EFFICIENT METHODS FOR SOLVING BIOMECHANICAL EQUATIONS

By

Toshiro K. Ohsumi

A Thesis Submitted to the Graduate
Faculty of Rensselaer Polytechnic Institute
in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY
Major Subject: Computer Science

Approved by the
Examining Committee:

Joseph E. Flaherty, Thesis Adviser

Franklin T. Luk, Member

David R. Musser, Member

Robert L. Spilker, Member

Linda R. Petzold, Member

Rensselaer Polytechnic Institute
Troy, New York

April 2003
(For Graduation May 2003)

EFFICIENT METHODS FOR SOLVING BIOMECHANICAL EQUATIONS

By

Toshiro K. Ohsumi

An Abstract of a Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Computer Science

The original of the complete thesis is on file
in the Rensselaer Polytechnic Institute Library

Examining Committee:

Joseph E. Flaherty, Thesis Adviser
Franklin T. Luk, Member
David R. Musser, Member
Robert L. Spilker, Member
Linda R. Petzold, Member

Rensselaer Polytechnic Institute
Troy, New York

April 2003
(For Graduation May 2003)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

There are more people to acknowledge than I can possibly remember. Projects of the type contained in this thesis is more a collaboration of many people rather than that of one individual.

Foremost, I would like to thank my advisor, Dean Joseph E. Flaherty, whose advice and shared knowledge and wisdom in all matters is immeasurable. I would also like to thanks my thesis committee members, who have all given me excellent advice and information.

One person who I greatly regret not putting on my thesis committee is Prof. Victor H. Barocas, in the Department of Biomedical Engineering at the University of Minnesota (Twin Cities Campus). His knowledge and patience was invaluable.

I would also very much like to thank Dr. M. Aiffa and Prof. S. Adjerid (Dept. of Mathematics, Virginia Polytechnic Institute and State University) whose mentoring in my early years at RPI have been invaluable.

There are many members of SCOREC, both past and present, to thank: Dr. M. W. Beall (Symmetrix), Dr. C. Bottasso (Politecnico di Milano), Dr. D. Datta, Dr. O. Klaas (Symmetrix), Dr. S. Lankalapalli, Dr. K. Pinchedez, Prof. J.-F. Remacle (Universite Catholique de Louvain), and Prof. M. S. Shephard.

There are also two visitors at RPI who I must thank: Prof. J. C. Butcher (Dept. of Mathematics, The University of Auckland, New Zealand) and Prof. D. Givoli (Dept. Aerospace Engineering, Technion—Israel Institute of Technology). Their time, knowledge, and comradery exemplify the reasons why visiting positions are offered.

There are many colleagues in the Department of Computer Science to thank for the research, particularly Dr. G. Chen, Dr. R. L. Loy (Argonne National Laboratory), Prof. W. Maniatty (Dept. Computer Science, University at Albany, New York), Prof. R. McNaughton, Dr. M. Nibhanupudi, Dr. C. Norton (Jet Propulsion Laboratory, NASA) Prof. B. Szymanski, Prof. J. D. Teresco (Williams College), and Mr. L. Ziantz (Dickinson College).

This research was partially supported by the National Science Foundation's (NSF) VIGRE Program (grant DMS-9983646) and NSF grant CCR-9896198.

All of the UML figures in this thesis was generated using TogetherSoft's (now Borland's) Together 3.2, which reads in include and source code directories, reverse-engineers the code, and produces UML diagrams.

Humans do not work in a vacuum, and to this I must express my thanks to those in the department that made this process bearable, particularly the department administrative assistants, past and present labstaff, and Dr. R. P. Ingalls who worked tirelessly in an often thankless job.

Finally, thanks to my parents without whom this would not have been possible.

ABSTRACT

Several steps are taken to produce a partial differential equation solving environment (PDESE) for solving time-dependent nonlinear biomechanical problems using adaptive finite element methods (FEMs) in one-, two-, and three-dimensions. We develop several methods to increase the efficiency of FEMs used to solve systems of partial differential equations arising in biomechanical models. The table look-up method is used to increase the speed of quadrature procedures specific to FEMs by interpolating the non-polynomial part of the integrand then using a pre-computed table of values. An enhanced prototype of a PDESE using h-adaptivity was used to solve a reaction-diffusion system of equations simulating the spread of Lyme disease taking into account vector dynamics. Procedures to correct sliver elements generated by mesh motion required for domain reshaping were developed and used in a h-adaptive two-dimensional micro-scale fluid pump simulation. The anisotropic biphasic theory (ABT) equations of Barocas and Tranquillo describing the formation of artificial arteries were solved using h-adaptivity and mesh motion in two dimensions. The efficacy of the adaptive procedures that were developed is shown. A modern object-oriented code closer to the ideal of a PDESE, Trellis, was enhanced and used to solve the ABT equations in three dimensions using h-adaptivity and mesh motion for the hexahedral interstitial cell traction assay (ICTA) and wound healing problems, hitherto unsolved. Warm restarts in DASPK were implemented within Trellis and shown to speed up adaptive computations by a factor of four for the ICTA problem.

CHAPTER 1
INTRODUCTION

1.1 Motivation

One goal of scientific computation is the creation of a partial differential equations solving environment (PDESE) which solves a broad class of partial differential equations (PDEs) by an automatic selection of [novel] algorithms without the user requiring the knowledge of the underlying software or hardware. A PDESE must provide a complete environment in which the systems of PDEs are solved, including discretizating the domain, solving the PDEs, and outputting the solution. Optionally, a PDESE may be able to handle optimization of problems modeled by PDEs and have a graphical user interface. The major component of a PDESE is a solver of systems of time-dependent nonlinear PDEs, which is the focus of this thesis. The efficient solution of PDEs is critical for a successful PDESE since, for example, the PDESE may include optimization routines that require accurately evaluating many large-scale PDEs with sharp gradients.

Systems of PDEs are used to model many biological and physical phenomena, such as plane wave non-dispersive sonar propagation and reflection (Helmholtz equation) [35], the motion of a viscous fluid (Navier-Stokes equations) [46, 131], or the formation of an artificial artery [23, 21, 22] (Barocas-Tranquillo biphasic equations). Finite element methods (FEMs) [80, 130, 146] are one such class of methods for solving PDEs that are particularly well-suited for geometrically complex domains. Techniques to improve the efficiency of FEMs are constantly needed, since ever larger and more complex systems are addressed. The advance in speed of high-performance processors is often negated by the ever-increasing size and complexity of the problems posed. Furthermore, optimization problems, such as finding the optimal forcing of water through a pump, requires solving a system of PDEs many times with different parameters. Since parameter changes may cause a significant change in the structure of the solution, any solution method must not be dependent on hard-coded methods and parameters defined within. Furthermore, since

the system of PDEs must be solved many times, the solution method must be efficient. Thus, we investigate analytic and automated software methods to increase the efficiency of FEMs.

1.2 Overview and Outline

In this thesis, we investigate several methods to improve the efficiency of FEMs and apply them to various problems of topical interest. We begin with a brief introduction to the FE notation used in this thesis. Standard mathematical notations and notions are briefly covered in §A. Historical reviews are presented at appropriate points within the text, rather than collected in one section.

FEMs require frequent evaluation of integrals of the form

$$\int_T P_1(\mathbf{x})\, P_2(\mathbf{x})\, f(\mathbf{x})\, d|\mathbf{x}| \tag{1.1}$$

where P_1 and P_2 are polynomials, f is an arbitrary (non-polynomial) function, and T is either a triangle or tetrahedron. In §2, we discuss the theory and application of a new method we have developed, *table look-up*, of computing (1.1). The idea is to approximate f by a polynomial, then look up the exact integral of the resultant integrand in a precomputed table. This new method has the potential of being extremely fast, depending on how easily f can be approximated. Furthermore, we have improved the efficiency of the table look-up method by splitting the interpolation problem and reduced the size of the table by exploiting symmetries in the hierarchical basis functions.

In §3, we investigate the adaptive solution of three nonlinear time-dependent problems in two dimensions using a prototype for a PDESE written by Aiffa [9] and extended herein. The first problem, in §3.2, is the axially symmetric anisotropic biphasic theory (ABT) equations of Barocas and Tranquillo, which models the formation of artificial arteries. Physically, one starts with a collagen gel that compacts. The final product is the artery and the goal is an initial configuration so that the artery is strong enough to withstand pulastile forces from blood flow as well as the stresses of surgical suture at the ends.

Thus, a motivation for creating a PDESE is to find the optimal initial configuration such that the least amount of material is used while retaining the necessary strength requirements. The ABT equations are nonlinear and have a domain that changes in time, and the solutions (*e.g.,* the concentration of the gel) have steep gradients, making the equations extremely difficult to solve accurately. The nonlinear behavior of the ABT equations does not easily fall within traditional classifications of PDEs, *i.e.,* parabolic or hyperbolic; thus, methods "tuned" to a particular type of PDE are not applicable. Our simulations are the first to use mesh refinement (h-adaptivity) and mesh-motion techniques to solve the ABT equations. We show the efficiency of using h-adaptive methods. A temporally-adaptive method-of-lines scheme is used to handle time-dependency, using DASPK [37], which uses an adaptive backward difference formula (BDF). DASPK is needed in order to handle the differential-algebraic equations (DAEs) [13] resulting from spatial discretization of the equations. We also simulate the formation of a reinforced artifical artery shunt using a complex domain. This is the first time the ABT equations were solved in a non-cylinderical domain. The second problem that we solved, in §3.3, is the viscous flow in a moving micro-fluidic pump. The domain moves to force fluid through the pump. The pump is typically used in drug-delivery systems or in toxic gas samplers. This novel problem is solved using mesh-motion to follow the moving domain, and h-adaptivity to capture sharp gradients in the y-velocity caused by re-entrant corners. The non-uniform scaling of the domain causes difficulties in solving this problem, which are addressed by rescaling the problem. Mesh-motion may create some badly shaped elements, which are corrected, using methods in §3.3.3.2. Since the pump is very small (order of micro-meters), efficiency is paramount. Finding parameters to maximize the efficiency required many runs; thus, necessitating efficient solution methods. These are the first simulations of this type of micro-fluidic pump. We also describe our h-adaptive solution of a new reaction-diffusion problem modeling a Lyme disease epidemic in §3.4 that was previously unsolved. The problem has a very sharp gradient in the initial condition, so h-adaptivity was required to solve this problem with accuracy in a reasonable amount of time. We demonstrated travelling wave solutions that substantiate the theory and verified the model for asymptotic

values.

