal{V}$ of dimension $p > m$ such that $p \ll n$ and the restriction of $A(s)$ to $\mathcal{V}$ provides good approximations to $\Lambda_1(s)$ and $\Lambda_2(s)$.

5.4.2 Initialization

During initialization, we may not know how large $\mathcal{V}$ must be to find all the unstable eigenvalues plus a few stable eigenvalues. Therefore, the projected version of the initialization routine calls Algorithm 1 in a loop. While not enough stable eigenvalues converge or there are no sufficiently large gaps between stable eigenvalues in the converged part spectrum, more eigenvalues are requested. If a suitable subspace cannot be found when a specified maximum number of eigenvalues are requested, the code exits with a diagnostic message.

5.4.3 Projected normalization and residual equations

Suppose $V \in \mathbb{R}^{p\times n}$ is an orthonormal basis for a projection space $\mathcal{V}$. Recall the n-by-m residual equation (5.35)

$$A(s)\bar{Q}_1(s) - \bar{Q}_1(s)\bar{T}_{11}(s) = 0.$$

We approximate the equation by assuming that $\bar{Q}_1(s) \approx \bar{Q}_1^h(s) := V\hat{Q}_1(s)$ and choosing $\bar{Q}_1^h(s)$ to satisfy the Galerkin condition

$$\begin{aligned} 0 &= V^T\left(A(s)\bar{Q}_1^h(s) - \bar{Q}_1^h(s)\bar{T}_{11}^h(s)\right) && (5.119)\\ &= V^T A(s)V\hat{Q}_1(s) - \hat{Q}_1(s)\bar{T}_{11}^h(s) && (5.120)\end{aligned}$$

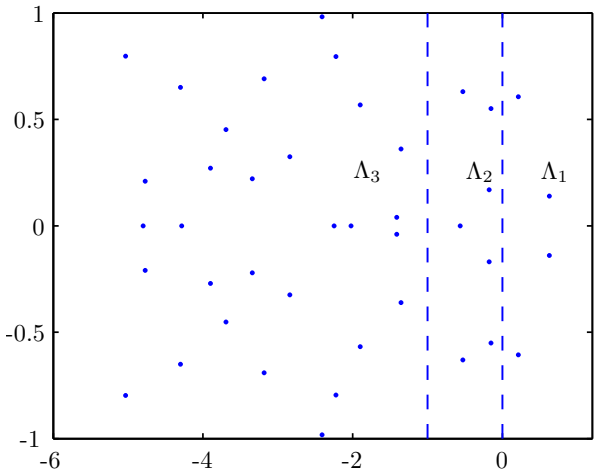

Figure 5.9. Eigenvalue sets in the projected CIS algorithm. In practice, Λ_3 will contain many more eigenvalues than Λ_1 and Λ_2.

We assume the same normalizing condition we used before:

$$Q_1(s_0)^T \bar{Q}_1^h(s) = \left(V^T Q_1(s_0)\right)^T \hat{Q}_1(s) = I \tag{5.121}$$

Once $\bar{Q}_1^h(s)$ has been computed, we can use Lemma 13 to compute the orthonormal basis $Q_1^h(s)$ for the same space which is closest to $Q_1(s_0)$ in the Frobenius norm. We will let $Q_2^h(s) \in \mathbb{R}^{n\times(p-m)}$ be an orthonormal basis for the orthogonal complement of $\text{span}(Q_1^h(s))$ in $\mathcal{V}$. Though we require continuity of $Q_1^h(s)$, it will not be important for our purposes to continuously define $Q_2^h(s)$.

We typically will use a projection space $\mathcal{V}$ which is itself an approximate maximal invariant subspace computed by an Arnoldi method. Suppose that $A(s_1)\mathcal{V} \subset \mathcal{V}$, and let $V^\perp \in \mathbb{R}^{n\times(n-p)}$ be an orthonormal basis for $\mathcal{V}^\perp$. Then at s_1, solutions to the Galerkin equation (5.120) span invariant subspaces of $A(s_1)$.

If $\mathcal{V}$ is a p-dimensional maximal invariant subspace corresponding to the rightmost part of the spectrum of $A(s_1)$, then we compute the leading two-by-two part of a

three-by-three block Schur form

$$\begin{aligned} A(s_1) &= \begin{bmatrix} Q_1^h(s_1) & Q_2^h(s_1) & V^\perp \end{bmatrix} \\ &\quad \begin{bmatrix} T_{11}^h(s_1) & T_{12}^h(s_1) & T_{13}^h(s_1) \\ 0 & T_{22}^h(s_1) & T_{23}^h(s_1) \\ 0 & 0 & T_{33}^h(s_1) \end{bmatrix} \\ &\quad \begin{bmatrix} Q_1^h(s_1) & Q_2^h(s_1) & V^\perp \end{bmatrix}^T \end{aligned}$$

The spectrum of the $T_{11}^h(s)$ block is the continued set of eigenvalues $\Lambda_1(s)$. The $T_{22}^h(s)$ block has a few of the rightmost remaining eigenvalues, which we use to diagnose overlap. The eigenvalues of the uncomputed block $T_{33}^h(s)$ are part of the spectrum which lies further from the imaginary axis. Figure 5.9 illustrates the three spectral sets corresponding to $T_{11}^h(s)$, $T_{22}^h(s)$, and $T_{33}^h(s)$ in the case when no overlap has occurred.

As in the dense case, we can eliminate $\bar{T}_{11}^h(s)$ from equation (5.120); we summarize this calculation in the following lemma.

Lemma 14. *Let $V^T Q_1(s_0)$ have the singular value decomposition*

$$V^T Q_1(s_0) = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} R^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} R^T \tag{5.122}$$

where $U \in \mathbb{R}^{p \times p}$, $\Sigma \in \mathbb{R}^{m \times m}$, and $R \in \mathbb{R}^{m \times m}$. Let

$$\widehat{T}^h(s) = \begin{bmatrix} \widehat{T}_{11}^h(s) & \widehat{T}_{12}^h(s) \\ E_{21}^h(s) & \widehat{T}_{22}^h(s) \end{bmatrix} := \begin{bmatrix} \Sigma & 0 \\ 0 & I \end{bmatrix} U^T V^T A(s) V U \begin{bmatrix} \Sigma^{-1} & 0 \\ 0 & I \end{bmatrix} \tag{5.123}$$

Then any solution to the Galerkin equation (5.120) and normalizing condition (5.121) can be written as

$$\hat{Q}_1(s) = U \begin{bmatrix} \Sigma^{-1} \\ \hat{Y}^h(s) \end{bmatrix} R^T \tag{5.124}$$

where $\hat{Y}^h(s) \in \mathbb{R}^{(p-m)\times m}$ *is a solution to the Riccati equation*

$$F^h(Y^h(s)) := \widehat{T}^h_{22}(s)Y^h(s) - Y^h(s)\widehat{T}^h_{11}(s) + E^h_{21}(s) - Y^h(s)\widehat{T}^h_{12}(s)Y^h(s) = 0. \quad (5.125)$$

Proof. Let $B(s) = U^T\hat{Q}^h_1(s)R$. Substituting the SVD (5.122) into (5.121), we have

$$\begin{aligned} I &= R\begin{bmatrix}\Sigma & 0\end{bmatrix}U^T\hat{Q}_1(s) && (5.126)\\ &= R\begin{bmatrix}\Sigma & 0\end{bmatrix}B(s)R^T && (5.127)\end{aligned}$$

If we multiply on the left by R^T and on the right by R, we have

$$I = \begin{bmatrix}\Sigma & 0\end{bmatrix}B(s) \quad (5.128)$$

Therefore, for some $Y^h(s) \in \mathbb{R}^{(p-m)\times m}$, $B(s)$ can be written as

$$B(s) = \begin{bmatrix}\Sigma^{-1}\\ Y^h(s)\end{bmatrix} = \begin{bmatrix}\Sigma^{-1} & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}I\\ Y^h(s)\end{bmatrix} \quad (5.129)$$

Now we substitute $\hat{Q}^h_1(s) = UB(s)R^T$ into the projected residual equation (5.120):

$$V^TA(s)VU\begin{bmatrix}\Sigma^{-1} & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}I\\ Y^h(s)\end{bmatrix}R^T - U\begin{bmatrix}\Sigma^{-1} & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}I\\ Y^h(s)\end{bmatrix}R^T\bar{T}^h_{11}(s) = 0 \quad (5.130)$$

If we multiply by $\begin{bmatrix}\Sigma & 0\\ 0 & I\end{bmatrix}U^T$ on the left and by R on the right, we have

$$\widehat{T}^h(s)\begin{bmatrix}I\\ Y^h(s)\end{bmatrix} = \begin{bmatrix}I\\ Y^h(s)\end{bmatrix}\left(R^T\bar{T}^h_{11}(s)R\right) \quad (5.131)$$

The first row of (5.131) gives an expression for $R^T\bar{T}^h_{11}(s)R$, which we can substitute into the second row to get the Riccati equation (5.125):

$$\begin{aligned} R^T\bar{T}^h_{11}(s)R &= \widehat{T}^h_{11}(s) + \widehat{T}^h_{12}(s)Y^h(s)\\ E^h_{21}(s) + \widehat{T}^h_{22}(s)Y^h(s) &= Y^h(s)\left(R^T\bar{T}^h_{11}(s)R\right)\\ &= Y^h(s)\widehat{T}^h_{11}(s) + Y^h(s)\widehat{T}^h_{12}(s)Y^h(s)\end{aligned}$$

□

In Theorem 3, we saw that for s sufficiently near s_0, the normalized basis for $\mathcal{R}(s)$ corresponded to the minimum norm solution for the Riccati equation (5.31). The norm of the Riccati unknown $Y(s)$ is equal to the distance $\|\bar{Q}_1(s) - Q_1(s_0)\|_F$. We now show that $\|Y^h(s)\|_F$ is similarly related to $\|\bar{Q}_1^h(s) - Q_1(s_0)\|_F$.

Lemma 15. *In the previous lemma, the distance from $\bar{Q}_1^h$ to $Q_1(s_0)$ is*

$$\|\bar{Q}_1^h(s) - Q_1(s_0)\|_F^2 = \|Y^h\|_F^2 + \|\Sigma^{-1}\|_F^2 - m \tag{5.132}$$

Proof. We decompose $Q_1(s_0)$ and $\bar{Q}_1^h(s)$ into components in three orthogonal spaces spanned by $V^\perp$, VU_1, and VU_2:

$$\begin{aligned} Q_1(s_0) &= V^\perp(V^\perp)^T Q_1(s_0) + VU_1\Sigma R^T & (5.133)\\ Q_1^h(s) &= VU_1\Sigma^{-1}R^T + VU_2Y^h(s)R^T & (5.134) \end{aligned}$$

where the first equation is a consequence of (5.122) and the second equation follows from (5.124). The difference is

$$Q_1(s_0) - Q_1^h(s) = \left\{ \begin{array}{l} V^\perp(V^\perp)^T Q_1(s_0) + \\ VU_1(\Sigma - \Sigma^{-1})R^T + \\ VU_2Y_h(s)R^T \end{array} \right\}. \tag{5.135}$$

Because the three components are orthogonal, the squared Frobenius norm is the sum of the squares of the Frobenius norms; that is

$$\|Q_1(s_0) - Q_1^h(s)\|_F^2 = \left\{ \begin{array}{l} \|V^\perp(V^\perp)^T Q_1(s_0)\|_F^2 + \\ \|VU_1(\Sigma - \Sigma^{-1})R^T\|_F^2 + \\ \|VU_2Y_h(s)R^T\|_F^2 \end{array} \right\}. \tag{5.136}$$

Because multiplication by an orthonormal matrix does not change the Frobenius

norm, we can write

$$\|Q_1(s_0) - Q_1^h(s)\|_F^2 = \left\{ \begin{array}{l} \|(V^\perp)^T Q_1(s_0)\|_F^2 + \\ \|\Sigma - \Sigma^{-1}|_F^2 + \\ \|Y_h(s)\|_F^2 \end{array} \right\} \tag{5.137}$$

$$= \left\{ \begin{array}{l} \|(V^\perp)^T Q_1(s_0)\|_F^2 + \\ (\|\Sigma\|_F^2 + \|\Sigma^{-1}|_F^2 - 2m) + \\ \|Y_h(s)\|_F^2 \end{array} \right\}. \tag{5.138}$$

Note that

$$m = \|Q_1(s_0)\|_F^2 = \|(V^\perp)^T Q_1(s_0)\|_F^2 + \|V^T Q_1(s_0)\|_F^2 \tag{5.139}$$

$$= \|(V^\perp)^T Q_1(s_0)\|_F^2 + \|\Sigma\|_F^2. \tag{5.140}$$

Now substitute

$$\|(V^\perp)^T Q_1(s_0)\|_F^2 = m - \|\Sigma\|_F^2 \tag{5.141}$$

into (5.138) to obtain the desired result.

□

Therefore, if s_1 is sufficiently near s_0 and $\mathcal{V}$ is itself an invariant subspace of $A(s_1)$ such that $\mathcal{R}(s_1) \subset \mathcal{V}$, the minimal norm solution to the projected Riccati equation (5.125) corresponds exactly to the minimal norm solution to the Riccati equation (5.31).

5.4.4 Projected predictors and correctors

The Euler predictor (5.101) and the finite difference version of the Euler predictor (5.102) are subtly different in the projected case. A projection subspace $\mathcal{V}$ which is an invariant subspace for $A(s_1)$ will generally not contain $\mathcal{R}(s_0)$; consequently, $Q_1(s_0)$ will not correspond to a solution to the projected Riccati equation (5.125)

at $s = s_0$. Worse, $E_{21}^h(s_0)$ will usually be nonzero. If we naively differentiate the relation $F^h(Y^h(s)) = 0$ and use the resulting differential equation to form an Euler-like approximation $Y_0^h(s_1)$ starting from a value of 0 for $Y^h(s_0)$, then to first order $F^h(Y_0^h(s_1))$ will be $E_{21}(s_0)$.

We can remedy this problem by requiring $\mathcal{R}(s_0) \subset \mathcal{V}$. However, a more straightforward alternative is to compute a secant prediction $\bar{Q}_1^{\text{pred}}(s_1)$ using (5.107), and then project

$$\bar{Q}_1^{h,\text{pred}}(s_1) = VV^T\bar{Q}_1^{\text{pred}}(s_1). \tag{5.142}$$

The corresponding projected Riccati predictor is then

$$Y_0^h(s_1) = U_2^T V^T \bar{Q}_1^{\text{pred}}(s_1)R \tag{5.143}$$

In the current code, we use the trivial predictor $Y_0^h(s_1) = 0$.

Once we have a predicted value $Y_0^h(s_1)$, we solve the projected Riccati equation with a Newton iteration, just as we did in the direct methods. We note that the projected matrix $V^T A(s)V$ will usually be dense, and so there seems to be little benefit to solving the unreduced equations. Just as in the direct case, alternate subspace selection methods based on eigenvalues and eigenvectors are possible.

5.5 Integrating the CIS algorithm into *MATCONT*

In the introduction, we described how invariant subspace continuation can be used to adapt bifurcation analysis methods for small problems in order to analyze much larger systems. In this section, we discuss one example of our work to use the CIS algorithm in this way to extend the bifurcation analysis code MATCONT [61]: using projected test functions to detect and locate Hopf bifurcations.

Let $x(s) = (u(s), \alpha(s)) \in \mathbb{R}^n \times \mathbb{R}$ be a smooth local parameterization of a solution branch of the stationary problem (5.2):

$$f(x(s)) = f(u(s), \alpha(s)) = 0.$$

We write the Jacobian matrix along this path as $A(s) := f_u(x(s))$. A solution point $x(s_0)$ is a *bifurcation point* if $\operatorname{Re} \lambda_i(s_0) = 0$ for at least one eigenvalue $\lambda_i(s_0)$ of $A(s_0)$. The point $x(s_0)$ is a *simple Hopf bifurcation* if the simple eigenvalue $\lambda_i(s_0)$ is a pure imaginary number and $\operatorname{Re}\left(\frac{d\lambda_i}{ds}(s_0)\right) \neq 0$.

A *test function* $\phi(s) := \psi(x(s))$ is a (typically) smooth scalar function that has a regular zero at a bifurcation point. A bifurcation point between consecutive continuation points $x(s_k)$ and $x(s_{k+1})$ is *detected* when

$$\psi(x(s_k))\psi(x(s_{k+1})) < 0. \tag{5.144}$$

Once a bifurcation point has been detected, it can be *located* by solving the system

$$\begin{cases} f(x) = 0, \\ g(x) = 0 \end{cases} \tag{5.145}$$

for an appropriate function g

The test function used in MATCONT to detect Hopf points is

$$\psi_{\text{Hopf}}(x(s)) := \det\left[2A(s) \odot I_n\right] = \prod_{i>j} \left(\lambda_i(s) + \lambda_j(s)\right), \tag{5.146}$$

where $\odot$ is the bialternate product [85]. Clearly $\psi_{HM}(x(s))$ is zero if $A(s)$ has a pure imaginary pair of eigenvalues $(\pm i\kappa)$; but note that ψ_{HM} is also zero if $A(s)$ has a pair of real eigenvalues which sum to zero $(\pm\kappa)$.

To detect Hopf bifurcations using the CIS algorithm, we introduce the test functions

$$\psi^{(1)}_{\text{Hopf}}(x(s)) = \prod_{m \geq i > j} \left(\lambda_i(s) + \lambda_j(s)\right), \tag{5.147}$$

$$\psi^{(2)}_{\text{Hopf}}(x(s)) = (-1)^{\#\left\{\lambda_i(s):\operatorname{Re}\lambda_i(s)\geq 0 \text{ and } \operatorname{Im}\lambda_i(s)>0\right\}}. \tag{5.148}$$

where $\lambda_1(s), \ldots, \lambda_m(s)$ are the eigenvalues of $T_{11}(s)$. We detect a Hopf bifurcation when

$$\psi^{(1)}_{\text{Hopf}}(x(s_k))\psi^{(1)}_{\text{Hopf}}(x(s_{k+1})) < 0 \text{ and } \psi^{(2)}_{\text{Hopf}}(x(s_k))\psi^{(2)}_{\text{Hopf}}(x(s_{k+1})) < 0 \tag{5.149}$$

The test function $\psi^{(1)}_{\text{Hopf}}$ is analogous to ψ_{Hopf}, while $\psi^{(2)}$ is used to exclude the case of two real eigenvalues that sum to zero.

A well-known method to locate a Hopf point (see e.g. e.g. [110, 85, 26]) is to solve the system

$$\begin{cases} f(x) = 0, \\ f_u(x)r - i\omega r = 0, \\ r^* r_0 - 1 = 0 \end{cases} \tag{5.150}$$

where $x \in \mathbb{R}^{n+1}$, $r \in \mathbb{C}^n$, and $\omega \in \mathbb{R}$. The reference vector $r_0 \in \mathbb{C}^n$ is given. Usually, the system (5.150) is converted to a system of $3n + 2$ real unknowns. Based on the CIS algorithm, we replace (5.150) with the system

$$\begin{cases} f(x) = 0, \\ T_{11}(x)r - i\omega r = 0, \\ r^* r_0 - 1 = 0 \end{cases} \tag{5.151}$$

where r is now a vector in $\mathbb{C}^m$. In contrast to (5.150), the system (5.151) involves $n + 2m + 2$ real unknowns.

Example 16. *The* 1D Brusselator *[113] is a well known model system for autocatalytic chemical reactions with diffusion. The problem is defined on* $\Omega = (0, 1)$ *by coupled differential equations for unknowns* u *and* v

$$\begin{aligned} \frac{d_1}{l^2}u'' - (b+1)u + u^2 v + a &= 0 \\ \frac{d_2}{l^2}v'' + bu - u^2 v &= 0 \end{aligned}$$

with boundary conditions

$$u(0) = u(1) = a \text{ and } v(0) = v(1) = \frac{b}{a}. \tag{5.152}$$

This problem exhibits a rich bifurcation scenario and has been used in the literature as a standard model for bifurcation analysis [147, 82, 55, 15, 45, 124]. Utilizing a second-order finite difference discretization

$$f'' \approx \frac{1}{h^2}(f_{i-1} - 2f_i + f_{i+1})$$

with $h = (N+1)^{-1}$*, the resulting discrete problem can be written in the form (5.2). This discretization of the Brusselator is used in a MATCONT example [61].*

In order to verify the accuracy of locating a Hopf point, we continue a constant solution branch: $u(x) = a$*,* $v(x) = \frac{b}{a}$*, with respect to* b*. In this case the values of* b *where Hopf bifurcation occurs are known analytically as a function of* N*, see e.g. [45, Eq. (24)]. In the table below* $n = 2N$ *is the dimension of the system (5.2);* t_{total} *and* t_{CIS} *are the average total time and the CIS time per continuation step, respectively, and* t_H *is the time for finding the value of* b *for which Hopf bifurcation occurs to least eight correct digits. The computations are performed on an 850 MHz Pentium III machine.*

n	256	512	1024	2048	4096
$t_{total} \frac{\text{sec.}}{step}$	1.13	1.56	2.20	4.35	13.8
$t_{CIS} \frac{\text{sec.}}{step}$	1.01	1.19	1.25	1.29	1.41
t_H sec.	9.1	9.5	10.7	18.8	39

These preliminary results indicate that the relative cost of the CIS computations decreases as n increases, and that for large n the cost of one Newton step for locating a Hopf bifurcation is approximately the same as that of one Newton step for a continuation step of (5.2). For this example, typically one Newton step was required in the corrector at each continuation step, while locating the Hopf bifurcation took three Newton steps on the extended system.

5.6 Conclusions and Future Work

In this paper, we have discussed the CIS algorithm for computing a smooth orthonormal basis for an invariant subspace of a parameter-dependent matrix, and we have extended it to make it more suitable for numerical bifurcation analysis. In particular, we have made the following contributions:

1. We have derived new sufficient conditions for the existence of a continuous invariant subspace connecting invariant subspaces of matrices at the end of a parameterized matrix curve.

2. We have extended the original CIS algorithm for dense problems with logic for adapting the continued subspace in order to ensure that it always includes information relevant to bifurcation analysis. Such adaptation is necessary when an bifurcation occurs or when there is an *overlap*: that is, when the real parts of eigenvalues change order.

3. We have extended our algorithm to work efficiently on large sparse matrices by exploiting Galerkin projection methods. The original CIS algorithm used direct methods for dense matrices, and so cost $O(n^3)$ work at each step.

4. We have incorporated the projection-based CIS algorithm into the MATCONT bifurcation analysis package, and we have applied the combined code to the Brusselator model problem.

Future work includes the following topics. We are still actively investigating how the information can most effectively be used for finding bifurcations from non-static equilibria, and how to best use the CIS algorithm in detecting and computing codimension-2 bifurcations along branches of Hopf and limit points. We are also in-

volved in using of the CIS algorithm in order to study the dependence of resonant frequencies of mechanical devices as design parameters are varied.

Chapter 6

HiQLab

6.1 Introduction

`HiQLab` is a finite element tool for simulating resonant MEMS. The source code is open and freely available [29]. We have included in `HiQLab` the models and algorithms described in previous chapters, including elements for perfectly matched layers, and thermoelastic damping and structure-preserving algorithms for model reduction and for eigencomputations. The main design goals of `HiQLab` are:

- **Flexible problem descriptions**: `HiQLab` was designed with parameter studies and optimization tasks in mind. For these tasks, it is useful to be able to *programatically* describe the problem in a way that naturally exposes relevant parameters.

- **Ease of extension**: The target problems for `HiQLab` involve interactions between different physical domains. To solve these problems easily, it is important to be able to quickly incorporate new elements and to construct new algorithms to take advantage of special problem structures.

- **Openness**: We wanted to share our code with the community and collaborators, and to allow others to build on it for academic or commercial use. It is easiest to do this with a completely open core library that does not depend intrinsically on any particular commercial systems.

While we focus on resonant MEMS, the architecture of `HiQLab` is general. The application-specific parts of the program mostly involve the specific types of structures we preserve in our algorithms and the interface elements seen by the users. The architecture of `HiQLab` incorporates several novel features, including the extensive use of scripting languages, built-in support for problem-specific scaling, and a global shape function facility. Our goal in this chapter is to describe `HiQLab` at a high level, with particular emphasis on these architectural features.

6.2 History

`HiQLab` draws on previous projects in which we have been involved, as well as on several related projects. We described some of these systems in our first chapter; we now review how these systems influenced the design of `HiQLab`.

6.2.1 SUGAR

The SUGAR system for modified nodal analysis of MEMS devices [160] is in many ways the direct ancestor of `HiQLab`. The name and the inspiration of SUGAR were both drawn from SPICE, an enormously successful family of circuit simulation systems which were initially designed at Berkeley and were subsequently developed into commercial tools by several companies. SPICE provides compact models of common circuit components such as resistors, capacitors, and transistors, and configurations of these circuit components are described using an input text file called a *netlist*.

The goal of the SUGAR project, and of related projects such as NODAS [71], was to provide a SPICE-like facility for compactly describing devices in terms of domain-specific elements, including both circuit elements like resistors and capacitors and also mechanical elements like beams and comb drives. In addition, SUGAR was initially designed as a MATLAB program [163], so that new models and solution algorithms might be quickly and easily added to the program.

Early versions of SUGAR were written entirely within MATLAB, which ultimately led to performance problems. In particular, the code to parse SUGAR netlists was written in MATLAB, and parsing input files proved to be the slowest part of many simulations. Also, in the early versions of SUGAR there was no clear distinction between the code for individual elements such as beams and capacitors and the code for general tasks such as assembling a stiffness matrix or a nonlinear solver iteration. We completely redesigned the system for version 2.0 so that there was a clean separation between element routines ("model functions" in SUGAR parlance), assembly and analysis routines, and the routines to process the netlist file. We also wrote a custom language system for SUGAR 2.0, with support for hierarchical design via a subroutine construct (subnets), libraries of parameters for different processes, and parameterized descriptions. We later redesigned SUGAR again, and in this third major version we moved replaced our custom language – which had grown unwieldy over time as we added features – with a modified version of the Lua language [99]. We also moved common routines such as matrix assembly out of MATLAB and into a core library written in C. This new organization proved so convenient that we chose to design `HiQLab` around the same basic architecture: of a core written in a compiled language, user interfaces and high-level analysis routines written in MATLAB, and a mesh description system based on Lua.

6.2.2 FEAPMEX

SUGAR was always designed around a SPICE-like model in which many of the individual elements in the system were unique and connectivity among elements could be irregular. In consequence, SUGAR was not well suited to continuum finite element simulations, which typically involve large numbers of elements with similar or identical properties, connected locally to a few spatially adjacent neighbors in a relatively regular way. As we increasingly became involved in collaborations to simulate the behavior of high-frequency resonant MEMS, we found we needed the capabilities of a conventional finite element code. At the same time, we wanted to continue using the MATLAB environment to test new numerical methods and to script parameter studies. Thus was FEAPMEX inspired.

FEAPMEX is a MATLAB interface to the academic finite element code FEAP [161]. By running FEAP and MATLAB in independent threads and passing control between the threads only when FEAP would normally wait on user input, FEAPMEX is able to provide broad access to the internal data structures used by FEAP with only minimal modifications to the FEAP code base. FEAPMEX provides hooks so that users can, from within MATLAB, access assembled mass and stiffness matrices, evaluate residual forces, and execute FEAP's plotting commands. Users can also control mesh generation from MATLAB, either by defining parameters to be used by FEAP on loading an input deck or by providing a complete list of node positions and element connectivities stored in MATLAB arrays. Thus, FEAPMEX does what it was originally intended to do: it provides a flexible MATLAB scripting interface to FEAP.

While the FEAP provides a wide array of elements and analysis capabilities, it seemed much less flexible than SUGAR when it came to writing new classes of elements and when adding new analysis routines written in compiled languages. Also,

the FEAP mesh description language does not have all the capabilities that were present in Lua. Furthermore, the FEAP source code is only available for a license fee. These limitations were increasingly vexing as we began to work on models of damping in resonant MEMS, for which we wanted new elements, new mesh input facilities, and new solvers.

6.2.3 `HiQLab`

`HiQLab` was conceived as a collection of MATLAB scripts and supporting C libraries which we used for prototyping new finite elements and new solvers. The C libraries initially provided only routines for finite element matrix assembly, which were often a bottleneck even for modest problems when written in MATLAB. As we began to include elements which performed more complicated calculations, we also moved the elements into compiled code. Though our initial intent was to incorporate these new elements into FEAP, and to use them via FEAPMEX, we changed our mind once the C library became sufficiently self-contained. We decided instead to move the mesh description container from MATLAB into the compiled language core and to improve the mesh description capabilities using the Lua language, thus repeating the same architectural pattern that worked so well for SUGAR.

6.3 Architectural overview

The core library of `HiQLab` is written in C++. This core library includes routines to assemble and manipulate the mesh data structure; numerical support libraries of basic matrix operations, shape functions, and quadrature weights; and interfaces through which the core interacts with the element library and the Lua meshing subsystem. In addition, the core library contains interface code to allow the incorporation

of outside numerical libraries such as ARPACK [114] and UMFPACK [56]. This core is compact, with just under 6500 lines of code; most of the core shares a great deal in common with other finite element systems, and below the object-oriented interfaces lie the same data structures as are used in traditional finite element codes, which can be found in standard references [98, 191].

The element library, which is also written in C++, contains elements for elasticity and scalar elements for thermal and electrostatic problems; coupled-field elements for thermoelasticity, piezoelectricity, and electrostatic attraction of elastic bodies; basic circuit elements, including resistors, capacitors, inductors, and voltage sources; and elements to impose general constraints on the system via Lagrange multipliers. The code supports one-dimensional, two-dimensional, axisymmetric, and fully three-dimensional problems. Other than the circuit elements and the Lagrange multiplier elements, all the elements in the system support the perfectly matched layer transformation described earlier in this dissertation.

There are two subsystems which implement a user interface for analysts to script and steer solution algorithms and to retrieve analysis results. The primary such user interface is hosted in MATLAB. Through this interface, one can solve equations for static equilibria, eigenvalue problems, and time-harmonic forced response; one can also plot the results. A second user interface based on Lua is available, though it has fewer features and currently lacks the graphics available from MATLAB. Despite these limitations, the Lua user interface is better than the MATLAB interface for non-interactive uses such as scripting automated tests; for debugging problems in which MATLAB interacts poorly with debugging tools; and for running on systems like the Itanium, where MATLAB is simply unavailable. Whether one uses the MATLAB interface or the Lua user interface, one usually uses Lua scripts to build problem descriptions in `HiQLab`. Glue code, which allows MATLAB and Lua to call the C++

core library in a natural way, is automatically generated from a high-level interface description file using tolua++ [120] and a customized version of Matwrap [91].