The most complex problem solved in this thesis is described in §4 where the ABT equations are solved using h-adaptivity with mesh motion in three dimensions in §4.2. No geometrical symmetries to reduce the dimension are assumed. This is very significant in that a much larger class of simulations may be explored, such as the wound problem in §4.2.2, which cannot be reduced to a two-dimensional problem. Indeed, it is the first time that the ABT equations have been solved in three-dimensions, let alone adaptively. To solve these equations, a new framework, Trellis, was developed at SCOREC [29], with contributions provided by this thesis. The data structure is described in §4.1.6. Our implementation of warm restart interfaces to DASPK, described in §4.1.8, greatly speeds up h-adaptive computation. Our addition of mesh motion and some nonlinear functionality enhance Trellis' capabilities. Trellis has most of the desirable features of a PDESE, namely the ability to solve a large class of problems efficiently and automatically. (It is not yet to the point of being able to shield the user from some of the underlying code or to have incorporated optimization routines.)

Finally, a summary of this thesis and future work are covered in §5. The full potential of the table look-up method may be realized in a future project that is detailed in §5.2. A number of avenues of future research are described.

1.3 Finite Element Methods Background

1.3.1 Introduction

The finite element method is briefly discussed for a general three-dimensional time-dependent problem. A detailed introduction of adaptivity and mesh motion is postponed until §3.2.3.5. For more in-depth coverage along with historical notes, the reader is referred to the references [80, 130, 146]. Some of the notation and notions used in this thesis appear in §A.

1.3.2 Finite Element Methods

We solve systems of PDEs of the form

$$\mathbf{g}\left(\mathbf{x}, t, \mathbf{u}, \frac{\partial \mathbf{u}}{\partial t}\right) + \mathbf{f}(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}) + \sum_{i=1}^{d} \frac{\partial}{\partial x_i} \mathbf{a}_i(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}) = 0, \qquad \mathbf{x} \in \Omega, \qquad t > 0, \tag{1.2}$$

where n is the dimension of the vector functions $\mathbf{u}$, $\mathbf{g}$, $\mathbf{f}$, and $\mathbf{a}_i$, $i = 1, 2, \ldots, d$. For time-dependent problems, $\mathbf{g}$ will be restricted to the vector function

$$\mathbf{g}_i = \alpha_i \frac{\partial \mathbf{u}_i}{\partial t}, \tag{1.3}$$

where α_i is either 0 or 1, and we will assume that appropriate initial conditions (for components of $\mathbf{u}$ where $\alpha_i = 1$),

$$\mathbf{u}_i(\mathbf{x}, 0) = \mathbf{u}_{i0}(\mathbf{x}), \tag{1.4}$$

where $\mathbf{u}_{i0}(\mathbf{x}) \in H^1(\Omega)$, have been imposed.

Multiplying (1.2) by a test function, $\mathbf{v} \in H_0^1(\Omega)$, and integrating on Ω gives the weak form

$$\left(\mathbf{g}\left(\mathbf{x}, t, \mathbf{u}, \frac{\partial \mathbf{u}}{\partial t}\right), \mathbf{v}\right)_\Omega + (\mathbf{f}(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \mathbf{v})_\Omega + \sum_{i=1}^{d} \left(\frac{\partial}{\partial x_i} \mathbf{a}_i(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \mathbf{v}\right) = \mathbf{0}. \tag{1.5}$$

Applying the divergence theorem (a specialization of Stokes Theorem, *cf.* §A, (A.17)) to (1.5), gives

$$\left(\mathbf{g}\left(\mathbf{x}, t, \mathbf{u}, \frac{\partial \mathbf{u}}{\partial t}\right), \mathbf{v}\right)_\Omega + (\mathbf{f}(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \mathbf{v})_\Omega - \sum_{i=1}^{d} \left(\mathbf{a}_i(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \frac{\partial \mathbf{v}}{\partial x_i}\right)_\Omega + \sum_{i=1}^{d} \langle \tilde{\mathbf{a}}_i, \mathbf{v} \rangle = \mathbf{0}, \tag{1.6}$$

where $\tilde{\mathbf{a}}_i = \mathbf{a}_i$ on $\Gamma \backslash \Gamma_N$ and a Neumann boundary condition on Γ_N.

Assume the domain Ω is tessellated into simplices, Ω_e, $e = 1, \ldots, N_\Delta$, such that $\cup_{e=1}^{N_\Delta} \Omega_e = \Omega$, and $\Omega_i \cap \Omega_j$ for $i \neq j$ is either $\emptyset$ or a simplex of dimension $d - 1$.

(We do not consider the case where the intersection's dimension is lower than $d-1$.) The collection of Ω_e is referred to as the *mesh*. The tessellation induces a $(d-1)$-dimensional tessellation on Γ, denoted as Γ_e, $e = 1, \ldots, N_\delta$, such that $\cup_{e=1}^{N_\delta} \Gamma_e = \Gamma$ and $\Gamma_i \cap \Gamma_j$ for $i \neq j$ is either $\emptyset$ or a simplex of dimension $d-2$. When Γ is curved, the tessellation may only be an approximation of Ω. This will be addressed in §2. Applying (1.6) on the mesh gives

$$\sum_{e=1}^{N_\Delta} \left(\mathbf{g}\left(\mathbf{x}, t, \mathbf{u}, \frac{\partial \mathbf{u}}{\partial t}\right), \mathbf{v} \right)_{\Omega_e} + \sum_{e=1}^{N_\Delta} \left(\mathbf{f}(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \mathbf{v}\right)_{\Omega_e} - \sum_{e=1}^{N_\Delta} \sum_{i=1}^{d} \left(\mathbf{a}_i(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \frac{\partial \mathbf{v}}{\partial x_i} \right)_{\Omega_e} + \sum_{i=1}^{d} \sum_{e=1}^{N_\delta} \left\langle \tilde{\mathbf{a}}_i(\mathbf{x}, t, \mathbf{u}, \nabla \mathbf{u}), \mathbf{v} \right\rangle_{\Gamma_e} = \mathbf{0}. \quad (1.7)$$

A necessary approximation is the discretization of the solution space. Assuming that the exact solution of (1.6) is $\mathbf{u} \in H^1(\Omega)$, then the space $H^1(\Omega)$ is approximated by the subspace S^N, which is the space of functions spanned by a complete set of polynomial basis functions of order p over the mesh. The basis functions are denoted as $N_i(\mathbf{x})$, $i = 1, 2, \ldots, N$. (Obviously, N depends on p.) Similarly, the space $H_0^1(\Omega)$ is approximated by the subspace S_0^N, where the basis functions are zero on Γ_D. Thus, one approximates a component of the solution $\mathbf{u}_k$ as a linear combination of basis functions

$$\mathbf{u_k} \approx \mathbf{U}_k = \sum_{i=1}^{N_k} a_{ik} N_i(\mathbf{x}), \quad (1.8)$$

where the coefficients a_{ik} are scalars. The function space approximations are denoted as the upper-case analog of the lower-case variable. The order of approximation of $\mathbf{u}$ need not be the same for all components of $\mathbf{u}$ (N_j may not be equal to N_k for $j \neq k$), leading to mixed methods [38]. A similar approximation is applied for the test function $\mathbf{v}$ with the provision that the order of approximation of $\mathbf{v}_k$ is the same as the order of approximation of $\mathbf{u}_k$. Then, one may replace $\mathbf{u}$ and $\mathbf{v}$ by $\mathbf{U}$ and $\mathbf{V}$

respectively in (1.7), giving

$$\sum_{e=1}^{N_\Delta}\left(\mathbf{g}\left(\mathbf{x},t,\mathbf{U},\frac{\partial\mathbf{U}}{\partial t}\right),\mathbf{V}\right)_{\Omega_e}+\sum_{e=1}^{N_\Delta}(\mathbf{f}(\mathbf{x},t,\mathbf{U},\nabla\mathbf{U}),\mathbf{V})_{\Omega_e}-\sum_{e=1}^{N_\Delta}\sum_{i=1}^{d}\left(\mathbf{a}_i(\mathbf{x},t,\mathbf{U},\nabla\mathbf{U}),\frac{\partial\mathbf{V}}{\partial x_i}\right)_{\Omega_e}+\sum_{i=1}^{d}\sum_{e=1}^{N_\delta}\langle\tilde{\mathbf{a}}_i(\mathbf{x},t,\mathbf{U},\nabla\mathbf{U}),\mathbf{V}\rangle_{\Gamma_e}=\mathbf{0},\quad(1.9)$$

and taking care to set the coefficients so that $\mathbf{U}$ fulfills the boundary conditions, resulting in a nonlinear system of equations for the coefficients of the basis functions in $\mathbf{U}$ when (1.2) is time independent, or a system of nonlinear differential algebraic equations (DAEs) otherwise. The exact form of the equations that arises is not discussed here as this is well known [130, 146]. The above is intended to only cover the nomenclature used.

There is a choice of polynomial basis functions available on a tessellation by simplices. One choice is to use a class of basis functions which are hierarchic (*e.g.* [43]). That is, if a complete set of hierarchic basis functions were available for order p, denoted as N_i, $i = 1, 2, \ldots N_p$, with the basis function arranged from smallest order polynomial to largest, then there exists a number $N_{p-1} < N_p$ such that the set of basis functions N_i, $i = 1, 2, \ldots N_{p-1}$ is a complete basis function set of order $p - 1$. This implies that higher order basis functions serve as corrections to lower order approximations.

Finally, it should be noted that the size of the elements in the mesh need not be uniform. Indeed, as the computation progresses, it is shown in §3.2 and §3.3 to be very efficient to alter — or adapt — the size of the elements to efficiently solve the problem. This process is known as h-adaptivity. Furthermore, although not used in this thesis, it is also possible to change the order of approximation locally over a patch of elements, and is known as p-adaptivity. On occasions when both types of adaptivity are used, the method is denoted as hp-adaptivity.