6.4 Core objects

The core data structures in `HiQLab` are similar to the core data structures described in standard references on finite element programming [98, 191], so our description of them will be brief. At the heart of `HiQLab` is the mesh object. This object serves as a central container for global state associated with a simulation, including

- Nodal coordinates
- Element connectivities and types
- Boundary condition arrays
- Displacement, velocity, acceleration, and residual force vectors
- Indexing structures for nodal, element, and global variables
- Pointers to the Lua interpreter and the elements owned by the mesh

In addition to methods for reading and writing the mesh data structures, the mesh object also provides functions to assemble residuals and tangent matrices.

The real work in the residual and tangent computations is done by element objects. Elements in `HiQLab` are an instance of the flyweight pattern [77]: most of the state for an element is stored externally, in the mesh data structure arrays, and it is passed into an element object on method calls. Each element object then corresponds to a particular type of physics and a way of computing material responses, things which are

shared by many elements. Element objects provide methods to perform the following tasks:

- Tell the mesh how many branch variables an element needs
- Mark which nodal degrees of freedom the element uses
- Assemble local residual and tangent matrix contributions
- Compute lumped L^2 projections of fields defined at Gauss points
- Evaluate Gauss point stresses
- Manage the mapping between the element order for nodal variables and the global order

Each element in the system implements the interface defined by the `Element` base class, and the core library interacts with the elements only through this interface.

Assembler objects provide a level of indirection between the elements and the data structures which will contain the accumulated element contributions to global vectors and matrices. Matrix assembler objects only allow one to add an element contribution to an existing matrix; vector assembler objects allow one to add or set individual entries. Because of this level of indirection, it is simple to assemble into different types of data structures. By default, element matrices are stored in coordinate form in a buffer; at the end of the assembly loop, the coordinate entries are sorted into column major order, summed in order to eliminate duplicate entries for the same coordinate location, and converted to compressed sparse column form. However, there are also other assemblers which build matrix data structures used by the PETSc [20] and Trilinos [90] parallel libraries.

The remaining routines in the core library perform linear algebra computations; evaluate shape functions and quadrature rules; and support the interfaces used in evaluating Lua callbacks, as we describe below.

6.5 The role of Lua

Lua was first developed in 1993 at PUC-Rio in Brazil, where it originally grew out of two other little languages: a problem description language for numerical simulators, and a configuration language for a report generation program. From the beginning, the language was designed to be small, simple, fast, and easy to embed in C programs. These are precisely the features which led us to choose Lua for use in `HiQLab`. The history of Lua is described in [101], and the language itself is described concisely in the user manual [99] and in an associated book [100]. Because of its speed and simplicity, Lua is widely used for scripting games, as well as in a variety of other applications where embedded scripts are useful.

Lua plays two distinct roles in `HiQLab`. First, Lua can be used as an alternative language to MATLAB for scripting user actions and solvers. The use of scripting languages to steer computations is well-established, and indeed was part of the motivation for the development of Lua. Lua has been used for steering computations in other finite element packages as well, as have languages like Python [142] and Tcl [179]; in older finite element codes, custom interpreted mini-languages often serve the same role.

Lua also serves in `HiQLab` as a problem description language. Even when using the MATLAB user interface, `HiQLab` uses the Lua interpreter to construct the mesh. We use Lua for problem description both because of the flexibility and clarity of the language and because of the ease with which the interpreter can be linked into a

MATLAB extension file; this ease of linking is not shared by Python and Tcl. As with solution descriptions, many finite element codes and circuit simulators provide some simple interpreted language for mesh description. In many simulation systems the problem description language is ad hoc, but other systems employ full-featured scripting languages; examples include the use of Tcl in OpenSees [137, 123], Python in PyFemax [79], Lua in pdelib [75], and Lisp in CADENCE [141]. Use of a full-featured language not only saves on development time, but it also saves simulator developers from the need to also become language designers.

6.5.1 Lua callbacks

To specify a finite element calculation, one needs to know the shape of the domain, the partial differential equations describing what happens on that domain (including boundary conditions), and how both the domain shape and the equations are to be discretized. Different parts of the problem are naturally expressed as functions over the domain: material properties are functions of position in the domain, boundary conditions are expressed as functions over some surface, and measurements of the system often take the form of a functional applied to the solution fields. In a traditional finite element code, these functions are specified in one of two ways. First, one might choose from a small parameterized library of possible functions; for example, it might be possible to specify that a particular field is constant along some flat surface. The problem with this mode of specification is obvious: the library of possible functions will be limited, so that specifying a constant along a flat surface is possible, but specifying a linearly varying field along some curved surface is hard. Second, one might choose to specify conditions node-by-node; for example, the analyst might specify that every hundredth node between 1000 and 2000 has zero displacement. There are two obvious problems with this approach: it is verbose except in the simplest cases;

and it ties the specification of the continuum problem to a particular discretization, so that changes to the mesh require changes to how the boundary conditions are specified.

By using a full-featured language for problem descriptions, one can specify mathematical functions defined on the problem domain in terms of program functions in the description language. The user specifies these functions as part of the problem specification, and when values of the function are needed at a node or a quadrature point in order to perform some computation, the main code can call the user function in order to obtain the value. We use this callback mechanism extensively in `HiQLab`. With every mesh is an associated Lua interpreter which contains the definitions of any callback functions; and by changing the behavior of specific callbacks dynamically, we can play with many problem parameters without reloading the mesh.

Callbacks from C++ to Lua are the basis of most of the novel features of the `HiQLab` problem description language, as we describe below. After describing how useful such callbacks can be, we address concerns about the performance of ubiquitous use of callbacks into an interpreted language like Lua.

Boundary conditions

Consider how essential boundary conditions are described. There are two components to the description of such conditions: first, one must specify the surface or subdomain where the conditions apply, and second one must specify the values to be applied to some field. `HiQLab` allows users to provide callback functions to describe both where the boundary lives and how the boundary values should be assigned. For example, to clamp the x displacement (u_x) along the line $y = 0$, we might write

```
clamp_boundary( function(x,y) return mesheq(x,0) end, 'ux' )
```

while to specify a displacement field which varies linearly with the y coordinate, we might write

```
mesh:set_bc(function(x,y)
  if mesheq(x,0) then  -- If |x| < tol
    return 'u ', a*y   -- x displacement = a*y
  end
end
```

Note that the `--` symbol marks the start of a Lua comment. In the first case, the argument provided to `clamp_boundary` defines an indicator function which is true for nodes on the surface to be clamped, and false elsewhere. In the second case, we specify the type of boundary conditions to be applied using a string return, and the boundary value in subsequent numerical return values. If the ith character of the returned string is `'u'`, we apply an essential boundary condition to the ith degree of freedom associated with the node; if it is `'f'`, we apply a natural condition; and if it is blank, we apply no condition.

`HiQLab` provides functions to apply boundary conditions in several ways. Beside specifying a surface where some value should be clamped or given a specified displacement, as above, one can describe a point load or an electrical ground in a natural way using functions like `point_load` and `circuit.ground`. All of these convenience functions are written in Lua, with a common mechanism to transfer the results from Lua into the C++ mesh. When the time comes to assemble problem boundary conditions, the C++ library calls a Lua function attached to the C++ mesh object. That Lua function in turn calls methods on the C++ mesh object to manipulate the boundary arrays. For example, the code to evaluate a callback function at every nodal point which returns boundary strings and boundary values (as in the second example above) looks like

```
function Mesh:form_nodal_bcs(bcfunc)
  for j = 1,self:numnp() do
```

```
    --
    -- Iterate through the list of boundary conditions (BCs) at
    -- node j, as returned from bcfunc.  For each BC, the
    -- iterator returns
    --    i     - The index of the nodal variable affected
    --    value - The force or displacement for variable i
    --    type  - 'u' or 'f' for displacement or force BC
    --
    for i,value,type in bc_iterator(bcfunc(self:x(j-1))) do
      self:set_bcode(i-1,j-1,type)  -- Record the type of BC
      self:set_bv(i-1,j-1,value)    -- Record the value
    end
  end
end
```

Drive and sense functions

In a typical frequency-response problem, we are interested in evaluating a transfer function like

$$H(\omega) = l^*(K - \omega^2 M)^{-1} f. \tag{6.1}$$

To evaluate $H(\omega)$, we need not only the mass and stiffness matrices, but also the drive function f and the sense functional l^*. Drive and sense functions are specified in `HiQLab` by making a Lua callback, with the mesh object and a vector assembler object as arguments. The vector assembler provides `set` and `add` methods, which can be used respectively to write or to add to an element in the vector under construction. As is the case with boundary conditions, there are several convenience functions written in Lua which can be used with this mechanism.

The `nodal2d_indicator` is one example of a cover function to construct drive and sense vectors. The purpose of this function is to allow the user to specify a vector with one nonzero value corresponding to a particular degree of freedom at a particular node. The node is specified by coordinates, and the degree of freedom can be specified by a string name or by an index (see Section 6.5.3 for information on the `HiQLab` system

for managing degrees of freedom). In its entirety, the `nodal2d_indicator` function is:

```
function nodal2d_indicator(dof,x,y,val)

  --
  -- First, check inputs and assign defaults.  The Lua or operator
  -- returns the first non-nil value, so a line like
  --   val = val or 1
  -- will set val to 1 if the user did not already assign a value.
  --
  assert(dof and x and y, 'Must define dof and node coordinates')
  dof = var_slots[dof] or dof
  val = val or 1

  --
  -- Return a function that acts on a vector assembler object by
  -- writing into the variable specified by the (dof,x,y) arguments
  --
  return function(mesh,v)
    for j = 1,mesh:numnp() do
      if mesheq(x,mesh:x(0,j-1)) and mesheq(y,mesh:x(1,j-1)) then
        v:set(mesh:inode(dof,j-1), val)
      end
    end
  end

end
```

Note that the `nodal2d_indicator` function uses the fact that Lua, like Scheme, provides lexical closures and first class functions. The `nodal2d_indicator` does not itself construct a vector; rather, it produces another function to construct the vector. For example, in a problem involving the measurement of a cantilever beam of length l, we use the following call to define a sense function to measure the vertical displacement of the beam tip:

```
tip_displacement = nodal2d_indicator('uy', l, 0)
```

From the MATLAB interface, the analyst can then construct the vector using the command

```
sense_vector = Mesh_get_vector(mesh, 'tip_displacement');
```

If one were to refine the mesh or reorder the indices associated with different degrees of freedom, we could get a new version of the sense vector by repeating the `Mesh_get_vector` call. There is no need to redefine `tip_displacement`.

Global shape functions

The finite element method is based on two ideas: the Galerkin method, and the definition of approximation bases via element shape functions. However, sometimes we want an approximation based only on a subspace of the full finite element space. For example, we use Krylov subspaces to build a reduced model from a finite element model, and we use other subspaces to introduce constraints into a problem definition. A convenient way to define these subspaces is through the introduction of *global shape functions*, or linear combinations of nodal basis functions. In `HiQLab`, these functions are defined in terms of Lua callbacks in the same way that drive and sense functions are defined: the callback receives the mesh object and a vector assembler as arguments, and uses these to define the coefficients of a linear combination of nodal basis functions. For example, in an electrical problem in which some surface is a conductor at a constant (but not fixed) potential, we might define a global shape for the entire surface:

```
-- Add a variable for a conductor along [-w/2,w/2] x {h}.
-- The variable has units of voltage 'V', and dual units of charge 'Q'
conductor = mesh:add_global(
  function(x,y)
    if mesheq(y,h) and meshbetween(x, -w/2, w/2) then
      return 1
    end
  end,
  'V', 'Q')
```

By default, when a nodal basis function is used in the definition of a global shape, the degree of freedom corresponding to the original basis function is removed from the system. Therefore the above definition not only adds a new global degree of freedom corresponding to a unit voltage on the entire conductor surface; it also removes the original degrees of freedom describing the voltage across that surface. Now all the electrical degrees of freedom on the conductor surface are tied to a single value, and the corresponding dual variable is the total charge on the conductor. The global degree of freedom for the conductor can now be connected to the voltage at some node in a controlling circuit, so that the circuit components and the continuum finite element model are integrated together in the same simulation.

Inhomogeneous material parameters

Part of the specification of the perfectly matched layers described in Chapter 3 is the definition of a *stretching function*, which dictates how quickly the coordinate system is deformed into the complex domain. In `HiQLab`, the stretch function for a PML element is defined by registering a Lua callback with the element. For example, in a one dimensional example problem, we might specify a linear stretching function supported on $[a, b]$ like this:

```
pml_element:set_stretch(function(x)
  return max( (x-a)/(b-a) * stretch_max, 0 )
end)
```

We can then change the rate of stretching and the length of the perfectly matched layer by changing the values of `a`, `b`, and `stretch_max` in the Lua environment.

Using the same type of interface, it is possible to support inhomogeneous material parameters on an element-by-element basis. However, this feature is not yet implemented.

Type	Order	Form matrix		Form residual		Factor	Solve	Total time	
		Lua	C++	Lua	C++			Lua	C++
1.5 GHz Pentium 4									
Scalar	1	0.24	0.17	0.15	0.08	0.17	0.02	0.58	0.44
Scalar	2	0.29	0.26	0.10	0.06	0.19	0.03	0.61	0.54
Scalar	3	0.51	0.48	0.11	0.08	0.20	0.04	0.86	0.80
Elastic	1	0.51	0.46	0.22	0.15	0.91	0.08	1.72	1.60
Elastic	2	0.68	0.65	0.19	0.15	0.85	0.16	1.88	1.81
Elastic	3	1.08	1.05	0.23	0.20	0.87	0.22	2.40	2.34
1.67 GHz PowerPC G4									
Scalar	1	0.19	0.09	0.13	0.04	0.13	0.05	0.50	0.31
Scalar	2	0.18	0.13	0.08	0.03	0.14	0.04	0.44	0.34
Scalar	3	0.25	0.20	0.08	0.03	0.16	0.06	0.55	0.45
Elastic	1	0.39	0.29	0.15	0.05	0.76	0.13	1.43	1.23
Elastic	2	0.49	0.43	0.10	0.05	0.71	0.17	1.47	1.36
Elastic	3	0.75	0.71	0.10	0.06	0.76	0.35	1.96	1.88

Figure 6.1. `HiQLab` timings (in seconds) of different stages in solving small (49-by-73 node) forced vibration problems for scalar wave and elastic wave problems on a domain with a PML transformation. We test bilinear, biquadratic, and bicubic elements using either Lua or C++ to specify the transformation function. On a Mac G4 laptop, the Lua versions of the tests were slower than the C++ versions by at most 16% for elastic case and 60% on scalar case; on an Intel Pentium 4 desktop, these figures are 7% and 32%. For larger problems, the factorization takes a larger fraction of the overall time, further reducing the relative overhead of Lua callbacks compared to overall solution time.

6.5.2 Callback performance

A natural objection to the use of Lua callbacks in `HiQLab` is that it might be too slow. Even in pure C++ programs, the expense of small callbacks is of great enough concern that some authors recommend template techniques to allow the compiler to inline callback code at compile time rather than using a conventional C++ design using a virtual method dispatch [173]. However, cross-language control transfers between C++ and Lua are fast compared to similar control transfers between C++ and other interpreted languages (such as MATLAB); and Lua is fast enough that the interpretation overhead is small compared to the overall cost of the calculations.

Thus far, our experience with `HiQLab` suggests that the cost of using callbacks in problem description is small compared to the rest of the cost of forming and solving a finite element discretization.

To illustrate the cost of using Lua callbacks, we time a small model problem describing radiation of waves into a half space modeled by a perfectly mached layer. Our domain consists of a 73-by-49 node block terminated on three of the four sides with perfectly matched layers. On the fourth side, we prescribe natural boundary conditions, except over a few nodes in the center, where impose time-harmonic displacement boundary conditions. We solve both scalar wave and elastic wave versions of the problem using bilinear, biquadratic, and bicubic elements. As each element computes its local tangent or residual contributions, it invokes a callback function on each node in order to evaluate the stretch functions that define the PML. The `HiQLab` framework allows these callbacks to be implemented in C++ or in Lua, and we have written equivalent callbacks in each language. In Figure 6.1 we compare the costs of using the Lua callbacks versus C++ callbacks at each stage of the calculation. The solution procedure consists of four steps: assembling a global tangent matrix, assembling a global residual vector, factoring the stiffness matrix, and solving a linear system. The costs of building the tangent matrix and the residual vector are higher when using Lua than when using C++ for callbacks, but the time to factor and solve the linear system is the same in either case.

On a PowerBook laptop, the cost to form the tangent stiffness for a scalar wave equation discretized with linear elements is roughly doubled by using Lua instead of C++ for callbacks. The cost of computing the element stiffnesses is greater for higher-order elements or for elastic elements than it is in the scalar case with linear elements. Also, the cost to factor the tangent matrix is greater in the elastic case than in the scalar case, and the time to factor the matrix grows more quickly with increasing problem size than the time to form the matrix. Therefore, the overhead

of using Lua to specify PML transformations should be worse for a small scalar wave problem than for almost any other problem we have considered. In this worse case, the use of Lua adds about 60% to the overall solution time on a PowerBook laptop. For the same size mesh, Lua adds only about 4% to the cost of solving an elastic problem discretized with bicubic elements. On an Intel Pentium 4, the relative overhead of using Lua callbacks is even smaller. For the disk resonator simulations we describe in Chapter 7, the overhead of using Lua callbacks to specify perfectly matched layer transformations is negligible.

6.5.3 Slots and scales

`HiQLab` uses a standard structure for indexing nodal degrees of freedom. Each active degree of freedom is assigned an index, which is stored in the two-dimensional `id` array. The index of the ith degree of freedom for the node with global index j is stored in `id(i,j)`. Let us call the ith nodal degree of freedom the ith "slot." Because we wish to solve multiphysics problems, we need a way to track which field is represented by each slot; that is, we need a way to prevent some elements from thinking `id(0,j)` is the index for the x displacement of node j while other elements think it is the index for the electrical potential at the same node. `HiQLab` uses a Lua table called `var_slots` to keep map field names to indices; for instance, if the x displacement degree of freedom were assigned to the first slot, we would have `var_slots.ux = 0`. When new elements types are added to an existing mesh, they request the indices of slots corresponding to the fields they use, so that each element's local notion of the ordering of fields can be made consistent with the global order in which slots are assigned.

Assigning text names to the fields in a calculation simplifies the interface for specifying boundary conditions and drive and sense functions. It also makes it relatively

simple to manage a list of scales for different problem fields. In a Lua table called `dim_scales`, there is a mapping between different fields and the units used to express them. For example, if the first field corresponds to x displacement, then the first two entries in the `dim_scales.vars` table will be `'L'` and `'F'` to indicate that the x displacement has units of length, and the dual field has units of force. The fields `dim_scales.L` and `dim_scales.F`, in turn, specify characteristic length and force scales for the problem. If these scales are undefined, they default to one. The Lua support code provides cover routines which compute characteristic scales based on a characteristic length scale together with a table of material properties. For example, the routine to nondimensionalize mechanical problems chooses a characteristic time scale by dividing the characteristic length by the acoustic wave velocity in the specified material.

The scaling information stored in the `dim_scales` table is used to construct two scaling vectors stored inside the mesh object: one for the primary variables in the problem, and one for the secondary variables. The initialization of the scaling vectors is done using a Lua callback function, so that the user can change how scales are assigned if the default behavior is not appropriate. For example, in the case of a long, thin structure, the user might choose to assign different characteristic scales for the x and y displacement fields. Inside the analysis routines, `HiQLab` uses these scaling vectors to nondimensionalize the linearized problems in order to prevent artificial ill conditioning. Also, the nondimensionalized increment and residual norms are used for convergence tests in the Newton iteration routine to compute nonlinear equilibrium solutions.

Though the `HiQLab` system of mapping fields to slots and to scales is simple, we are unaware of other codes which provide the same capabilities. As with many other aspects of the code, the flexibility of the `HiQLab` system of assigning index slots and characteristic scales is based on the combination of a simple mechanism implemented

in the compiled core – in this case, element-to-global slot maps and the mesh scaling vectors – together with Lua cover functions to implement standard policies. Because the Lua code governs the policy of how scales and slots are assigned, it is possible for users to change the policy without recompiling the program.

6.6 The MATLAB interface

The MATLAB interface has two parts: a library of low-level routines for interacting with the C++ and Lua libraries, and a library of higher-level routines built by combining the low-level routines and the facilities of MATLAB. The low-level interfaces are built from an automatic wrapper generation tool based on Matwrap [91]. Except for helpers to pack results, fill in default arguments, and cope with differences between zero-based and one-based indexing, the interfaces to these routines are identical to their C++ counterparts. The high-level interfaces can be grouped into three main categories: administrative routines, numerical methods, and plots. We describe these now.

The `Mesh_load` routine creates a Lua interpreter and loads it with data from MATLAB; initializes the default paths used in Lua; executes a mesh generation script in the Lua interpreter; and initializes the resulting mesh object. The return values are a new mesh object and, optionally, a Lua interpreter object which can be used to call Lua methods or change the values of any variables in Lua that are used by callback functions. A typical call to `Mesh_load` might look like

```
param.length = 100e-6;  % Set the value of 'length' used in the mesh
[mesh,L] = Mesh_load('beammesh.lua', param);
```

The `param` argument to `Mesh_load` is the primary means used to parameterize the mesh construction from MATLAB. This argument is optional, however, and we consider it good practice to provide default parameter values in the Lua file. This can be

done very concisely using the Lua `or` operator, which returns the first non-`nil` value it sees. For example, in the `beammesh.lua` file, we might have

```
length = length or 50e-6
width  = width  or 2e-6
```

By providing default values in this way, one documents the parameters that control the mesh behavior and a set of reasonable defaults for those parameters.

Once the mesh has been loaded, one typically runs an analysis routine. `HiQLab` provides routines to compute nonlinear equilibrium states, time-harmonic linearized frequency responses, and free vibration modes. There are also routines to compute reduced-order models for faster frequency-response calculations. There are specialized modal analysis and model reduction routines for specific structures; in particular, we provide specialized routines for analysis of problems with perfectly-matched layers and thermoelastic damping, using both the algorithms described elsewhere in this dissertation and algorithms developed by T. Koyama [109, 108]. After running an analysis routine, we can manipulate the results with further scripts, or we can simply plot them. The `HiQLab` plot routines are unsophisticated, but we do include color plots of the solution fields, animations of time-harmonic deformations, and Bode plots of frequency-response behavior.

6.7 Conclusions

The `HiQLab` software architecture uses a clean but conventional organization of core modules to handle basic data structures, together with scripting language interfaces for scripting analyses and for problem description. The primary interface for scripting analyses is written in MATLAB, which allows us to quickly prototype new numerical algorithms; and the primary interface for mesh description is written in

Lua, which we chose for its speed, simplicity, and small size, as well as for the ease which we could embed it into the larger system.

The new features of `HiQLab` include the new algorithms and elements described elsewhere in this dissertation; and new architectural features based on callbacks to interpreted Lua functions, which define the mathematical functions that make up the problem description concisely and without direct reference to the details of the domain discretization. We use this callbacks for specifying boundary conditions, drive and sense functionals, global shape functions, and the stretching functions used to specify perfectly matched layers. Our architecture separates the responsibility for general purpose mechanisms for finite element problems, which are implemented in C++, from the domain-specific policies based on those mechanisms, which are written in Lua. For example, by implementing the storage of scaling vectors in the C++ core while constructing the vectors in Lua, we are able to provide a general-purpose framework for problem nondimensionalization, which the user can customize as part of the problem specification *without* the need to delve into the compiled core libraries.

Chapter 7

MEMS Examples

7.1 Introduction

In this chapter, we discuss two MEMS resonator simulations using `HiQLab`. We first examine in detail the behavior of a family of disk-shaped resonators. Our simulations show the effectiveness of the perfectly matched absorbing layer and the accuracy of our structured model reduction method. We also discuss a mode-interference phenomenon which substantially affects the performance of these devices. This phenomenon was first observed in our simulations, and then verified in subsequent laboratory experiments. Our second example is a checkerboard-shaped resonator; we use this example to show the effectiveness of the structure-preserving second-order Arnoldi algorithm. These examples are drawn from previously published papers [31, 30, 33]

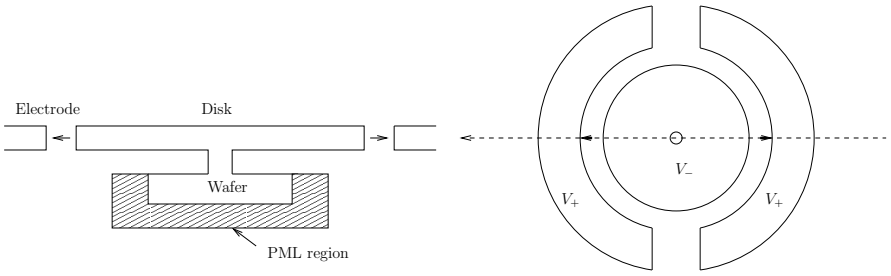

Figure 7.1. Schematic of a radial disk resonator. An overhead view (right) shows the arrangement of the resonating disk and the electrodes which force it. An idealized cross-section (left) is used in an axisymmetric simulation, where the wafer substrate is treated as semi-infinite using a perfectly-matched layer.

7.2 Study of disk resonators

We now examine in detail the response of several radial disk resonators which have been fabricated recently [176, 175, 31]. In the laboratory, these devices have shown quality factors as high as 55000 at frequencies of up to 1.14 GHz. Through numerical experiments, we explain in detail the mechanism of anchor loss in disk resonators; the predictions of our models are supported by recent experimental work [31], which we also partially report here.

A schematic of a disk resonator is shown in Figure 7.1. A thin disk is supported on a post above a substrate. The disk is surrounded by drive electrodes, and the potential difference between the disk and the drive electrodes pulls the disk radially outward at the rim. The disk is driven near the frequency of the first or second axisymmetric, bulk radial, in-plane mode. In [176], the disk is made of polysilicon; [175] describes both polysilicon and polydiamond disks; [31] is concerned with poly-SiGe disks.

In `HiQLab` simulations, the computed thermoelastic damping in these disk resonators is too small to be a dominant dissipation mechanism. Similarly, based on experiments in which Q changes little between vacuum-packed disk resonators and those operating in air, we do not believe air damping dominates. The disks are constructed of materials with low intrinsic loss, so that material loss should be small.

However, a major source of energy loss in these devices is the propagation of elastic waves down the supporting post and into the wafer below, where they are largely dissipated. Relative to the size of the resonating device, the wafer is large. We assume that the wafer is effectively infinite in extent, so that none of the waves that radiate into the substrate will be reflected. We use a perfectly matched layer to model the wafer as a semi-infinite half space. We will also assume axisymmetry in our simulations.

In the actual devices, there are nitride and oxide films between the device and the wafer. For our simulations we have ignored these geometric features as they typically only have a minor effect on resonator performance. Note that this is a simplification in our model, not an inherent limitation of the PML technology; indeed, one of the attractions of PMLs is the ability to handle layered media and other embedded scatterers.

In the examples to follow, we examine issues of

1. Mesh convergence – h and p.
2. Elucidation of the physical mechanism of anchor loss.
3. Design sensitivity in such resonators.
4. Performance of the model reduction method.

7.2.1 Convergence of Q

We first consider the polysilicon disk resonator described in [176], which has a disk 20 μm in diameter and 2 μm thick, supported 0.5 μm above the substrate by a post 2 μm in diameter. When the disk was driven in the second radial mode, the measured Q was 7330 in vacuum and 6100 in air, and the center frequency was

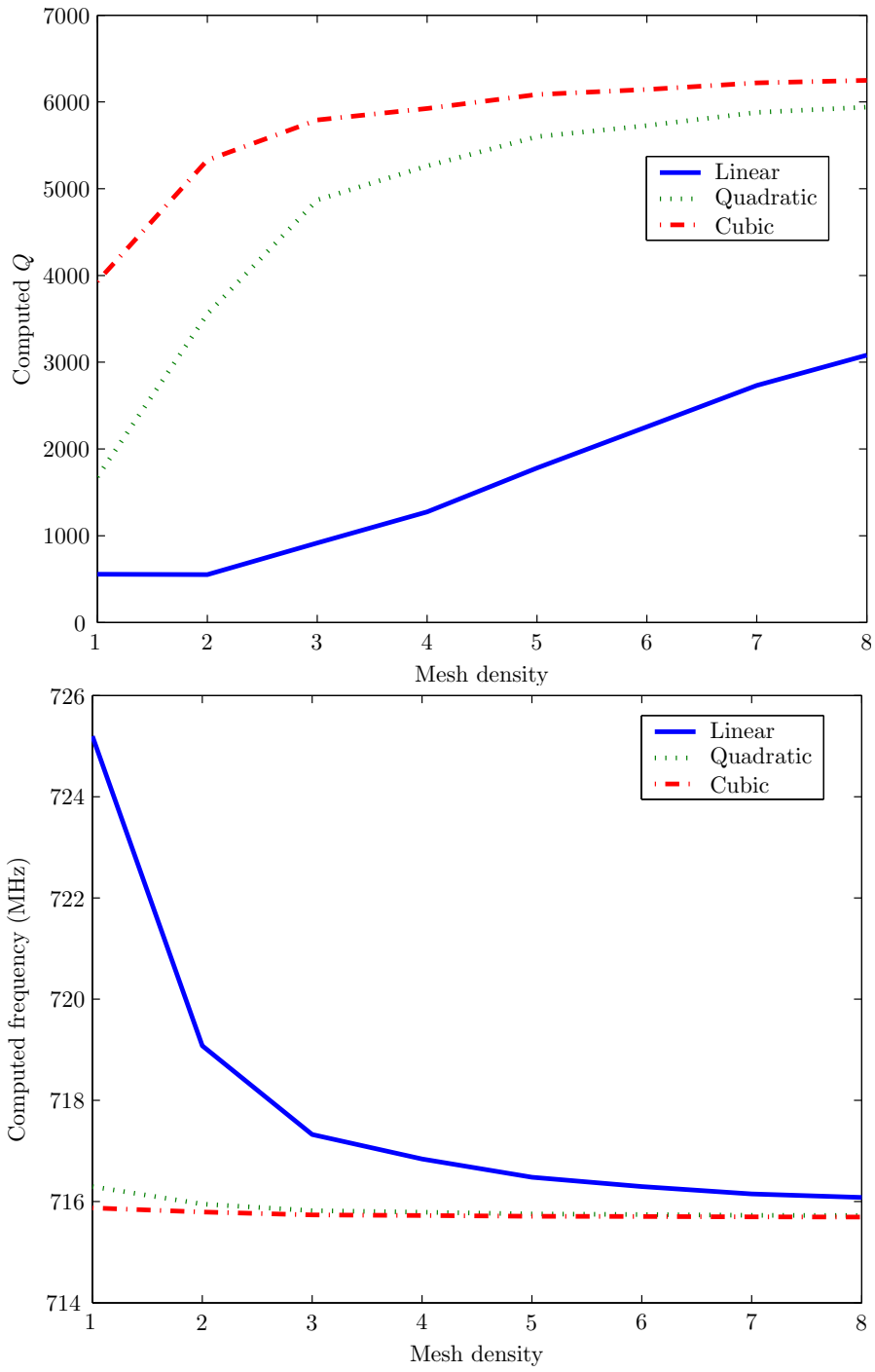

Figure 7.2. Convergence of Q and ω_{center} for the second radial mode of the polysilicon disk in [176].