CHAPTER 2
TABLE LOOK-UP METHOD

2.1 Introduction

Efficient implementation of p- and hp-adaptive finite element methods on serial and parallel computers has been the focus of recent research [52, 53, 65, 103, 109]. Issues related to element-level computation on curved domains involve

1. the choice of variable-order shape functions,
2. the construction of geometric approximations on large elements, and
3. the efficient evaluation of integrals appearing in element matrices and vectors.

Dey et al. [54, 55, 120] address the first two issues and present a framework for specifying and evaluating variable-order shape functions on conforming, unstructured meshes containing mixed topology elements. In part, they also discuss numerical integration [55].

Traditional Gaussian integration techniques [2] on tetrahedral elements are unavailable for integration orders beyond eight [48]. Although higher order integration methods exploiting tensor products are available [59, 76, 121], they are not efficient when used with curvilinear elements or elements not admitting tensor-product shape functions. If the integrand is a polynomial then the necessary integrals can be precomputed and stored. This idea has been used with hierarchical finite elements for bilinear functionals on two-dimensional domains [116]. Precomputed matrices for triangular elements for second-order elliptic partial differential equations on planar domains are also available [122]. Atkins and Shu [15] used this approach with the discontinuous Galerkin method for hyperbolic equations. Symbolic computation employs a similar idea [101, 102, 141].

In curvilinear domains, the integrands are not polynomials even when coefficients and bases are; hence, the precomputed integral tables cannot be applied directly to the entire integrand. Here, we develop integral tables for three-

dimensional curvilinear finite elements associated with p-refinement by interpolating non-polynomial parts of the integrand by polynomials to an order sufficient to ensure the optimal convergence rate of the finite element solution and evaluating the resulting element level polynomial integrals exactly using precomputed tables. Symmetries in the polynomial integral entries are identified using the concept of parametric coordinate permutations to reduce the number of unique nonzero entries that require precomputation and storage.

The discretization error of this procedure may be reduced by a proper choice of the interpolating polynomial. Chen and Babuška [41, 42] construct an interpolant on tetrahedra where the error constant has a linear growth with polynomial degree. The points are symmetrically disposed on tetrahedra and are the minimum number necessary to interpolate a complete polynomial of a given degree.

Several methods of integration are reviewed in §2.2. They all exhibit some deficiencies that are addressed by the table look-up method described in §2.3. Next, techniques for optimizing the implementation of the table look-up method are presented in §2.4 followed by theoretical bounds on the integration error in §2.5. Finally, a numerical example comparing common methods and the table look-up method is given in §2.6.

2.2 Elemental Integrals and Their Approximation

In solving (1.9), one must compute integrals over a mesh element, Ω_e. For example, we consider a special case of (1.2) in three-dimensions,

$$c(\mathbf{x})\nabla^2 u(\mathbf{x}) + \kappa^2 u(\mathbf{x}) = 0, \qquad \mathbf{x} \in \Omega, \tag{2.1}$$

where u is a scalar variable and $c(\mathbf{x})$ is a given function, known as the generalized Helmholtz equation. (2.1) gives rise to the special case of (1.9)

$$\sum_{i=1}^{N_\Delta} \left(\kappa^2 U, V\right)_{\Omega_e} - \sum_{i=1}^{N_\Delta} \left(c(\mathbf{x})\nabla U, \nabla V\right) + \sum_{i=1}^{N_\delta} \langle U, V \rangle = 0. \tag{2.2}$$

We will focus on first two terms of (2.2) noting that the treatment of the last term is handled in a similar manner but in a lower dimension. In the numerical Helmholtz equation examples of §2.6, $c(\mathbf{x}) = 1$, and we use polynomial basis functions. However, the ideas presented in this chapter may be readily generalized to (1.2). This will be discussed in §5.2.

A typical contribution to an element stiffness, mass, *etc.* matrix $\mathbf{K}^e$ of a second-order linear partial differential equation, such as (1.2), has the form [80]

$$K_{ij}^e = \int_{\Omega_e} \frac{\partial^\alpha N_i}{\partial x_k^\alpha} c(\mathbf{x}) \frac{\partial^\beta N_j}{\partial x_\ell^\beta} d|\mathbf{x}|, \qquad \alpha, \beta = 0, 1, \qquad k, l = 1, 2, 3, \tag{2.3}$$

where N_i is a shape function, $\mathbf{x} = [x_1, x_2, x_3]^T$ is a position vector, $d|\mathbf{x}|$ is a volume infinitesimal, $c(\mathbf{x})$ is a coefficient of the differential system or a linearization of the nonlinear problem, and Ω_e is the domain of element e. With $\alpha = \beta = 1$, the entries (2.3) correspond to those of a typical element stiffness matrix $\mathbf{K}^e$; $\alpha = \beta = 0$ yields entries of an element mass matrix $\mathbf{M}^e$; $\alpha = 0$, $\beta = 1$ yields entries of a convection matrix $\mathbf{C}^e$; and $\alpha = 0$ and omission of the second shape function yields terms of a load vector $\mathbf{f}^e$. Indeed, integrals arising from natural boundary conditions also have this form on domains Γ_e which is the boundary of Ω_e [80].

Since quadrature rules are not available for an arbitrarily shaped tetrahedral elements, Ω_e, the integral is mapped from Ω_e onto a canonical element, T, for which quadrature rules are known, as shown in Figure 2.1.

The original element, Ω_e, may have one or more curved faces, in which case the transformation is nonlinear. Otherwise, the transformation is linear. The construction of the transformation may be found in [127] and [55].

Transforming (2.3) to an integral on a canonical element T yields an integral of the form

$$I = \int_T \Theta(\boldsymbol{\xi}) \Upsilon(\boldsymbol{\xi}) d|\xi| \tag{2.4a}$$

with $\boldsymbol{\xi} = [\xi_1, \xi_2, \xi_3]^T$ where

$$\Theta(\boldsymbol{\xi}) = \frac{\partial^\alpha N_i}{\partial \xi_m^\alpha} \frac{\partial^\beta N_j}{\partial \xi_n^\beta}, \qquad \Upsilon(\boldsymbol{\xi}) = c(\mathbf{x}(\boldsymbol{\xi})) \frac{\partial^\alpha \xi_m}{\partial x_k^\alpha} \frac{\partial^\beta \xi_n}{\partial x_\ell^\beta} J(\boldsymbol{\xi}), \tag{2.4b}$$

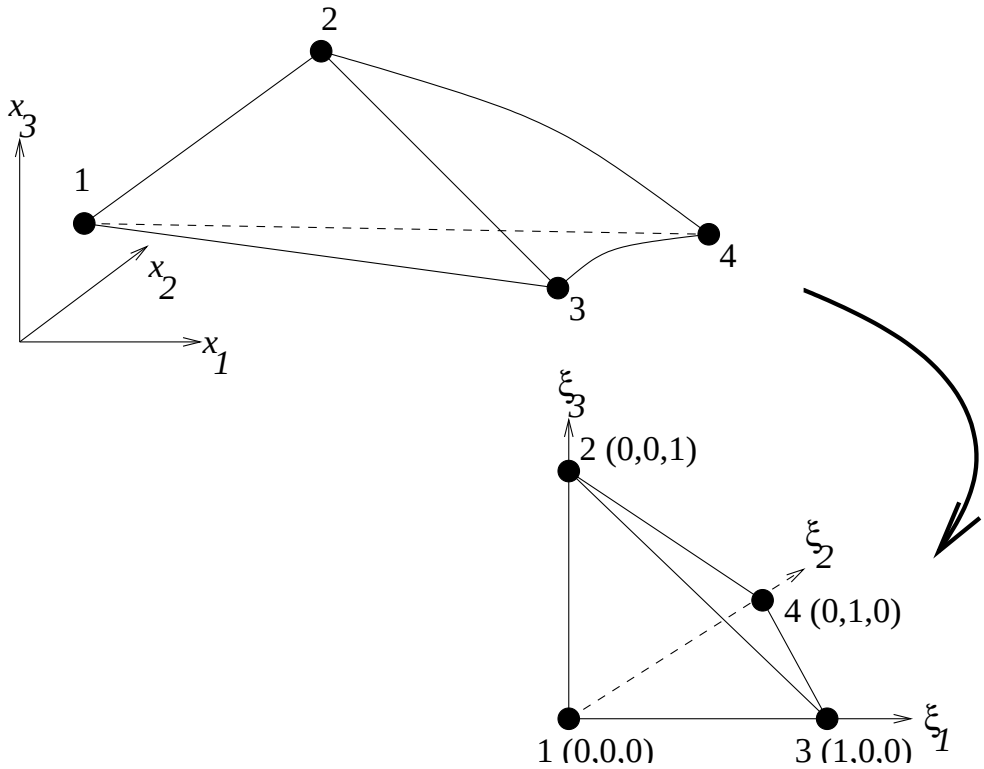

Figure 2.1: Mapping of an arbitrary element Ω_e to a canonical element T.

for $\alpha, \beta = 0, 1$ and $m, n, k, l = 1, 2, 3$, and $\mathbf{x}(\boldsymbol{\xi})$ is a mapping of Ω_e in physical $\mathbf{x}$ space to T in computational $\boldsymbol{\xi}$ space having the Jacobian

$$J(\boldsymbol{\xi}) = \det \begin{bmatrix} \frac{\partial x_1}{\partial \xi_1} & \frac{\partial x_1}{\partial \xi_2} & \frac{\partial x_1}{\partial \xi_3} \\ \frac{\partial x_2}{\partial \xi_1} & \frac{\partial x_2}{\partial \xi_2} & \frac{\partial x_2}{\partial \xi_3} \\ \frac{\partial x_3}{\partial \xi_1} & \frac{\partial x_3}{\partial \xi_2} & \frac{\partial x_3}{\partial \xi_3} \end{bmatrix}. \tag{2.5}$$

The function $\Theta(\boldsymbol{\xi})$ is a polynomial with polynomial shape functions and, hence, is easily integrated. The metrics $\partial \xi_m / \partial x_k$ are functions of $1/J(\boldsymbol{\xi})$ and are generally not polynomials even when $\mathbf{x}(\boldsymbol{\xi})$ is polynomial. The exception occurs with linear transformations where J is constant. Even in this case, $c(\mathbf{x}(\boldsymbol{\xi}))$ need not be a polynomial; therefore, $\Upsilon(\boldsymbol{\xi})$ is generally not a polynomial and, typically, will not be integrable by symbolic means.