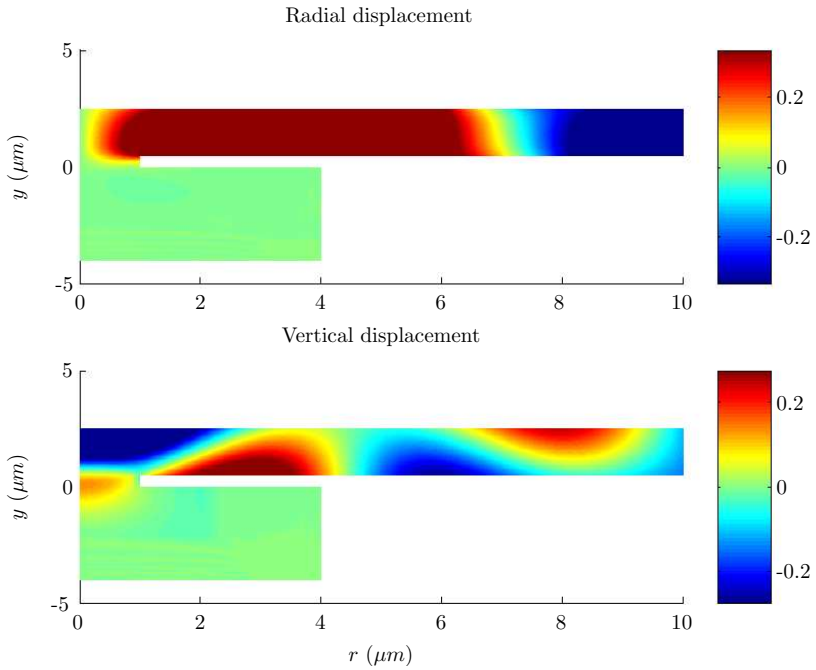

Figure 7.3. Forced displacement (real part) for the disk resonator model at 715 MHz.

733 MHz. In our best-resolved simulation, we computed a Q value of 6250 at a center frequency of 715.6 MHz. Perhaps surprisingly, relatively fine resolution was required to obtain convergence. When the mesh was under-resolved, the Q factor was drastically underestimated, possibly because the small flux reaching the anchor base could not be resolved by the mesh, and was therefore overestimated. We computed the value of Q using both linear and higher-order elements at several mesh densities; see Fig. 7.2. For a given mesh density parameter m, we chose elements so that nodes were as near as possible to $1/m$ μm apart. We computed Q and ω_{center} by using ARPACK in shift-and-invert mode [114] with an initial shift of 715 MHz to find the closest complex-valued eigenvalue; most of the time in these computations was spent in the sparse LU factorization of the shifted matrix. The plot shows a clear advantage to p-refinement for this class of problems.

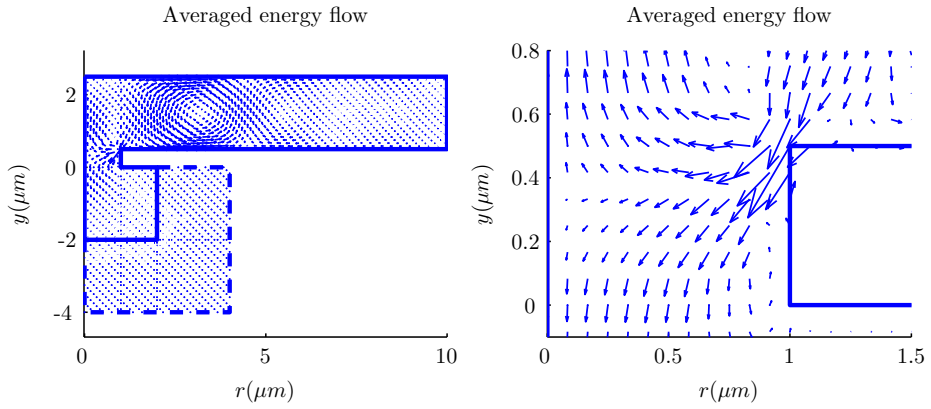

Figure 7.4. Time-averaged energy flux vector field in the disk resonator driven at 715 MHz. The left plot shows the full field; the right plot shows only the region in the vicinity of the post.

7.2.2 Observed energy loss mechanism

Figure 7.3 shows the behavior of the same disk when driven at 715 MHz, just slightly below the resonant frequency. The amplitudes of both the radial and vertical displacements are shown at the point when the forcing is maximal. Even though the design intent was to excite a pure radial mode, it is clear that the mode contains a bending component. This is a reflection of the fact that pure radial modes are not possible in supported structures. The majority of the displacement occurs in the disk itself, but there is some motion in the post as well. Though the forcing is in the radial direction, the Poisson effect leads to motion in the vertical direction at the center of the disk. This vertical "pump" motion results in displacement waves that travel down the post and into the substrate. In an animation, it is possible to see low-amplitude waves radiating away from the post to be absorbed into the perfectly matched layer.

To better understand the behavior shown in Figure 7.3, we compute the energy density flux:

$$F(t) = -\operatorname{Re}(\sigma e^{i\omega t})\operatorname{Re}(v e^{i\omega t}) \tag{7.1}$$

where σ is the stress tensor and v is the velocity vector. Since the energy flux changes over time, we time-average over a single period to obtain the mean energy density

flux:

$$\bar{F} = -\frac{1}{2}\,\mathrm{Re}(\sigma v^*) \tag{7.2}$$

where v^* is the complex conjugate of the velocity. For a standing wave, the displacement and stress are pure real, and $v = iku$ is pure imaginary, so there is no mean energy flux. Therefore, the mean energy flux field tells us something about the departure from the lossless standing-wave behavior. Figure 7.4 shows the mean energy flux for a region of the resonator near the edge of the post. The flux vectors in the body of the disk form cycles which carry energy around inside the disk, but do not let it escape into the substrate. Near the post, however, the cycle pattern is broken, and the energy flux plot shows a "spray" of energy that travels down the post and into the substrate.

7.2.3 Mode mixing and design sensitivity in the disk resonator

To further test our simulation technology on anchor loss, we now consider a series of five poly-$Si_{0.4}Ge_{0.6}$ disk resonators with 41.5 μm radii disks with different thicknesses [31]. Figure 7.5 shows one such 41.5 μm radius resonator. The disk itself is supported on a conical post with upper radius 1.49 μm, lower radius 1.61 μm, and nominal height 1 μm. The drive is clearly not fully axisymmetric but we model it as such for simplicity. For the material we use a density of 4127 kg/m^3 computed by linear interpolation and assume a Poisson ratio of 0.28; Young's modulus was estimated from an acoustic measurement as 139 GPa.

Figure 7.6 shows the computed in-phase radial and vertical displacements in one of the disks when it is driven with a radial forcing on its outer edge; the computed Q of the dominant mode is 140000. The radial motion is coupled to a small bending motion due to the stem as mentioned earlier. Thus, as in the prior example, the

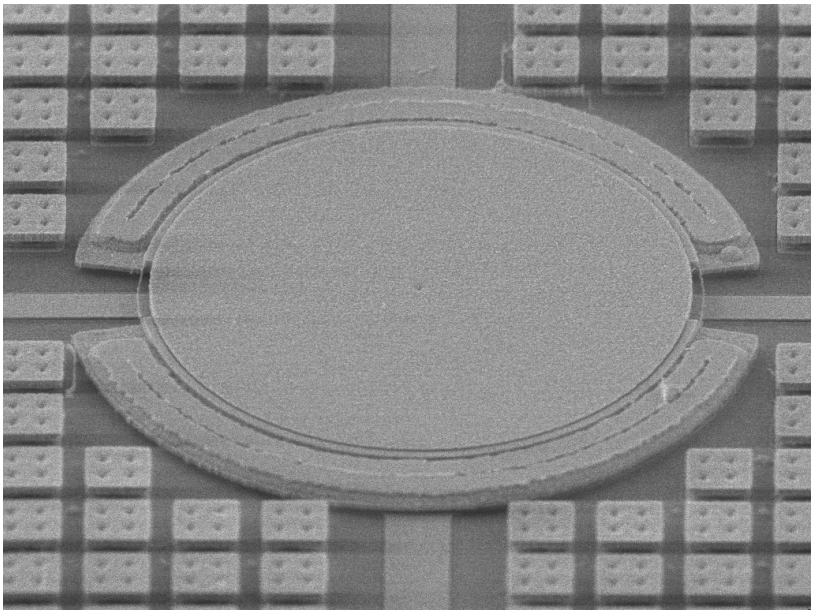

Figure 7.5. SEM of 41.5 μm radius poly-SiGe disk resonator.

dominant mode for this disk is not a pure radial motion. The bending motion of the mode along with the Poisson effect induces a vertical motion in the stem which pumps displacement waves into the substrate, where they carry away the energy of the resonance.

Figure 7.7 shows measured Q values from the five 41.5 μm disks. The error bars on thickness are indicative of the limits of the SEM geometry measurement method. Also shown in Figure 7.7 is simulation data using the measured geometry from the resonators. The two curves correspond to Q values for the two eigenvalues nearest the shift (45 MHz). The high Q curve is associated with the radial extension mode we wish to drive; the low Q mode is dominated by a bending motion. The agreement between the measured and computed Q values is good, and the trend with respect to changing disk thickness is captured well.

The presence of the nearby second mode has a large influence on the high Q mode's quality factor. This is seen in the large swings in the curves of Fig. 7.7 which are computed solely from the system eigenvalues. A good method of visualizing the pole

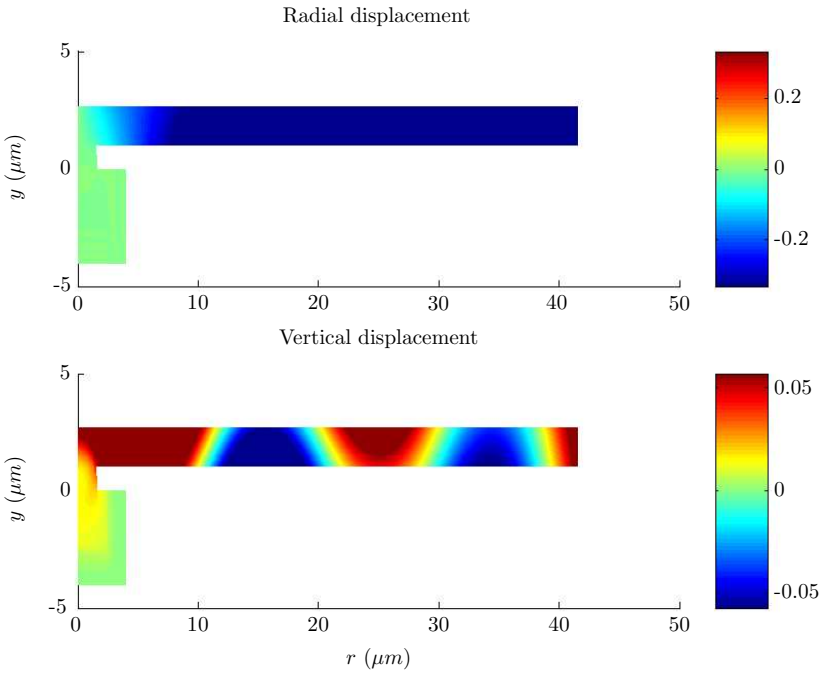

Figure 7.6. Radial and vertical displacement fields illustrating the mode mixing. Disk radius is 41.5 μm and film thickness is 1.6 μm. The drive frequency is 45 MHz and $Q = 140000$. Displacement contours are in units of μm.

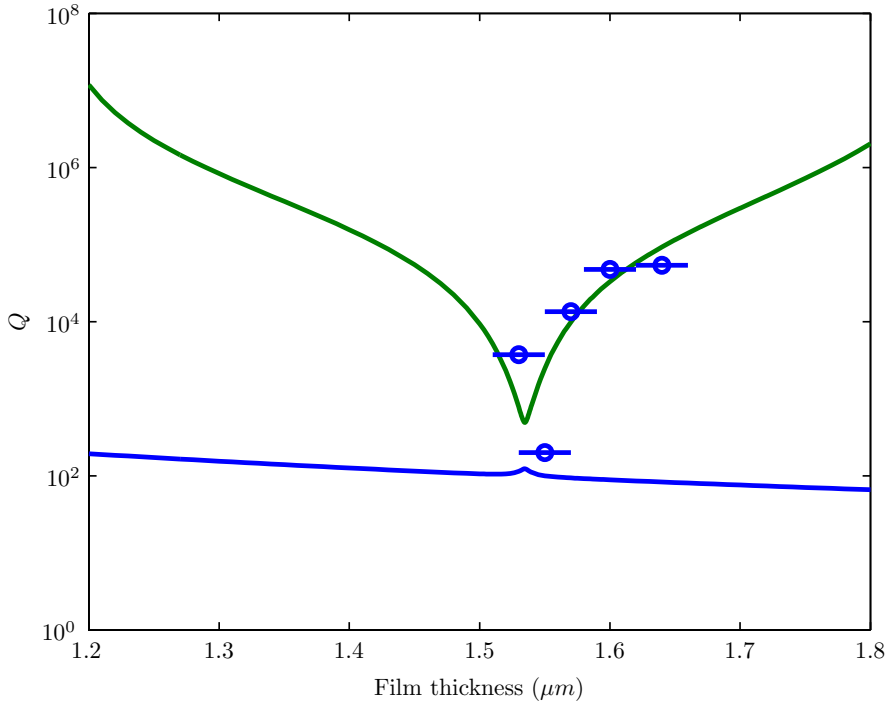

Figure 7.7. Measured and computed quality factors in 41.5 μm radius disk with varying film thickness. Upper curve indicates Q from eigenvalue closest to the shift. Lower curve indicates Q from next nearest eigenvalue.

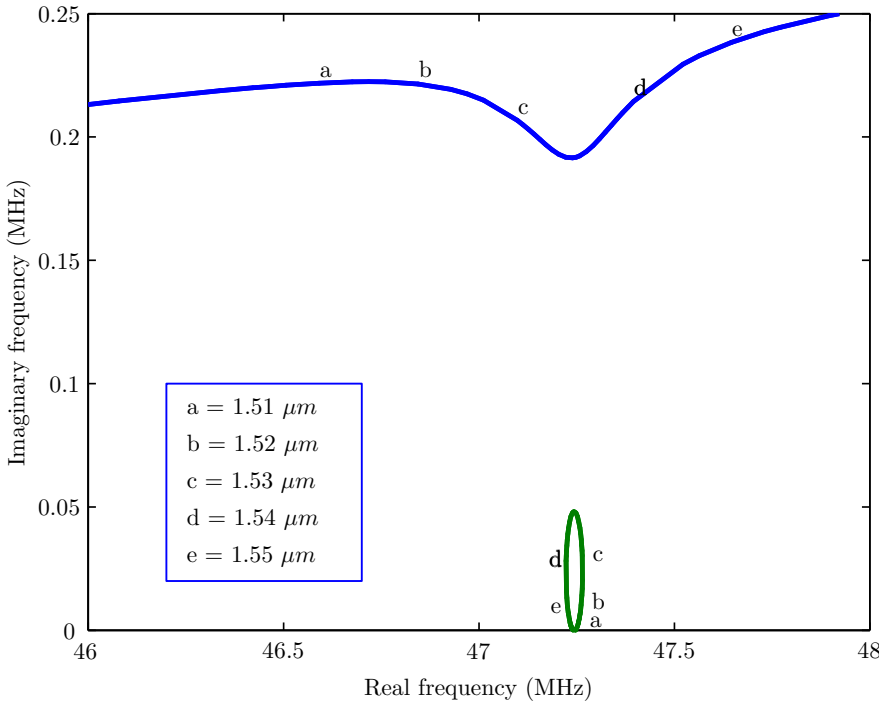

Figure 7.8. Root-locus plot parameterized by varying film thickness. Lower curve corresponds to the first mode and the upper curve to the next nearest mode.

interaction is to examine a root-locus diagram for the two interacting poles (eigenvalues) parameterized by film thickness. Figure 7.8 was computed for the 41.5 μm radius disks. As the thickness changes the two poles approach each other. The first mode's frequency first moves away from the real axis increasing damping, then back toward the real axis decreasing damping. The speed of the first pole increases the closer it is to the second pole in the complex plane. The dip in Q correlates well with the thickness at which the poles are closest.

7.2.4 Performance of model reduction method

As seen above, the bending-dominated mode significantly affects the resonant peaks associated with the radial dominated modes but not in a way that is immediately obvious. For this reason, Bode plots are helpful in understanding such systems. As an example of our model reduction technology, we start with a shift drawn from

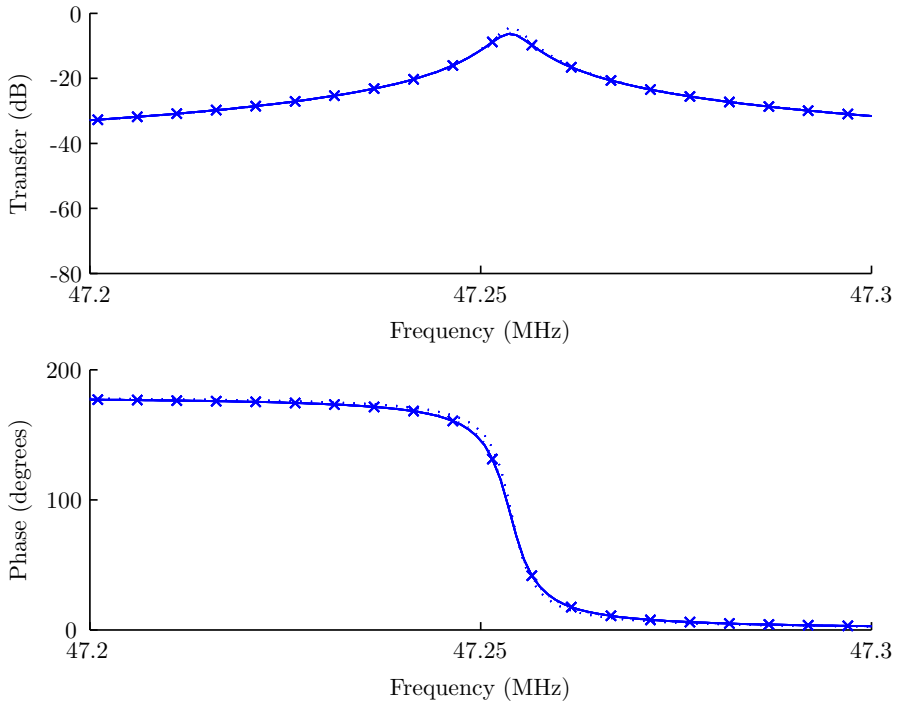

Figure 7.9. Bode plot of the disk resonator with the full model (x), a structure-preserving reduced model (solid line), and a standard Arnoldi reduced model (dashed line).

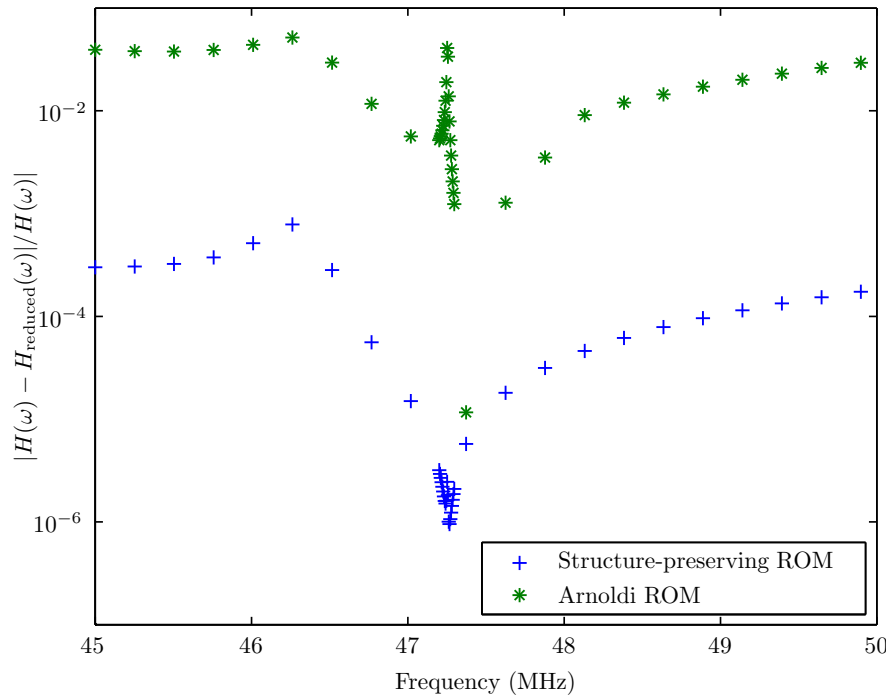

Figure 7.10. Errors in a structure-preserving reduced model (+) and a standard Arnoldi reduced model (*). Both reduced models are generated from the same number of Krylov vectors.

a hand estimate for the disk frequency:

$$\omega_{\text{shift}} = 2.405c/R, \tag{7.3}$$

where R is the disk radius and c is the compression-wave velocity in the disk material. With this shift, we require only two steps of shift-and-invert Arnoldi to resolve the two-dimensional invariant subspace for both relevant eigenvalues. We perform the projections in two ways: first, using a standard Arnoldi projection (with three complex vectors); and second, using the symmetry-preserving projection described in Chapter 3 that splits the real and imaginary parts of the Arnoldi vectors (with five real vectors). Both reduced models produce Bode plots which closely match the original 24265 degree of freedom system (Figure 7.9). In Figure 7.10, we compare the accuracy of the two models; clearly, the structure-preserving algorithm is more accurate.

7.2.5 Summary

Our motivation for studying PMLs was to investigate anchor loss in MEMS resonators. We have demonstrated the utility of our method by analyzing the behavior of a family of disk resonators. Our developed tools allow us, in detail, to describe the physical mechanism by which these resonators lose energy by radiation of elastic waves from the anchor. Because MEMS fabrication processes are not exact, we have analyzed the effect of variations in the thickness of the resonating disks. Our analysis showed that the quality factor of these resonators is highly sensitive to the film thickness, and experimental results confirmed this analysis. Our new tools allow us to explain the variations in the quality factor in terms of interactions between the desired radial extension mode and a parasitic bending mode which resonates at nearly the same frequency. The sensitivity to film thickness was first discovered numerically, and later was verified experimentally.

7.3 Checkerboard resonators and SOAR

As an application, we build a reduced-order model from a finite element simulation of a prototype MEMS filter [33]. The goal for this device is to produce a high-frequency bandpass filter to replace, for example, the surface acoustic wave (SAW) devices used in cell phones. The device (Figure 7.11) consists of a checkerboard of silicon squares which are linked at the corners [28]. The "checkers" are held at a fixed voltage bias relative to the drive and sense electrodes placed around the perimeter. A radio-frequency (RF) voltage variation on drive electrodes at the northwest corner creates an electrostatic force which causes the device to move in plane. The motion induces changes in capacitance at the sense electrodes at the southwest corner of the device. The induced motion is typically very small; the checker squares are two microns thick and tens of microns on a side, and the maximum displacement is on the order of tens of nanometers.

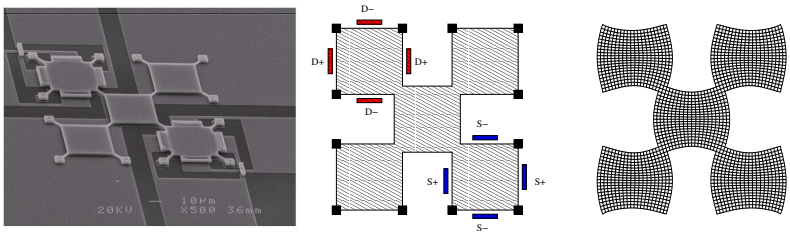

Figure 7.11. Illustration of a checkerboard resonator. The SEM picture (left) shows a fabricated device, and the simulation (right) shows one resonant mode excited during operation. The motion is excited at the northwest corner and sensed at the southeast corner (center).

A single square pinned at the corners exhibits a "Lamé mode." If the interaction between squares was negligible, the system would have a five-fold eigenvalue corresponding to independent Lamé-mode motions for each square. The coupling between the squares causes the five eigenvalues to split, so there are several poles near each other; consequently, the array has low motional resistance near the target frequency.

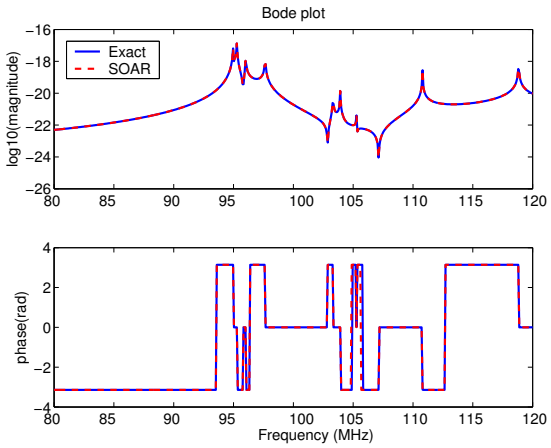

Figure 7.12. Bode plots from finite element model and reduced order model for a 3-by-3 checkerboard resonator

The same idea of using weakly coupled arrays has also been applied to other RF resonator designs [57].

We model the checkerboard with linear 2D plane stress elements and an empirical viscous damping model. Even for a small mesh ($N = 3231$), model reduction is beneficial. Using a reduced model with 150 degrees of freedom, we obtain a Bode plot which is visually indistinguishable from the plot for the unreduced system (Figure 7.12). We have also created a visualization tool which designers can use to see the forced motion at different frequencies (Figure 7.13). With a reduced model, we can compute the shape of the motion at a specified frequency within tens of milliseconds instead of seconds, quickly enough for the user to interactively scan through frequencies of interest. Designers can use these visualizations to build intuition about different strategies for constructing anchors, connecting resonator components, and placing drive and sense electrodes.

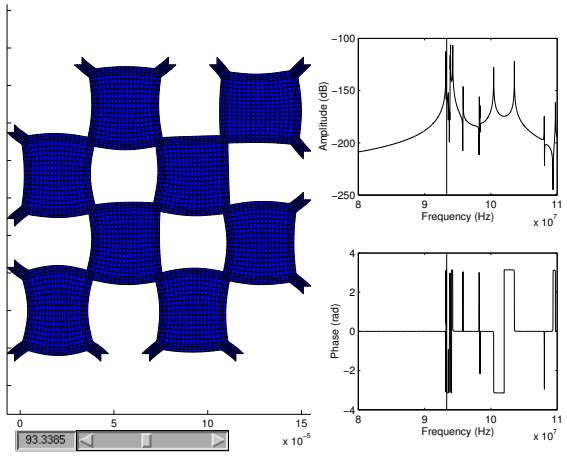

Figure 7.13. Screenshot of a visualization tool for observing forced response shapes for in-plane resonators. Using a reduced model, the tool can compute and plot the response shape quickly enough for interactive use

7.4 Conclusion

In this chapter, we have described simulations of two classes of resonant MEMS: a family of disk resonators, and a family of checkerboard-shaped resonators. For each resonator, we have constructed numerical simulations in `HiQLab`, and used structure-preserving model reduction methods to produce fast frequency-response simulations. In each case, the interactions between multiple resonant modes proved critical, so that naive projection onto a single mode is inadequate for exploring even the local frequency-response behavior. Our investigation of the disk resonator example led to the discovery of a previously unsuspected mode-interference effect, which has a major influence on device performance.

Bibliography

[1] ABAQUS, Inc. `http://www.abaqus.com/`.

[2] M. A. Abdelmoneum, M. U. Demirci, and C. T.-C. Nguyen. Stemless wine-glass-mode disk micromechanical resonators. In *Proceedings of Transducers 2003*, pages 698–701. IEEE, 2003.

[3] R. Abdolvand, G. K. Ho, A. Erbil, and F. Ayazi. Thermoelastic damping in trench-refilled polysilicon resonators. In *Proceedings of the 12th International Conference on Solid State Sensors, Actuators and Microsystems (Transducers 03)*, pages 324–327, Boston, June 2003.

[4] P.-A. Absil, R. Sepulchre, P. Van Dooren, and R. Mahony. Cubically convergent iterations for invariant subspace computation. *SIAM J. Matrix Anal. Appl.*, 26(1):70–96, 2004.

[5] R. Aigner, S. Marksteiner, L. Elbrecht, and W. Nessler. RF-filters in mobile phone applications. In *Proceedings of Transducers 03*, pages 891–894, Boston, June 2003.

[6] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[7] J.C. Andle and J.F. Vetelino. Acoustic wave biosensors. *Sensors and Actuators A*, 44(3):167–176, September 1994.

[8] ANSYS, Inc. `www.ansys.com`.

[9] ANSYS, Inc. Ansys multiphysics. `www.ansys.com/products/multiphysics.asp`.

[10] ANSYS, Inc. ANSYS 10.0 new features, June 2005. `www.ansys.com/products/newfeatures/default.asp`.

[11] A. C. Antoulas and D. C. Sorensen. Approximation of large-scale dynamical systems. Technical Report TR01-01.pdf, Department of Computational and Applied Mathematics, Rice University, 2001.

[12] A. C. Antoulas, D. C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. Technical Report TR00-38.pdf, Department of Computational and Applied Mathematics, Rice University, 2000.

[13] P. Arbenz and M. E. Hochstenbach. A Jacobi-Davidson method for solving complex symmetric eigenvalue problems. *SIAM J. Sci. Comp.*, 25:1655–1673, 2004.

[14] P. Ashwin, K. Böhmer, and Z. Mei. A numerical Liapunov-Schmidt method with applications to Hopf bifurcation on a square. *Math. Comp.*, 64:649–670, 1995.