Without exact integration, (2.4a) is approximated in a manner consistent with the accuracy of the finite element interpolation of the exact solution of the partial

differential system. If the basis contains complete polynomials of at most degree p, then $\Theta(\boldsymbol{\xi})$ is a complete polynomial of at most degree $2p$.

The complexity of quadrature procedures is appraised in terms of the number of times the integrand is evaluated. For the purposes of computing and comparing the complexity of various methods, we assume the integral must be calculated to order ϱ accuracy, *i.e.*, the quadrature procedure must be exact for all polynomial integrands of degree ϱ or less.

2.2.1 Symmetric Gaussian Quadrature

Symmetric Gaussian quadrature [49] provides an approximation of (2.4a) having the form

$$I \approx \sum_{\nu=1}^{n_s(\varrho)} w_\nu \Theta(\boldsymbol{\xi}^{(\nu)}) \Upsilon(\boldsymbol{\xi}^{(\nu)}) \tag{2.6}$$

where w_ν and $\boldsymbol{\xi}^{(\nu)}$, $\nu = 1, 2, \ldots, n_s(\varrho)$, are the Christoffel weights and evaluation points, respectively, and $n_s(\varrho)$ is the number of evaluation points needed for order ϱ. The evaluation points are symmetrically placed within T.

Symmetric Gaussian quadrature has been the preferred integration rule for use with finite element methods since rules with a (nearly) minimal number of function evaluations for a given order are known for low orders [48]. The complexity of an evaluation is $O(\varrho^3)$ [49, 84] for a method of order ϱ. Unfortunately, the points and weights are only known to order eight for integration on tetrahedra [48].

2.2.2 Generalized Product Rule (GPR) Quadrature

By mapping T to a $[-1,1] \times [-1,1] \times [-1,1]$ cube, (2.4a) can be expressed as a product of three one-dimensional integrals [54]. The mapping from T to the cube, shown in Figure 2.2.2, is given by [54]

$$\xi_1 = \frac{1}{8}(1+\xi_1')(1-\xi_2')(1-\xi_3') \tag{2.7}$$

$$\xi_2 = \frac{1}{4}(1+\xi_2')(1-\xi_3') \tag{2.8}$$

$$\xi_3 = \frac{1}{2}(1+\xi_3') \tag{2.9}$$

$$\tag{2.10}$$

so that the Jacobian is

$$\left|\frac{\partial \xi_i}{\partial \xi'_j}\right| = \frac{1}{64}\left(1-\xi'_2\right)\left(1-\xi'_3\right)^2 . \tag{2.11}$$

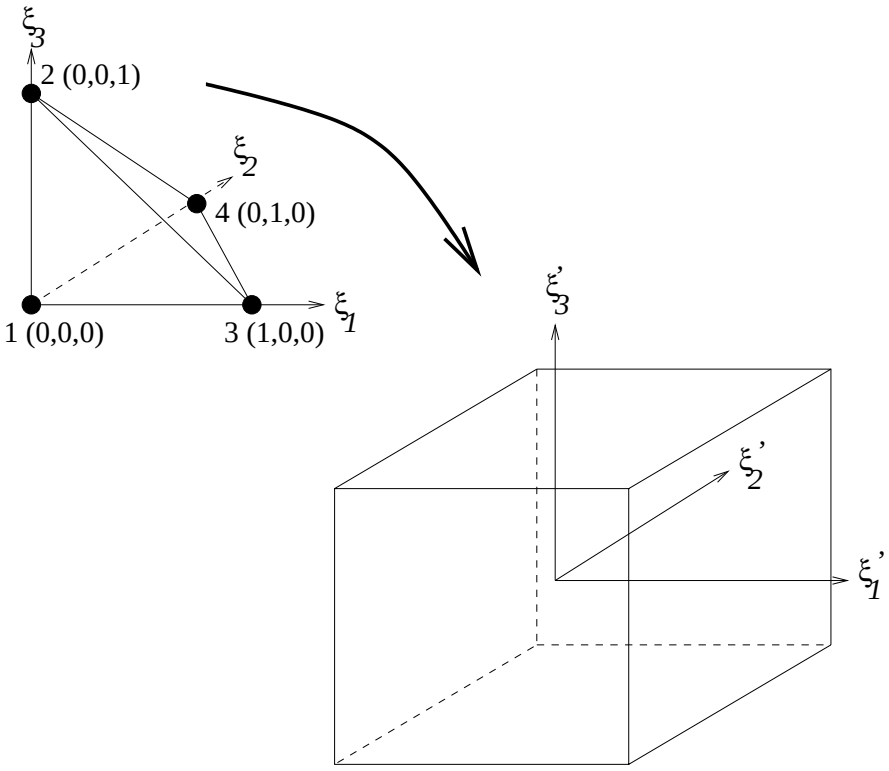

Figure 2.2: GPR coordinate transformation

Using one-dimensional Gaussian quadrature, the complexity in each coordinate direction is $O(\varrho)$; thus, the total complexity is $O(\varrho^3)$. This method of integration is known as a *Generalized Product Rule* (GPR) [49] and is preferred to the symmetric product rules of Grundmann and Möller [71] since the GPR weights are positive [90]; thus, accuracy is not lost due to round-off error. Since neither the shape nor the symmetry of the tetrahedral element is used, GPR integration will use more points than symmetric Gaussian quadrature for a given order; however, the rules for one-dimensional Gaussian quadrature are available for all ϱ.

2.2.3 Sum Factorization and Vector Quadrature

If $\Theta(\boldsymbol{\xi})\Upsilon(\boldsymbol{\xi})$ can be factored as $F_1(\xi_1)F_2(\xi_2)F_3(\xi_3)$ then (2.4a) is evaluated in $O(\varrho)$ operations as [59]

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} F_1(\xi_1)F_2(\xi_2)F_3(\xi_3)d\xi_1 d\xi_2 d\xi_3 = \int_{-1}^{1} F_1 d\xi_1 \int_{-1}^{1} F_2 d\xi_2 \int_{-1}^{1} F_3 d\xi_3. \quad (2.12)$$

This method of quadrature is known as sum factorization.

However, for curved domains, $\Theta(\boldsymbol{\xi})\Upsilon(\boldsymbol{\xi})$ cannot be factored exactly. Therefore, $\Theta(\boldsymbol{\xi})\Upsilon(\boldsymbol{\xi})$ must be interpolated to a tensor-product form, requiring $O(\varrho^3)$ operations. A serious difficulty in interpolating the integrand in this manner is that the basis is not complete for order ϱ. Thus, if F_ν, $\nu = 1, 2, 3$, are polynomials whose degrees sum to ϱ or less, then the functional factorization cannot represent all polynomials of three variables of order ϱ or less. For example, the function $1 + \xi_2 + 2\xi_3 + 3\xi_2\xi_3$ cannot be exactly represented as a product $F_1(\xi_1)F_2(\xi_2)F(\xi_3)$ of polynomials. Thus, a higher-order interpolation would be needed to achieve the same order of accuracy as Gaussian quadrature, making sum factorization inefficient.

Suppose that $\Theta(\boldsymbol{\xi})$ and $\Upsilon(\boldsymbol{\xi})$ are approximated as

$$\Theta(\boldsymbol{\xi}) \approx \sum_{\nu=1}^{n_v(\varrho)} a_\nu \varphi_\nu(\boldsymbol{\xi}), \qquad \Upsilon(\boldsymbol{\xi}) \approx \sum_{\nu=1}^{n_v(\varrho)} b_\nu \varphi_\nu(\boldsymbol{\xi}) \quad (2.13)$$

with a basis $\{\varphi_\nu(\boldsymbol{\xi})\}_{\nu=1}^{n_v(\varrho)}$ that is orthogonal in $L^2(T)$. Then (2.4a) is approximated as

$$I \approx \sum_{\nu=1}^{n_v(\varrho)} a_\nu b_\nu. \quad (2.14)$$

This method is called vector factorization. Evaluation of the sum and interpolation has $O(\varrho^3)$ complexity. In comparison, the table look-up method (§2.3) will require only the interpolation of Υ, and thus be less expensive than vector quadrature.

Thus, sum factorization and vector quadrature both require the solution of an interpolation problem on curved domains. Due to the cited inefficiencies, these quadrature methods are not considered further.

2.3 Integration by Table Look-Up

As an alternative to the quadrature techniques of §2.2, we describe a table look-up procedure where the non-polynomial portion Υ of (2.4a) is approximated by the polynomial of degree q

$$\Upsilon(\boldsymbol{\xi}) \approx \Upsilon^*(\boldsymbol{\xi}) = \sum_{\nu=1}^{n_t(q)} a_\nu \phi_\nu(\boldsymbol{\xi}), \tag{2.15a}$$

where

$$n_t(q) = \frac{(q+1)(q+2)(q+3)}{6} \tag{2.15b}$$

for a complete polynomial of degree q, and the result is integrated exactly to yield

$$I \approx \int_T \Theta(\boldsymbol{\xi})\Upsilon^*(\boldsymbol{\xi})\, d|\xi| = \sum_{\nu=1}^{n_t(q)} a_\nu \int_T \Theta(\boldsymbol{\xi})\phi_\nu(\boldsymbol{\xi})\, d|\xi|. \tag{2.16}$$

While any basis may be used in (2.15a), it seems expedient to use the same basis function as the finite element solution's, *i.e.* $\phi_\nu = N_\nu$, $\nu = 1, 2, \ldots, n_t(q)$. If the coefficients a_ν, $\nu = 1, 2, \ldots, n_t(q)$, are to be determined by interpolation, then Chen and Babuška [41, 42] provide a "nearly optimal" set of points which limit the growth of the constant of the interpolation error on tetrahedra to $O(q)$. The points are symmetrically disposed on T and are the minimal number to interpolate a polynomial of degree q. Since Υ depends only on the partial differential system and the mesh geometry (cf. (2.4b)), it requires only one interpolation for all combinations of i and j in (2.4b).