[15] P. Ashwin and Z. Mei. A Hopf bifurcation with Robin boundary conditions. *J. Dynamics and Differential Equations*, 6:487–503, 1994.

[16] R. J. Astley. Infinite elements for wave problems: a review of current formulations and an assessment of accuracy. *International Journal for Numerical Methods in Engineering*, 49:951–976, 2000.

[17] I. Babuska and J. Osborn. Eigenvalue problems. In P.G. Ciarlet and J.L. Lions, editors, *Handbook of Numerical Analysis, Vol II: Finite Element Methods*, pages 643–787. North-Holland, 1991.

[18] Z. Bai. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied Numerical Mathematics*, 43:9–44, 2002.

[19] Z. Bai and Y. Su. Dimension reduction of second-order dynamical systems via a second-order Arnoldi method. *SIAM Journal of Scientific Computing*, 26(5):1692–1709, 2005.

[20] Satish Balay, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2001. http://www.mcs.anl.gov/petsc.

[21] M. Bao, H. Yang, Y. Sun, and P. J. French. Modified Reynolds' equation and analytical analysis of squeeze-film air damping of perforated structures. *Journal of Micromechanics and Microengineering*, 13:795–800, July 2003.

[22] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*. SIAM, Philadelphia, PA, 1994.

[23] U. Basu and A. Chopra. Perfectly matched layers for time-harmonic elastodynamics of unbounded domains: theory and finite-element implementation. *Computer Methods in Applied Mechanics and Engineering*, 192:1337–1375, 2003.

[24] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, 2000.

[25] J.-P. Bérenger. A perfectly matched layer for the absorption of electromagnetic waves. *Journal of Computational Physics*, 114:185–200, 1994.

[26] W.-J. Beyn, A. Champneys, E. J. Doedel, Yu. A. Kuznetsov, B. Sandstede, and W. Govaerts. Numerical continuation and computation of normal forms. In B. Fiedler, editor, *Handbook of Dynamical Systems III: Towards Applications*, chapter 4. Elsevier, 2001.

[27] W-J Beyn, W. Kleß, and V. Thümmler. Continuation of low-dimensional invariant subspaces in dynamical systems of large dimension. In B. Fiedler, editor, *Ergodic Theory, Analysis, and Efficient Simulation of Dynamical Systems*, pages 47–72. Springer, 2001.

[28] S. Bhave, D. Gao, R. Maboudian, and R. T. Howe. Fully differential poly-SiC Lamé mode resonator and checkerboard filter. In *Proceedings of MEMS 05*, Miami, FL, January 2005.

[29] D. Bindel. HiQLab project page.

[30] D. S. Bindel and S. Govindjee. Elastic PMLs for resonator anchor loss simulation. *International Journal for Numerical Methods in Engineering*, 64:789–818, 2005.

[31] D. S. Bindel, E. Quévy, T. Koyama, S. Govindjee, J. W. Demmel, and R. T. Howe. Anchor loss simulation in resonators. In *MEMS 2005, IEEE international conference*. IEEE, January 2005.

[32] David Bindel, James Demmel, and Mark Friedman. Continuation of invariant subspaces for large bifurcation problems. In *Proceedings of the SIAM Conference on Linear Algebra*, Williamsburg, VA, 2003.

[33] David S. Bindel, Zhaojun Bai, and James W. Demmel. Model reduction for RF MEMS simulation. In *Proceedings of PARA 04*, volume 3732/2006 of *Lecture Notes in Computer Science*. Springer, June 2004.

[34] David S. Bindel, James W. Demmel, and Mark Friedman. Continuation of invariant subspaces for large bifurcation problems. Technical Report UCB/EECS-2006-13, EECS Department, University of California, Berkeley, February 2006.

[35] M.A. Biot. Thermoelasticity and irreversible thermodynamics. *Journal of Applied Physics*, 27(3):240–253, March 1956.

[36] J. Bosec. *Continuation of Invariant Subspaces in Bifurcation Problems*. PhD thesis, University of Marburg, 2002.

[37] J. H. Brandts. The Riccati method for eigenvalues and invariant subspaces of matrices with inexpensive action. *Linear Algebra and its Applications*, 358(1):335–365, January 2003.

[38] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer, second edition, 2002.

[39] Jay Brotz. Damping in CMOS-MEMS resonators. Master's thesis, Carnegie Mellon University, Department of Electrical and Computer Engineering, June 2004.

[40] E. A. Burroughs, R. B. Lehoucq, L. A. Romero, and A. J. Salinger. Linear stability of flow in a differentially heated cavity via large-scale eigenvalue calculations. Technical Report SAND2002-3036J, Sandia National Laboratories, 2002.

[41] R. N. Candler, H. Li, M. Lutz, W.-T. Park, A. Partridge, G. Yama, and T. W. Kenny. Investigation of energy loss mechanisms in micromechanical resonators. In *Proceedings of the 12th International Conference on Solid State Sensors, Actuators and Microsystems (Transducers 03)*, pages 332–335, Boston, June 2003.

[42] P. Chadwick. Thermoelasticity, the dynamic theory. In I.N. Sneddon and R. Hill, editors, *Progress in solid mechanics*, volume 1, chapter 6, pages 265–330. North-Holland Publishing Company, 1960.

[43] Francoise Chatelin. *Spectral Approximation of Linear Operators*. Academic Press, 1983.

[44] C. S. Chien and M. H. Chen. Multiple bifurcations in a reaction-diffusion problem. *Computers Math. Applic.*, 35(8):15–39, 1998.

[45] C. S. Chien, Z. Mei, and C. L. Shen. Numerical continuation at double bifurcation points of a reaction-diffusion problem. *Int. J. Bifur. and Chaos*, 8(1):117–139, 1997.

[46] H. Cho, J. Kang, S. Kwak, K. Hwang, J. Min, J. Lee, D. Yoon, and T. Kim. Integration of PDMS microfluidic channel with silicon-based electromechanical cantilever sensor on a CD chip. In *MEMS 2005, IEEE international conference*. IEEE, January 2005.

[47] Y.-H. Cho, B. M. Kwak, A. P. Pisano, and R. T. Howe. Viscous damping model for laterally oscillating microstructures. *IEEE/ASME Journal of Microelectromechanical Systems*, 3:81–87, 1994.

[48] R.G. Christian. The theory of oscillating-vane vacuum gauges. *Vacuum*, 16:175–178, 1966.

[49] Phillippe G. Ciarlet. *The Finite Element Method for Elliptic Problems*, volume 40 of *Classics in Applied Mathematics*. SIAM, 2002.

[50] K. A. Cliffe, A. Spence, and S. J. Tavener. The numerical analysis of bifurcation problems with application to fluid mechanics. In A. Iserles, editor, *Acta Numerica*, volume 9, chapter 2, pages 39–132. Cambridge University Press, 2000.

[51] F. Collino and P. Monk. Optimizing the perfectly matched layer. *Computer Methods in Applied Mechanics and Engineering*, 164:157–171, 1998.

[52] F. Collino and P. Monk. The perfectly matched layer in curvilinear coordinates. *SIAM Journal of Scientific Computing*, 19:2061–2090, 1998.

[53] F. Collino and C. Tsogka. Application of the perfectly matched absorbing layer model to the linear elastodynamic problem in anisotropic heterogeneous media. *Geophysics*, 66:294–307, 2001.

[54] M.C. Cross and R. Lifshitz. Elastic wave transmission at an abrupt junction in a thin plate with application to heat transport and vibrations in mesoscopic systems. *Physical Review B*, 64, 2001.

[55] G. Dangelmayr. Degenerate bifurcations near a double eigenvalue in the Brusselator. *J. Austral. Math. Soc. Ser. B*, 28:486–535, 1987.

[56] T. A. Davis. Algorithm 832: UMFPACK — an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30:196–199, 2004.

[57] Mustafa U. Demirci, Mohamed A. Abdelmoneum, and Clark T.-C. Nguyen. Mechanically corner-coupled square microresonator array for reduced series motional resistance. In *Proc. of the 12th Intern. Conf. on Solid State Sensors, Actuators, and Microsystems*, pages 955–958, Boston, June 2003.

[58] J. W. Demmel. Three methods for refining estimates of invariant subspaces. *Computing*, 38:43–57, 1987.

[59] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, 1997.

[60] J. W. Demmel, L. Dieci, and M. J. Friedman. Computing connecting orbits via an improved algorithm for continuing invariant subspaces. *SIAM J. Sci. Comp.*, 22(1):81–94, 2001.

[61] A. Dhooge, W. Govaerts, Yu.A. Kuznetsov, W. Mestrom, and A. M. Riet. MATLAB continuation software package CL_MATCONT, January 2003. `http://www.math.uu.nl/people/kuznet/cm/`.

[62] L. Dieci and T. Eirola. On smooth decompositions of matrices. *SIAM J. Matrix Anal. Appl.*, 20(3):800–819, 1999.

[63] L. Dieci and M. J. Friedman. Continuation of invariant subspaces. *Numerical Linear Algebra Applications*, 8:317–327, 2001.

[64] L. Dieci and A. Papini. Point-to-periodic and periodic-to-periodic connections. Preprint, 2003.

[65] L. Dieci and A. J. Rebaza. Continuation of eigendecompositions. *To appear in FGCS: Future Generation Computer Systems – Computational Science*, 2002.

[66] E. J. Doedel and H. Sharifi. Collocation methods for continuation problems in nonlinear elliptic PDEs, issue on continuation. In D. Henry and A. Bergeon, editors, *Continuation Methods in Fluid Mechanics*, volume 74 of *Notes on Numerical Fluid Mechanics*, pages 105–118. Vieweg, 2000.

[67] A. Duwel, J. Gorman, M. Weinstein, J. Borenstein, and P. Ward. Experimental study of thermoelastic damping in MEMS gyros. *Sensors and Actuators A*, 103:70–75, 2003.

[68] EDA industry working groups. `www.eda-stds.org`. Includes pointers to references and working groups concerned with Verilog and VHDL.

[69] A. Edelman, T. Arias, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl.*, 20(2):303–353, 1998.

[70] B. Engquist and A. Majda. Absorbing boundary conditions for the numerical simulation of waves. *Mathematics of Computation*, 31:629–651, 1977.

[71] Gary K. Fedder and Qi Jing. NODAS 1.3 – nodal design of actuators and sensors. In *Proceedings of BMAS 98: IEEE/VIUF Int. Workshop on Behavioral Modeling and Simulation*, October 1998.

[72] R. Freund. Krylov-subspace methods for reduced-order modeling in circuit simulation. *Journal of Computational and Applied Mathematics*, 123:395–421, 2000.

[73] M. J. Friedman. Improved detection of bifurcations in large nonlinear systems via the Continuation of Invariant Subspaces algorithm. *Int. J. Bif. and Chaos*, 11(8):2277–2285, 2001.

[74] M. J. Friedman and M. E. Jackson. An improved RLV stability analysis via a continuation approach. Technical report, NASA Marshall Space Flight Center, 2002.

[75] J. Fuhrmann, Th. Koprucki, and H. Langmach. pdelib: an open modular tool box for the numerical solution of partial differential equations. design patterns. In W. Hackbusch and G. Wittum, editors, *Proceedings of the 14th GAMM Seminar Kiel on Concepts of Numerical Software*, January 1998.

[76] M.A. Gallis and J.R. Torczynski. An improved Reynolds-equation model for gas damping of microbeam motion. *Journal of Microelectromechanical Systems*, 13(4):653–658, August 2004.

[77] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[78] K. Georg. Matrix-free numerical continuation and bifurcation. *Numerical Functional Analysis and Optimization*, 22:303–320, 2001.

[79] R. Geus and P. Arbenz. PySparse and PyFemax: a Python framework for large scale sparse linear algebra. In *PyCon03*, Washington, DC, March 2003.

[80] D. Givoli. *Numerical Methods for Problems in Infinite Domains*. Elsevier, 1992.

[81] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The John Hopkins University Press, 1989.

[82] M. Golubitsky and D. G. Schaeffer. *Singularities and groups in bifurcation theory, Vol 1*. Springer-Verlag, 1985.

[83] J. P. Gorman. Finite element model of thermoelastic damping in MEMS. Master's thesis, Massachusetts Institute of Technology, June 2002.

[84] W. Govaerts. Computation of singularities in large nonlinear systems. *SIAM J. Numerical Anal.*, 34:867–880, 1997.

[85] W. Govaerts. *Numerical methods for bifurcations of dynamical equilibria*. SIAM Publications, Philadelphia, 2000.

[86] W. Govaerts, J. Guckenheimer, and A. Khibnik. Defining functions for multiple Hopf bifurcations. *SIAM J. Numerical Anal.*, 34:1269–1288, 1997.

[87] Z. Hao and F. Ayazi. Support loss in micromechanical disk resonators. In *Proceedings of MEMS 05*, pages 137–141. IEEE, 2005.

[88] Z. Hao, A. Erbil, and F. Ayazi. An analytical model for support loss in micromachined beam resonators with in-plane flexural vibration. *Sensors and Actuators A*, 109:156–164, 2003.

[89] I. Harari, M. Slavutin, and E. Turkel. Analytical and numerical studies of a finite element PML for the Helmholtz equation. *Journal of Computational Acoustics*, 8:121–137, 2000.

[90] Michael A. Heroux, Roscoe A. Bartlett, Vicki E. Howle, Robert J. Hoekstra, Jonathan J. Hu, Tamara G. Kolda, Richard B. Lehoucq, Kevin R. Long, Roger P. Pawlowski, Eric T. Phipps, Andrew G. Salinger, Heidi K. Thornquist, Ray S. Tuminaro, James M. Willenbring, Alan Williams, and Kendall S. Stanley. An overview of the Trilinos project. *ACM Trans. Math. Softw.*, 31(3):397–423, 2005.

[91] Gary Holt. Matwrap. `lnc.usc.edu/~holt/matwrap/`.

[92] D. Homentcovschi and R.N. Miles. Viscous damping of perforated planar micromechanical structures. *Sensors and Actuators A*, 119:544–552, 2005.

[93] H. Hosaka, K. Itao, and S. Kuroda. Damping characteristics of beam-shaped micro-oscillators. *Sensors and Actuators A*, 49:87–95, 1995.

[94] M. Hoummady and D. Hauden. Acoustic wave thermal sensitivity: Temperature sensors and temperature compensation in microsensors. *Sensors and Actuators A*, 44(3):177–182, September 1994.

[95] B. H. Houston, D. M. Photiadis, M. H. Marcus, J. A. Bucaro, Xiao Liu, and J. F. Vignola. Thermoelastic loss in microscale oscillators. *Applied Physics Letters*, 80:1300–1302, 2002.

[96] James S. Howland. The Livsic matrix in perturbation theory. *Journal of Mathematical Analysis and Applications*, 50:415–437, 1975.

[97] X.M.H. Huang, X.L. Feng, C.A. Zorman, M. Mehregany, and M.L. Roukes. VHF, UHF, and microwave frequency nanomechanical resonators. *New Journal of Physics*, 7, 2005. `http://www.njp.org`.

[98] T. J. R. Hughes. *The Finite Element Method*. Dover, 2000.

[99] R. Ierusalimschy, L. H. de Figueiredo, and W. Celes. *Lua 5.0 Reference Manual*, November 2003.

[100] Roberto Ierusalimschy. *Programming in Lua*. Ingram and Baker & Taylor, 2003.