The determination of a_ν, $\nu = 1, 2, \ldots, n_t(q)$, in (2.15a) by interpolation requires the solution of the linear algebraic system

$$\boldsymbol{\Phi}\mathbf{a} = \mathbf{d} \tag{2.17a}$$

where

$$\Phi_{mn} = N_n(\boldsymbol{\xi}^{(m)}), \qquad d_m = \Upsilon(\boldsymbol{\xi}^{(m)}), \qquad m, n = 1, 2, \ldots, n_t(q), \tag{2.17b}$$

with $\boldsymbol{\xi}^{(m)}$, $m = 1, 2, \ldots n_t(q)$, being the Chen and Babuška [41, 42] interpolation points. The algebraic complexity of solving (2.17) depends on the finite element basis. Use of a Lagrangian basis [80] renders it diagonal. The situation is more complex with the hierarchical bases commonly used for p-refinement [43, 130]. Elements of the hierarchical basis are associated with a specific mesh entity (region, face, edge, or vertex) and they vanish, to maintain continuity, on the boundary of all regions (finite elements) not containing that entity. The Chen and Babuška interpolation points are also on vertices, edges, faces, and element interiors. Thus, an interpolation at a vertex will only involve the (linear) shape function that is identified with that vertex. All other shape functions vanish there, and the coefficient in (2.17) that is associated with the vertex will be explicitly determined. Likewise, when interpolation occurs at a point on an edge, the only nontrivial coefficients of (2.17) are those identified with the edge and the two vertices bounding the edge. With the vertex coefficients determined, the edge coefficients may determined by block forward substitution. Determination of coefficients associated with faces and finite elements follow in similar fashion. Thus, (2.17) may be put in a block lower triangular form with coefficients and interpolation points ordered and determined by vertices, edges, faces, and regions. Unfortunately, there are $O(q^3)$ points and unknowns within element interiors, so this ordering leaves a large block to be determined when q is large. Assuming a direct solution procedure saving the factorization of $\boldsymbol{\Phi}$, the algebraic complexity of the block forward substitution would still be $O(q^6)$, as it would be for an arbitrary ordering. In determining the computational cost, we assume that integrand evaluations incur a much larger expense than algebraic operations; thus, we neglect the $O(q^6)$ cost involved with forward substitution. Moreover, since the interpolation is split over mesh entities, the coefficient bounding the cost is small ($< 1/6$); thus, for moderate q's used in practical problems, neglecting the forward substitution cost is justified.

In addition to this algebraic cost, there is also the complexity associated with evaluating Υ in (2.17) at $O(q^3)$ points (*cf.* (2.15b)). Other costs associated with accessing memory to retrieve the value of an integral (2.4a) for a particular choice of α, β, *etc.* are assumed to be negligible. We show in §2.5 that $q \geq p$ is required to

maintain the optimal $O(h^p)$ convergence rate of the finite element method with exact integration; thus, the complexity of table look-up has the same $O(p^3)$ complexity associated with other quadrature methods (§2.2). If the integrand were polynomial, table look-up would be exact with an $O(1)$ complexity.

The table look-up method has several advantages:

1. The topology-dependent basis and interpolation points use fewer evaluations than, *e.g.*, product formulations.

2. A hierarchical basis with the Chen and Babuška [41, 42] interpolation points renders the interpolation problem (2.17) block lower triangular, and provides computational savings for moderate values of q.

3. The method presents a general approach to high-order integration on geometrically complex regions. The basis need not satisfy any special properties.

4. When $\Upsilon(\boldsymbol{\xi})$ is constant, table look-up provides exact integration in $O(1)$ time.

5. Table look-up may be used with a non-polynomial but integrable basis [16].

Relative to other quadrature methods, table look-up requires the solution of an interpolation problem on each finite element; however, this problem must only be solved once for each differential equation and mesh.

2.4 Implementation and Optimization of Table Look-Up

The integration table contains values of

$$\int_T \frac{\partial^\alpha N_i(\boldsymbol{\xi})}{\partial \xi_m^\alpha} \frac{\partial^\beta N_j(\boldsymbol{\xi})}{\partial \xi_n^\beta} N_k(\boldsymbol{\xi})\, d|\xi|. \tag{2.18}$$

With shape functions of degree p and interpolation of degree q, the tables for $\mathbf{K}^e$ or $\mathbf{M}^e$ would contain $O(p^6 q^3)$ entries for second-order, three-dimensional problems. Various symmetry and orthogonality considerations can reduce the size of these tables, and we describe an approach based on the shape functions of Carnevali *et al.* [43].

Repetitive zero and nonzero entries in the tables need not be stored. Instead, memory may be reduced by using integers to point to the unique (double-precision) floating point entries. This is done by using two arrays: one array holds unique floating point results of the integrals in a *result table*, and a second *index table* stores integers pointing to the result. A function converts the index $i, j, k, m, n, \alpha, \beta$ of an integral (2.18) to a unique entry in the index table which gives the appropriate answer in the result table. The tables are represented as one-dimensional arrays to reduce the number of pointers. Entries in the result table were generated using the symbolic integration system PARI [26], which manipulates polynomials in factors of 5 to 100 times faster than interpreted symbolic packages, such as Maple [97].

A reduction of the dimension of the index table is obtained from the symmetry of integration under cyclic permutation of the parametric coordinates, since integration is invariant under coordinate rotations. For example,

$$\int_T \frac{\partial^\alpha N_i\left(\xi_l,\xi_m,\xi_n\right)}{\partial \xi_l^\alpha}\frac{\partial^\beta N_j\left(\xi_l,\xi_m,\xi_n\right)}{\partial \xi_n^\beta} N_k\left(\xi_l,\xi_m,\xi_n\right)\ d|\xi| = \int_T \frac{\partial^\alpha N_i\left(\xi_m,\xi_n,\xi_l\right)}{\partial \xi_m^\alpha}\frac{\partial^\beta N_j\left(\xi_m,\xi_n,\xi_l\right)}{\partial \xi_l^\beta} N_k\left(\xi_m,\xi_n,\xi_l\right)\ d|\xi| \quad (2.19)$$

A further reduction in size is obtained by exploiting the symmetry or antisymmetry of the shape functions under a permutation of variables. For example, the edge shape functions of Carnevali [43] have the identity

$$N_i(\xi,\eta) = s(N_i) \times N_i(\eta,\xi), \quad (2.20)$$

where $s(N_i)$ is 1 if the degree of N_i is even and -1 otherwise. Thus, for example, one needs only to store integrals of

$$\int_T N_i\left(\xi_1,\xi_2,\xi_3\right) N_j\left(\xi_1,\xi_2\right) N_k\left(\xi_1,\xi_2,\xi_3\right)\ d|\xi| \quad (2.21)$$

but not of

$$\int_T N_i\left(\xi_1,\xi_2,\xi_3\right) N_j\left(\xi_2,\xi_1\right) N_k\left(\xi_1,\xi_2,\xi_3\right)\ d|\xi| \quad (2.22)$$

since the value of (2.22) is the same or the negative of (2.21) depending on the order

	$\mathbf{K}^e$		$\mathbf{M}^e$		$\mathbf{f}^e$	
	Asymp.	Compact	Asymp.	Compact	Asymp.	Compact
Entries (Millions)	0.5	0.05	0.5	0.04	0.007	0.0005
Size (MB)	4	0.4	4	0.32	0.06	0.004

Table 2.1: Unique nonzero precomputed entries for $p = 6$ compared to asymptotic estimates for full dense matrices and vectors.

of N_j.

Table 2.1 shows the reduction in size of the index table by exploiting symmetries present in the shape functions of Carnevali *et al.* [43] with $p = 6$. The asymptotic entries in Table 2.1 assume that the elemental matrices and vectors are dense and asymmetric. The shape functions of Carnevali *et al.* [43] also satisfy an orthogonality condition when the difference in degrees of the product of two shape function derivatives in $\mathbf{K}^e$ exceeds four. This sparsity, however, has not been exploited. Furthermore, if the element matrices are symmetric, this allows a further reduction of the number of index entries.

2.5 Integration Error Estimates

We examine the errors due to approximate integration for a Galerkin problem of the form: find $u \in H_0^1$ such that

$$A(v, u) = (v, f), \qquad \forall v \in H_0^1, \tag{2.23}$$

where $A(v, u)$ is a bilinear form satisfying the conditions of continuity and coercivity on H^1

$$|A(v, w)| \leq C_1 \|v\|_{1,\Omega} \|w\|_{1,\Omega}, \qquad A(v, v) \geq C_2 \|v\|_{1,\Omega}^2, \qquad \forall v, w \in H_0^1(\Omega). \tag{2.24}$$

The linear form (v, f) denotes the L^2 inner product over Ω. We also let $|\cdot|_{k,\Omega}$, $k \geq 1$, denote a seminorm involving derivatives of only order k on $H^k(\Omega)$, as opposed to the norm on $H^k(\Omega)$ where the derivatives were taken for all derivatives of order k and less.

With an exact integration and exact representation of Ω, a finite element solution $U \in S_0^N \subset H_0^1$ converges as $O(h^p)$ in H^1, where h is the maximum length of an element edge. The software used in §2.6 employs an exact mapping of Ω_e to the canonical element T [55]; thus, perturbations of this rate are only due to approximate integration. Towards quantifying this effect, let the strain energy and load potential in (2.23) be approximated as $A_*(v,u)$ and $(v,f)_*$. If $A_*(U,U)$ is coercive, then the finite element solution $U^* \in S_{*,0}^N$ with inexact integration satisfies [136]

$$\begin{aligned}\|u-U^*\|_{1,\Omega} \leq & C \inf_{V\in S_0^N}\left\{\|u-V\|_{1,\Omega} + \sup_{W\in S_0^N}\frac{|A(V,W)-A_*(V,W)|}{\|W\|_{1,\Omega}}\right. \\ & \left. + \sup_{W\in S_0^N}\frac{|(W,f)-(W,f)_*|}{\|W\|_{1,\Omega}}\right\}.\end{aligned} \tag{2.25}$$

In order to maintain the optimal rate of convergence, the two perturbations in (2.25) must also converge as $O(h^p)$. A regularity assumption is necessary for our perturbation analysis.

Definition 1. Let $v \in H^s(T)$, $s \geq 0$, then the transformation from T to Ω_e is *regular* if the Jacobian J is bounded and there exists a constant $C > 0$ such that

$$|v|_{s,T} \leq C\left\{\inf_{\boldsymbol{\xi}\in T} J(\boldsymbol{\xi})\right\}^{-\frac{1}{2}} h_e^s \|v\|_{s,\Omega_e} \tag{2.26}$$

where h_e is the length of the longest edge of Ω_e.

As noted, the Jacobian J of the transformation is not necessarily a polynomial; however, if it is then optimal convergence results are available as indicated in the following theorem.

Theorem 1. *Assume $J(\boldsymbol{\xi})$, $\boldsymbol{\xi} \in T$, is a polynomial of degree q satisfying (2.26), then the perturbation errors in (2.25) are $O(h^p)$ when integrals (2.4a) are evaluated by Gaussian quadrature of order $2p-1+q$.*

Proof. Wait and Mitchell [136] establish the result when $A(v,u)$ corresponds to a Laplacian-like operator. The results may easily be generalized to situations where

terms of the form Cvu, $C > 0$, are present in $A(v, u)$. □

For integration by table look up, we concentrate on a single element Ω_e and establish that

$$|A_e(V, W) - A_{*,e}(V, W)| \leq Ch_e^p \|V\|_{p+1,\Omega_e} \|W\|_{1,\Omega_e}, \tag{2.27}$$

where A_e is the portion of A on Ω_e. Having done this, it is clear that the bounds in (2.25) on Ω will be satisfied. Bounds for the perturbation involving (W, f) proceed along similar lines and will not be presented.

Definition 2. Let $L_k(\mathbf{x})$ be a Lagrange basis function over the domain T, with $L_i(\mathbf{t}_j) = \delta_{ij}$, where the set of interpolation points is $\mathbf{t}_j$. Assume the interpolation points are chosen such that the set L_k, $k = 1, \ldots, N_q$, is complete to order q. Then the *Lebesgue constant* is [42]

$$\lambda_q(T) = \sup_{\boldsymbol{\xi} \in T} \sum_{k=1}^{N_q} |L_k(\boldsymbol{\xi})| \,. \tag{2.28}$$

By definition, the Lebesgue constant is dependent on the location of the interpolation points as well as the order. The optimality of the interpolation points of Chen and Babuška [42] satisfy $\lambda_q(T) = O(q)$, unlike other interpolation points where $\lambda_q(T)$ may grow exponentially in q [41, 42]. This property is used in Theorem 2.

Proposition 1. *Assume* $\Upsilon \in C^\infty(T)$. *Then*

$$\|\Upsilon - \Upsilon^*\|_{\infty,T} \leq C_1(C_2 + \lambda_q(T)) \frac{|T|^{q-2}}{(q/3)!} |\Upsilon|_{q,T}, \qquad q > 2, \tag{2.29}$$

where $\lambda_q(T)$ *is the Lebesgue constant and* $|T|$ *is the volume of the canonical tetrahedron* ($= 1/6$ *for a right unit tetrahedron).*

Proof. Let $P_q(T)$ be the space of polynomials of degree q or less on T, then [41]

$$\|\Upsilon - \Upsilon^*\|_{\infty,T} \leq (C_2 + \lambda_q(T)) \inf_{\rho \in P_q(T)} \left\{ \sup_{\boldsymbol{\xi} \in T} |\rho(\boldsymbol{\xi}) - \Upsilon(\boldsymbol{\xi})| \right\} \tag{2.30}$$

Bounding the least deviation by an interpolation error yields (2.29) [33]. □

Let us apply this result to a single term in the strain energy.

Theorem 2. *Assume (2.26) holds, interpolation is performed at the points of Chen and Babuška [41], and* $\Upsilon \in C^\infty(T)$*, then there exists a constant* $C_1 > 0$ *such that*

$$\left| \int_T \Theta(\Upsilon - \Upsilon^*)\, d|\boldsymbol{\xi}| \right| \leq C_1(C_2 + q)\frac{|T|^{q-2}}{(q/3)!}\|V\|_{p+1,\Omega_e}\|W\|_{1,\Omega_e}\|\Upsilon\|_{q,\Omega_e}h_e^q, \qquad q > 2. \tag{2.31}$$

Proof. Adding the squares of (2.26) with $s = 0, 1$ yields

$$\|v\|_{1,T} \leq C \left\{ \inf_{\boldsymbol{\xi}\in T} J(\boldsymbol{\xi}) \right\}^{-\frac{1}{2}} \|v\|_{1,\Omega_e}. \tag{2.32}$$

Let N_i and N_j in (2.4b) be designated as V and W for consistency with (2.27) and use the generalized Hölder inequality and (2.29) to obtain

$$\left| \int_T \Theta(\Upsilon - \Upsilon^*)\, d|\boldsymbol{\xi}| \right| \leq C_1(C_2 + \lambda_q(T))\frac{|T|^{q-2}}{(q/3)!}\|V\|_{1,T}\|W\|_{1,T}|\Upsilon|_{q,T}. \tag{2.33}$$

Using the regularity conditions (2.26) and (2.32) yields

$$\left| \int_T \Theta(\Upsilon - \Upsilon^*)\, d|\boldsymbol{\xi}| \right| \leq C_1(C_2 + \lambda_q(T))\frac{|T|^{q-2}}{(q/3)!}\|V\|_{1,\Omega_e}\|W\|_{1,\Omega_e}\|\Upsilon\|_{q,\Omega_e}h_e^q. \tag{2.34}$$

The Chen and Babuška [41] interpolation points satisfy $\lambda_q(T) = O(q)$. This with replacement of $\|V\|_{1,\Omega_e}$ by $\|V\|_{p+1,\Omega_e}$ gives (2.31). □

Corollary 1. *Under the conditions of Theorem 2 then (2.27) is satisfied with* $q = p$*,* $p > 2$*.*

Proof. The energy difference $|A_e(V, W) - A_{*,e}(V, W)|$ is a linear combination of terms proportional to the left side of (2.31). Since the term

$$(C_2 + q)\frac{|T|^{q-2}}{(q/3)!}\|\Upsilon\|_{q,\Omega_e} \leq C \tag{2.35}$$

for fixed q, then (2.31) is bounded by $O(h_e^q)$ terms. Thus, perturbations due to quadrature are $O(h^p)$ if $q = p$. □

Thus, we have proved that the table look-up method will converge optimally at the same rate as the finite element method if the order of interpolation of the non-polynomial term in the integrand is the same as the order of interpolation of the finite element solution.

2.6 Numerical Examples

Our numerical examples involve solutions of the Helmholtz equation (2.1) subject to the Neumann data

$$\frac{\partial u}{\partial r} = \frac{\partial}{\partial r}\left(\frac{e^{i\kappa r}}{r}\right), \qquad \mathbf{x} \in \partial\Omega, \tag{2.36}$$

where r is a spherical distance. Problems are solved on various three-dimensional domains Ω centered at the origin having planar and curvilinear boundaries. With the specified boundary conditions, the solution of the boundary value problem (2.1, 2.36) is

$$u = \frac{e^{i\kappa r}}{r} \tag{2.37}$$

regardless of the shape of Ω.

Results are obtained with wave number $\kappa = 3$ by uniform p-refinement of an initial mesh using integration by table look-up, symmetric Gaussian quadrature, and GPR. Results with symmetric Gaussian quadrature are only presented for $p \leq 4$ since higher-order methods are unavailable. The integrals leading to the stiffness and mass matrices for this problem are of the form (2.3) with $\alpha = \beta$ and $c(\boldsymbol{x})$ constant. The integrals are computed to the minimum order of Gauss quadrature or table look-up interpolation to maintain optimal convergence as discussed in §2.5. The times to compute $\mathbf{K}^e$ and $\mathbf{M}^e$ are compared for the various quadrature rules. Since the cost of doing the polynomial interpolation of $\Upsilon(\boldsymbol{\xi})$ terms is $O(1)$ for domains with planar boundaries, it demonstrates the savings that can be realized using the table look-up method in elemental computations for elements that do not have any curvilinear boundary entities.

Example 1. We solve (2.1, 2.36) with Ω an octant of the region between two concentric cubes centered at the origin and having edge lengths one and two. The

mesh describing Ω has 1261 elements, all having planar faces.

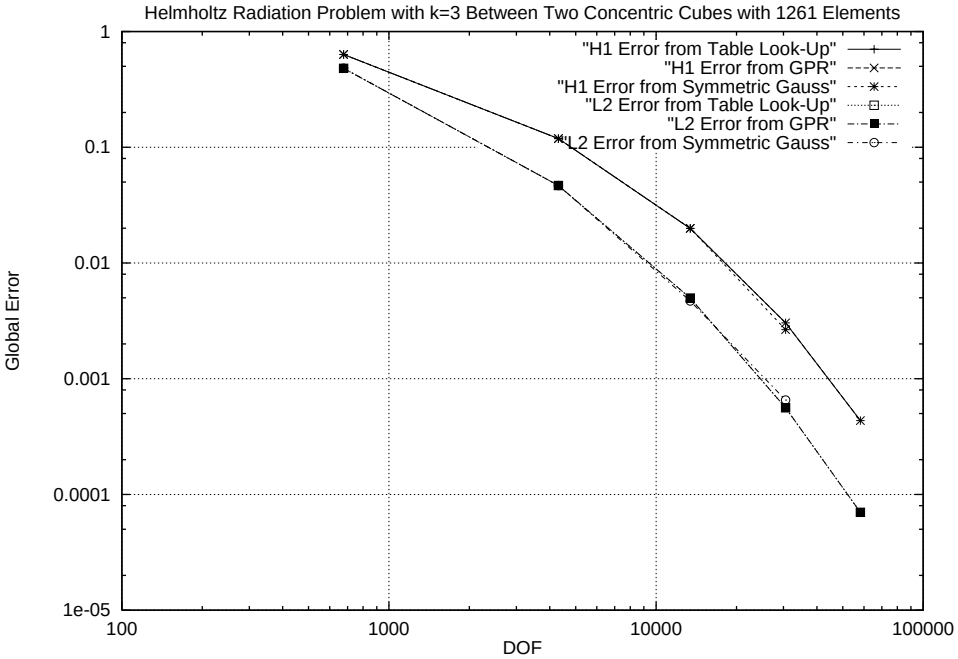

Figure 2.3: Global errors in L^2 and H^1 as a function of DOF for Example 1.