[101] Roberto Ierusalimschy, Luiz Henrique de Figueiredo, and Waldemar Celes. The evolution of Lua. In *ACM History of Programming Languages III*. To appear.

[102] Conventor Inc. `http://www.conventor.com/coventorware`.

[103] IntelliSense Software Corp., 2005. `http://intellisensesoftware.com/`.

[104] S.G. Joshi. Flow sensors based on surface acoustic waves. *Sensors and Actuators A*, 44(3):191–197, September 1994.

[105] Z. Kádár, W. Kindt, A. Bossche, and J. Mollinger. Quality factor of torsional resonators in the low-pressure region. *Sensors and Actuators A*, pages 299–303, 1996.

[106] T. Kato. *Perturbation Theory for Linear Operators*. Springer-Verlag, corrected printing of the second edition edition, 1995.

[107] G. T. A. Kovacs. *Micromachined transducers sourcebook*. McGraw-Kill, 1998.

[108] Tsuyoshi Koyama, David S. Bindel, and Sanjay Govindjee. Reduced order modeling for simulation of TED in MEMS resonators. Manuscript in preparation., June 2006.

[109] Tsuyoshi Koyama, David S. Bindel, Wei He, Emmanuel Quevy, James W. Demmel, Sanjay Govindjee, and Roger T. Howe. Simulation tools for damping in high frequency resonators. In *12th International Conference on Solid-State Sensors, Actuators, and Microsystems (Transducers '03)*, November 2005.

[110] Yu. A. Kuznetsov. *Elements of Applied Bifurcation Theory, Second edition.* Springer-Verlag, New York, 1998.

[111] P.Y. Kwok, M.S. Weinberg, and K.S. Breuer. Fluid effects in vibrating micromachined structures. *Journal of Microelectromechanical Systems*, 14(4):770–780, August 2005.

[112] T. H. Lee. *The Design of CMOS Radio-Frequency Integrated Circuits.* Cambridge University Press, 2nd edition, 2004.

[113] R. Lefever and I. Prigogine. Symmetry-breaking instabilities in dissipative systems II. *J. Chem. Phys.*, 48:1695–1700, 1968.

[114] R. B. Lehoucq, D. C. Soensen, and C. Yang. *ARPACK User's Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods.* SIAM Publications, Philadelphia, 1998.

[115] Gary Li and Henry Hughes. Review of viscous damping in micro-machined structures. In Eric Peeters and Oliver Paul, editors, *Micromachined Devices and Components VI*, volume 4176 of *Proceedings of SPIE*, pages 30–46. SPIE, 2000.

[116] Jing-Rebecca Li. *Model Reduction of Large Linear Systems via Low Rank System Gramians.* PhD thesis, Massachusetts Institute of Technology, 2000.

[117] R.-C. Li and Z. Bai. Structure-preserving model reduction using a Krylov subspace formulation. *Comm. Math. Sci.*, 3:179–199, 2005.

[118] R. Lifshitz and M. L. Roukes. Thermoelastic damping in micro- and nanomechanical systems. *Physical Review B*, 61:5600–5609, 2000.

[119] R.M. Lin and W.J. Wang. Structural dynamics of microsystems – current state of research and future directions. *Mechanical systems and signal processing*, 20:1015–1043, 2006.

[120] Ariel Manzur. tolua++. `www.codenix.com/~tolua`, 2006.

[121] S. J. Martin, G. C. Frye, and K. O. Wessendorf. Sensing liquid properties with thickness-shear mode resonators. *Sensors and Actuators A*, 44(3):209–218, September 1994.

[122] W. Mason. *Physical Acoustics and the Properties of Solids.* D. Van Nostrand Company, Inc., 1958.

[123] F. McKenna. *Object-oriented finite element programming: frameworks for analysis, algorithms, and parallel computing*. PhD thesis, University of California, Berkeley, 1997.

[124] Z. Mei. *Numerical bifurcation analysis for reaction-diffusion equations*. PhD thesis, University of Marburg, 1997.

[125] MEMSCAP, Inc. MEMSCAP CAD Solutions. `http://www.memscap.com/products-cad.html`.

[126] R.E. Mihailovich and N.C. MacDonald. Dissipation measurements of vacuum-operated single-crystal silicon microresonators. *Sensors and Actuators A*, 50:199–207, 1995.

[127] T. Mukherjee, G. Fedder, and J. White. Emerging simulation approaches for micromachined devices. *IEEE Transactions on Computer Aided Design*, December 2000.

[128] R. Mullen and T. Belytschko. Dispersion analysis of finite element semidiscretizations of the two-dimensional wave equation. *International Journal for Numerical Methods in Engineering*, 18:11–29, 1982.

[129] COMSOL Multiphysics. `http://www.comsol.com/`.

[130] A. Nathan and H. Baltes. *Microtransducer CAD - Physical and Computational Aspects*. Springer-Verlag, 1999.

[131] William E. Newell. Miniaturization of tuning forks. *Science*, 161(3848):1320–1326, September 1968.

[132] Roger G. Newton. *Scattering Theory of Waves and Particles*. Dover Publications, second edition, 2002.

[133] C. T.-C. Nguyen. Transceiver front-end architectures using vibrating micromechanical signal processors. In *Dig. of Papers, Topical Meeting on Silicon Monolithic Integrated Circuits in RF Systems*, pages 23–32, 2001.

[134] C. T.-C. Nguyen. Vibrating RF MEMS for low power wireless communications. In *Proceedings of the 2001 International MEMS Workshop (iMEMS01)*, pages 21–34, Singapore, July 2001.

[135] A. N. Norris and D. M. Photiadis. Thermoelastic relaxation in elastic structures with applications to thin plates. arXiv:cond-mat/0405323 v2, November 2004.

[136] W. Nowacki. *Dynamic problems of thermoelasticity*. Noordhoff International Publishing, 1975. Translated by Henryk Worski from the 1966 Polish edition.

[137] Open system for earthquake engineering simulation (OpenSees). `opensees.berkeley.edu`.

[138] Y.-H. Park and K. C. Park. High-fidelity modeling of MEMS resonators–part I: Anchor loss mechanisms through substrate. *Journal of Microelectromechanical Systems*, 13:238–247, 2004.

[139] Y.-H. Park and K. C. Park. High-fidelity modeling of MEMS resonators–part II: Coupled beam-substrate dynamics and validation. *Journal of Microelectromechanical Systems*, 13:248–257, 2004.

[140] B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM, 1997.

[141] Edwin S. Petrus. SKILL: a Lisp based extension language. In *LUV 93: Proceedings of the third international conference on Lisp users and vendors*, pages 71–79, New York, NY, USA, 1993. ACM Press.

[142] Python programming language – official website. `www.python.org`.

[143] R. H. Randall, F. C. Rose, and C. Zener. Intercrystalline thermal currents as a source of internal friction. *Physical Review*, 56:343–349, August 1939.

[144] Michael Reed and Barry Simon. *Methods of Mathematical Physics IV: Analysis of Operators*. Academic Press, 1978.

[145] T. V. Roszhart. The effect of thermoelastic internal friction on the Q of micromachined silicon resonators. In *Proceedings of the Solid State Sensors and Actuators Workshop*, pages 13–16, Hilton Head Island, SC, 1990. IEEE.

[146] Yousef Saad. *Numerical Methods for Large Eigenvalue Problems*. Manchester University Press, 1992. Available at `www-users.cs.umn.edu/~saad/books.html`.

[147] D. Schaeffer and M. Golubitsky. Bifurcation analysis near a double eigenvalue of a model chemical reaction. *Arch. Rational Mech. Anal.*, 75:315–347, 1981.

[148] S. Senturia. *Microsystem Design*. Kluwer Academic Publishers, 2001.

[149] S.D. Senturia. Simulation and design of microsystems: a 10-year perspective. *Sensors and Actuators A*, 67:1–7, 1998.

[150] D. Sherman. An investigation of MEMS anchor design for optimal stiffness and damping. Technical report, University of California, Department of Mechanical Engineering, 1996. (M.S. Report).

[151] G. M. Shroff and H. B. Keller. Stabilization of unstable procedures: The recursive projection method. *SIAM J. Numerical Anal.*, 30:1099–1120, 1993.

[152] D. C. Sorensen and A. C. Antoulas. Projection methods for balanced model reduction. Technical Report TR01-03.pdf, Department of Computational and Applied Mathematics, Rice University, 2001.

[153] V. T. Srikar and Stephen D. Senturia. Thermoelastic damping in fine-grained polysilicon flexural beam resonators. *Journal of Microelectromechanical Systems*, 11:499–504, 2002.

[154] J. B. Starr. Squeeze film damping in solid-state accelerometers. In *Proceedings of the IEEE Solid-State Sensors and Actuators Workshop*, pages 44–47, Hilton Head Island, SC, June 1990.

[155] G oran Stemme. Resonant silicon sensors. *J. Micromech. Microeng.*, 1991.

[156] G. W. Stewart. Error and perturbation bounds for subspaces associated with certain eigenvalue problems. *SIAM Review*, 4:727–764, 1973.

[157] G. W. Stewart. *Matrix Algorithms, Volume II: Eigensystems*. SIAM, 2001.

[158] G. W. Stewart and Ji guang Sun. *Matrix Perturbation Theory*. Academic Press, San Diego, CA, 1990.

[159] G. Strang and G. Fix. *An Analysis of the Finite Element Method*. Wellesley-Cambridge Press, 1988.

[160] SUGAR group. SUGAR: A MEMS simulation tool. `http://mems.sourceforge.net`.

[161] R. L. Taylor. FEAP. `http://www.ce.berkeley.edu/~rlt/feap`, 2004.

[162] F. Teixeira and W. Chew. Complex space approach to perfectly matched layers: a review and some new developments. *International Journal of Numerical Modelling*, 13:441–455, 2000.

[163] The MathWorks, Inc. `www.mathworks.com`.

[164] H. Tilmans, M. Elwenspoek, and J.H.J. Fluitman. Micro-resonant force gauges. *Sensors and Actuators A*, 30:35–53, 1992.

[165] L. N. Trefethen. Pseudospectra of matrices. In D. F. Griffiths and G. A. Watson, editors, *Numerical Analysis 1991*, pages 234–262. Longman Scientific and Technical, Harlow, Essex, UK, 1991.

[166] L. N. Trefethen and M. Embree. *Spectra and Pseudospectra*. Princeton University Press, 2005.

[167] E. Turkel. Introduction to the special issue on absorbing boundary conditions. *Applied Numerical Mathematics*, 27:327–329, 1998.

[168] E. Turkel and A. Yefet. Absorbing PML boundary layers for wave-like equations. *Applied Numerical Mathematics*, 27:533–557, 1998.

[169] J. M. Varah. On the separation of two matrices. *SIAM J. Numerical Anal.*, pages 212–222, 1979.

[170] T. Veijola, H. Kuisma, and J. Lahdenperä. Model for gas film damping in a silicon accelerometer. In *Proceedings of Transducers 97*, pages 1097–1100, Chicago, June 1997. IEEE.

[171] T. Veijola and T. Mattila. Compact squeezed-film damping model for perforated surface. In *Proceedings of Transducers 01*, pages 1506–1509. IEEE, June 2001.

[172] T. Veijola and M. Turoswki. Compact damping models for laterally moving microstructures with gas-rarefaction effects. *Journal of Microelectromechanical Systems*, 10(2):263–273, June 2001.

[173] Todd Veldhuizen. Techniques for scientific C++. Technical Report 542, Indiana University Computer Science Department, August 2000.

[174] Adrian Venema. Preface to a special issue on acoustic-wave-based microsensors. *Sensors and Actuators A*, 44(3):165, September 1994.

[175] J. Wang, J. E. Butler, T. Feygelson, and C. T.-C. Nguyen. 1.51-GHz nanocrystalline diamond micromechanical disk resonator with material-mismatched isolating support. In *Proceedings of MEMS 2004*, pages 641–644, Boston, 2004.

[176] J. Wang, Z. Ren, and C. T-C. Nguyen. Self-aligned 1.14 GHz vibrating radial-mode disk resonators. In *12th International Conference on Solid-State Sensors, Actuators, and Microsystems (Transducers '03)*, pages 947–950, 2003.

[177] K. Wang, Y. Yu, A.-C. Wong, and C. T.-C. Nguyen. VHF free-free beam high-Q micromechanical resonators. In *Technical Digest, 12th International IEEE Micro Electro Mechanical Systems Conference*, pages 453–458, 1999.

[178] X. Wang, M. Judy, and J. White. Validating fast simulation of air damping in micromachined devices. In *Proceedings of TRANSDUCERS 02*, pages 210–213. IEEE, 2002.

[179] Brent B. Welch. *Practical Programming in Tcl and Tk*. Prentice Hall, second edition, 1997.

[180] J. Yang, T. Ono, and M. Esashi. Mechanical behavior of ultrathin microcantilever. *Sensors and Actuators A*, 82:102–107, 2000.

[181] J. Yang, T. Ono, and M. Esashi. Energy dissipation in submicrometer thick single-crystal silicon cantilevers. *Journal of Micromechanical Systems*, 11:775–783, 2002.

[182] Y.-J. Yang and P.-C. Yen. An efficient macromodeling methodology for air damping effects. *Journal of Microelectromechanical Systems*, 14(4):812–827, August 2005.

[183] K. Y. Yasumura, T. D. Stowe, E. M. Chow, T. Pfafman, T. W. Kenny, B. C. Stipe, and D. Rugar. Quality factors in micron- and submicron-thick cantilevers. *Journal of Microelectromechanical Systems*, 9:117–125, 2000.

[184] W. Ye and S. Hutcherson. Air damping of microbeam resonators in a low vacuum. In *Proceedings of the 12th International Conference on Solid-State Sensors, Actuators and Microsystems*, pages 772–775. IEEE, June 2005.

[185] W. Ye, X. Wang, W. Hemmert, D. Freeman, and J. White. Air damping in laterally oscillating microresonators: A numerical and experimental study. *Journal of Microelectromechanical Systems*, 12:557–566, 2003.

[186] Mohammad I. Younis. *Modeling and Simulation of Microelectromechanical Systems in Multi-Physics Fields*. PhD thesis, Virginia Polytechnic Institute and State University, Department of Engineering Mechanics, June 2004.

[187] C. Zener. Internal friction in solids I: Theory of internal friction in reeds. *Physical Review*, 52:230–235, 1937.

[188] C. Zener. Internal friction in solids II: General theory of thermoelastic internal friction. *Physical Review*, 53:90–99, 1938.

[189] C. Zener. *Elasticity and Anelasticity in Metals*. University of Chicago Press, 1948.

[190] C. Zener, W. Otis, and R. Nuckolls. Internal friction in solids III: Experimental demonstration of thermoelastic internal friction. *Physical Review*, 53:100–101, 1938.

[191] O. C. Zienkiewicz and R. Taylor. *The Finite Element Method, Volume 1: The Basis*. Butterworth-Heinemann, fifth edition, 2000.

[192] O. C. Zienkiewicz and R. Taylor. *The Finite Element Method, Volume 3: Fluid Dynamics*. Butterworth-Heinemann, fifth edition, 2000.

[193] J.D. Zook, D.W. Burns, H. Guckel, J.J. Sniegowski, R.L. Engelstad, and Z. Feng. Characteristics of polysilicon resonant microbeams. *Sensors and Actuators A*, 35:51–59, 1992.

[194] Maciej Zworski. Resonances in physics and geometry. *Notices of the AMS*, 46(3):319–328, March 1999.