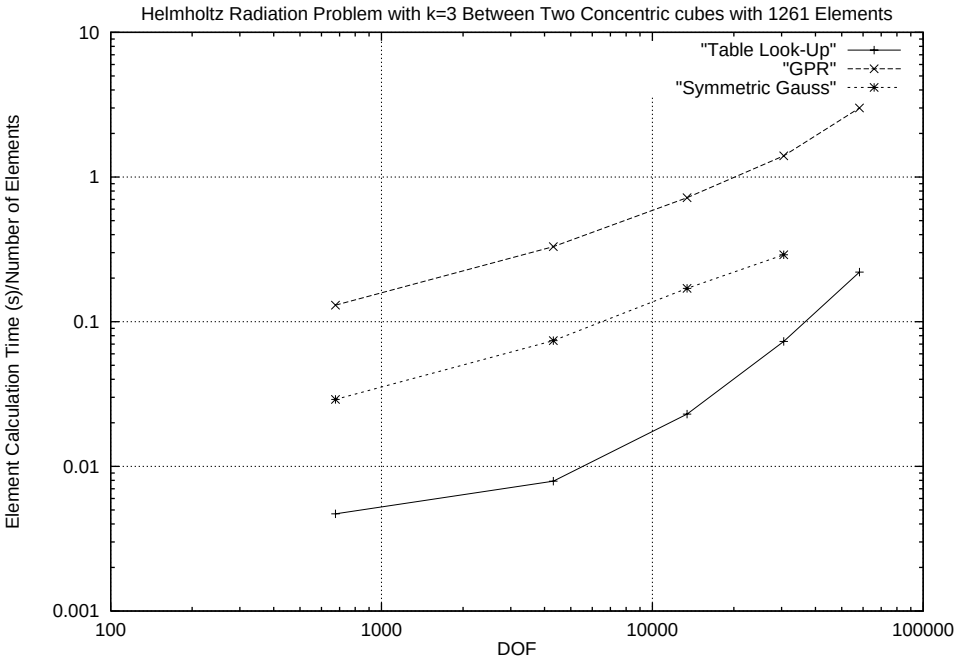

Figure 2.4: Time to evaluate elemental integrals as a function of DOF for Example 1.

In Figures 2.3 and 2.4 we present the global error and the computational time as a function of the degrees of freedom (DOF) for computations performed with p ranging from 1 to 5. All integration methods are exact, so the finite element solution errors are decreasing at an exponential rate with p (Figure 2.3). Results with table look-up (with $q = 0$ for this example) are between five and ten times faster than symmetric Gaussian quadrature and between ten and thirty seven times faster than GPR. Even though the table look-up method has a running time of $O(1)$, the time per element increases with order in Figure 2.4 since the number of integrals calculated per element increases.

Example 2. We solve (2.1, 2.36) with Ω being the upper half of a $\pi/6$ radian sector between two concentric spheres of radius one and three centered at the origin.

From Figure 2.5, the table look-up method is competitive with the other methods. The lack of an exponential convergence rate at higher orders is explained in the next section.

Results in Figure 2.6, likewise, indicate the superiority of the table look-up method at higher orders. Symmetric Gaussian quadrature was faster at $p = 4$ but higher-order formulas are unavailable. For methods which are available at higher order, clearly, the table look-up method is faster. While the time the table look-up method takes seems to be approaching the GPR, it should be noted that some software speedups, such as efficient cache utilization and more efficient solutions of the linear algebraic systems, have not been implemented in the table look-up method; thus, resulting in somewhat slow timings for larger orders. The GPR procedures have, however, been optimized. Nevertheless, the potential of this method should be clear for higher orders. The results show the theory presented in §2.5 holds even for problems where $A_*(U, U)$ is not coercive. However, we mention in passing that we have obtained similar results for the Poisson equation for which the theory holds.

There is a lack of optimal convergence at higher orders in Figure 2.5. This can be explained for Gauss quadrature by the following theorem.

Theorem 3. *Let $I_n(f)$ be the n-th order quadrature interpolant of f in a d-dimensional*

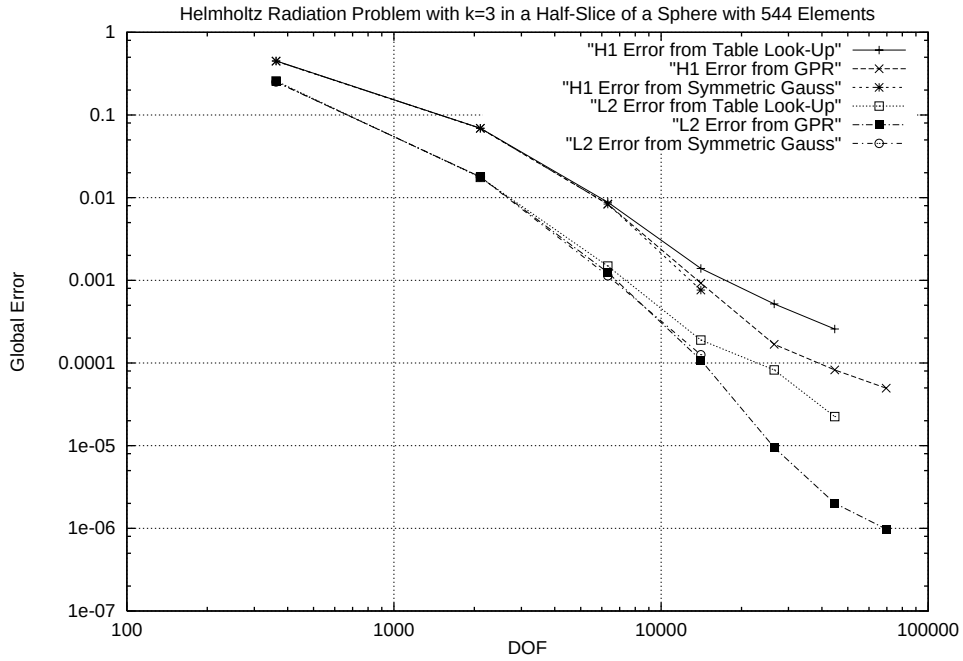

Figure 2.5: Global errors in L^2 and H^1 as a function of DOF for Example 2.

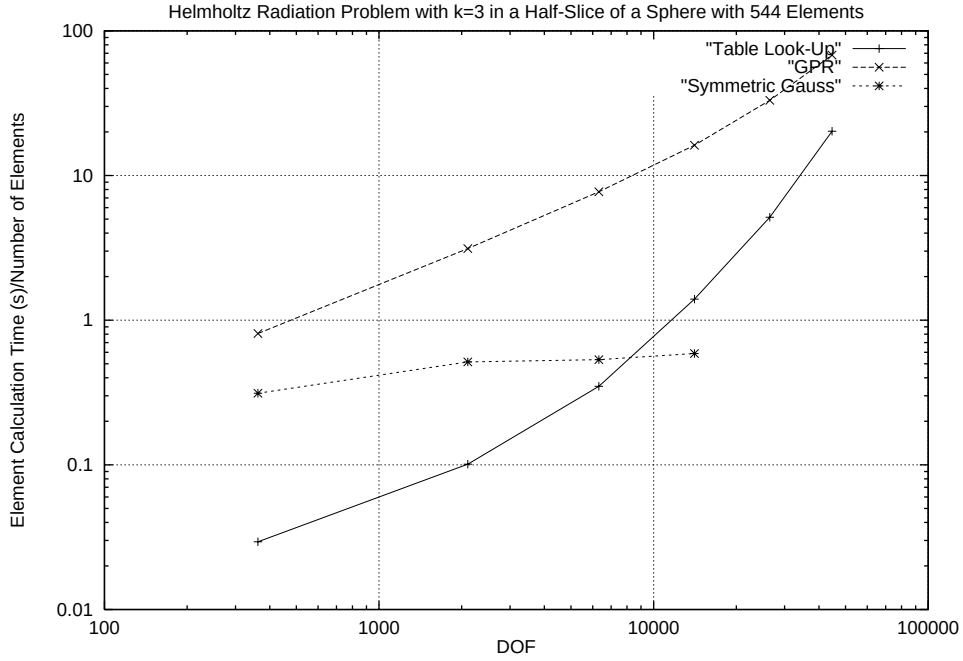

Figure 2.6: Time to evaluate elemental integrals as a function of DOF for Example 2.

domain Ω_e. *Then*

$$\left| \int_{\Omega} f(x) d\Omega_e - I_n(f) \right| \leq \frac{C}{(n+1)!} \|f\|_{\infty,\Omega_e}, \tag{2.38}$$

where C *contains a factor of* $\max\left\{|\Omega_e|, |\Omega_e|^{n+1}\right\}$.

Proof. cf. [83] □

The lack of optimal convergence at higher orders for the table look-up method can be explained by (2.35). In both cases, (2.38) and (2.35), the error is bounded by the n-th derivative of the integrand or Υ. If the maximum of the n-th derivative increases faster than $1/(n+1)!$, then the error does not decrease with increasing integration or interpolation order. Plotting the absolute value of the maximum of

$$\frac{\partial^n J}{\partial r^n}/(n+1)! \tag{2.39}$$

versus n in Figure 2.7, we show that the ratio is increasing thus explaining the lack of optimal convergence.

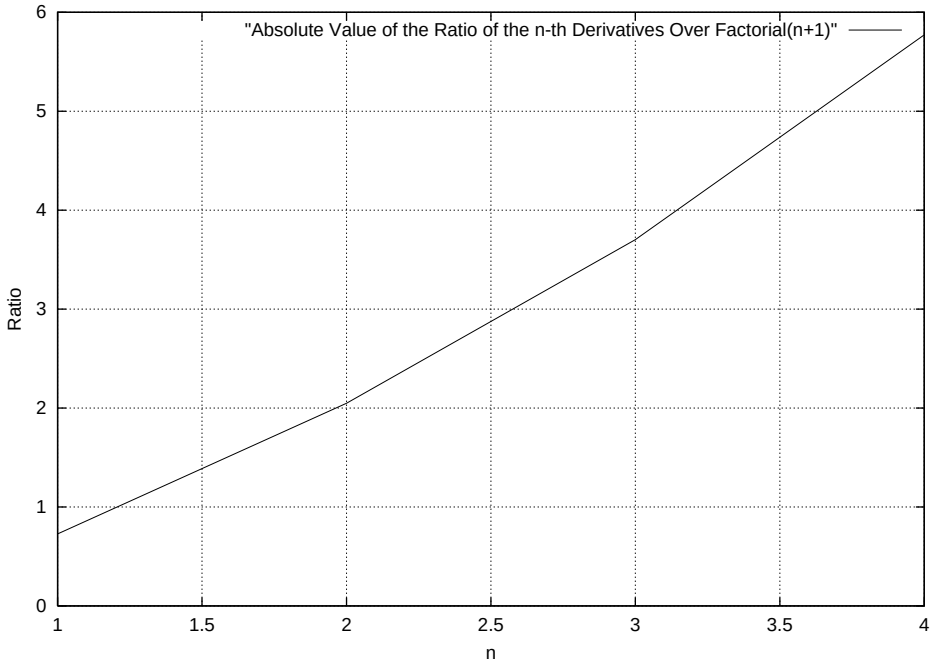

Figure 2.7: Ratio of the nth derivative of the Jacobian to $(n+1)!$.

2.6.1 Analysis of the Numerical Examples

In our two numerical examples, we have shown two cases where the table look-up method compared favorably with traditional methods. The examples with planar elements are the most favorable for table look-up, which produces the exact result in $O(1)$ time.

The worse-case problems for table look-up occur when interpolating a term that has an exponentially increasing derivative. Even in this case, table look-up is competitive with other methods. Since Gauss quadrature rules interpolate the entire integrand, the polynomial that multiplies the non-polynomial term reduces the rate at which the derivative of the interpolant grows.

Thus, table look-up is ideal for planar elements or $\Upsilon(\boldsymbol{\xi})$'s derivatives grow slowly. In meshes with a majority of elements in the interior, most of the elements are planar. Thus, especially for such problems, the table look-up method is the clear choice of integration method.

2.7 Discussion

A new technique for effective numerical computation of the element level integrals was presented. It is based on approximating the non-polynomial part of the integrand by polynomials followed by use of precomputed values of the resulting integrals. The table look-up method is well suited for computation on curved domains compared to other methods since it is more efficient, more flexible, and has fewer limitations on the maximum order of the method available. The Helmholtz equation showed the validity of these assertions.

While using optimal interpolation points [42] limits one to $q = 8$, table look-up may be used with other non-optimal interpolation points (such as equidistributed) which do not have an upper limit on their order, although the error may be greater, due to a larger Lebesgue constant. Thus, using non-optimal interpolation points, one may consider any order of integration using the table look-up method.

The number of precomputed integral entries required to be stored for various element level integrals can be reduced by investigating additional symmetries in the shape functions and their derivatives, if they exist. The efficiency of using

polynomial integrand approximation followed by use of precomputed values for element level integrals needs to be evaluated for three-dimensional element topologies besides tetrahedron, such as in spheres for meshless methods [32].

CHAPTER 3
APPLICATIONS IN TWO DIMENSIONS

3.1 Introduction

In this chapter, three different problems are solved using a prototype PDESE code written by Aiffa and the author. The code is discussed in §3.2.3. The code falls short in several categories of being a PDESE:

- Problems are very hard to code correctly,
- No warm restarts of DASPK, so it is not the most efficient solver,
- New methods are difficult to incorporate, and
- It handles problems only in two dimensions.

Nevertheless, with some modifications, the prototype PDESE was successful in solving adaptively many problems of interest, such as those in §3.2 through §3.4. Due to the adaptive procedures, it is shown that the prototype PDESE is efficient relative to more traditional methods. In §4, a more modern code, Trellis, will be shown to have addressed the above shortcomings.

3.2 Adaptive Finite Element Analysis of the Anisotropic Biphasic Theory of Tissue-Equivalent Mechanics

3.2.1 Introduction

The mechanical interaction of motile cells with collagen fibers, present in bones, tendons, and other connective tissue (used generally for both small fibers (fibrils) and fibers) in the surrounding extracellular matrix (ECM), is fundamental to cell behavior in soft tissues and tissue equivalent (TE) systems (highly entangled networks of collagen fibers and, thus, to many biomedical problems and tissue engineering applications [125, 95, 79, 93, 77, 68, 138]). TE systems may be configured in

a laboratory. The anisotropic biphasic theory (ABT) describes the mechanical interactions between cells and the ECM [21]. It accounts for the biomechanical feedback loop in which cells deform the surrounding network of ECM fibers, which induces network alignment with inhomogeneous deformation and cell alignment (contact guidance). Cell alignments may be enhanced by the application of a magnetic field as shown in Figures 3.1 and 3.2 [133].

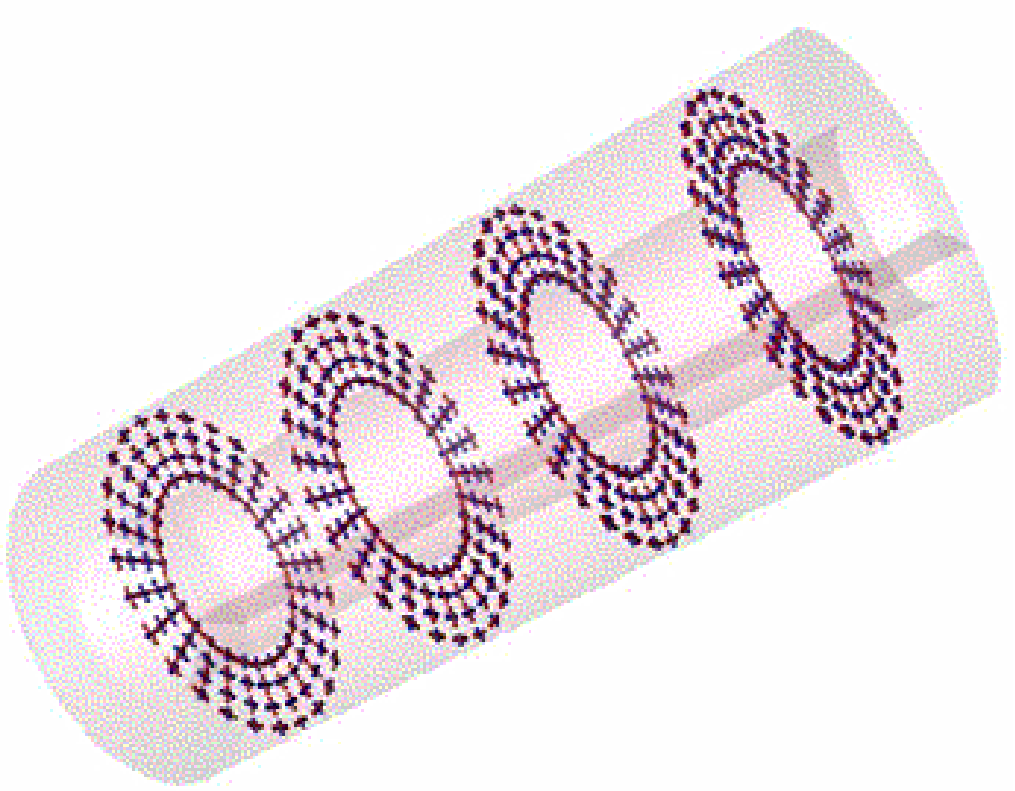

Figure 3.1: Initially cells are aligned by a magnetic field.

Understanding TE mechanics is central to the design and fabrication of bioartificial tissues, including skin [94] and artificial arteries [93, 133]. Previously, the ABT equations were solved using non-adaptive methods [133]. In this investigation, we solve the system of nonlinear partial differential equations governing the ABT theory in §3.2.2 by an adaptive finite element software system described in §3.2.3 [9]. To be more specific, solutions are obtained within a method-of-lines framework using finite element techniques in space and backward difference formula (BDF)

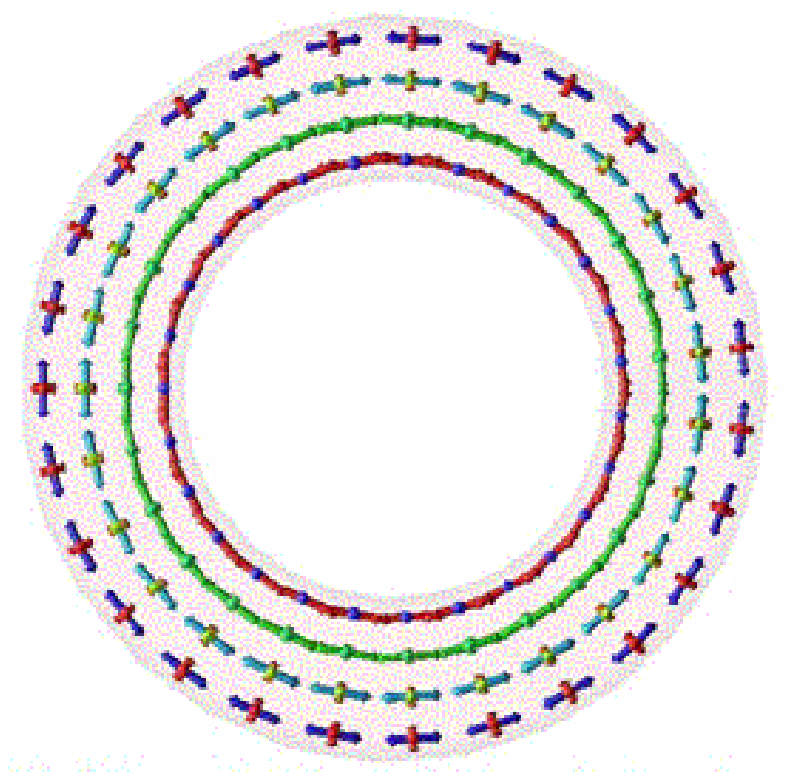

Figure 3.2: At $t = 36$ hours, there is enhanced alignment.

software in time. The spatial mesh is adaptively refined, coarsened, and relocated (hr-refinement) in response to an error indicator covered in §3.2.3.4 and material deformation. With this technology, we show that solutions of the ABT system and, hopefully, other bio-chemical systems can be obtained with greater efficiency and reliability.

The adaptive finite element software has been applied to three axisymmetric ABT problems in order to demonstrate its advantages relative to more traditional solution techniques in §3.2.4. Problems involve a cylindrical isometric cell traction assay (ICTA), covered in §3.2.4.1 [22], a model bioartificial artery in §3.2.4.2 [133, 93], and a model bioartificial artery with reinforced ends in §3.2.4.3. Each problem undergoes extensive material distortion which is accurately tracked by mesh relocation. Two of the problems also have singularities due to discontinuities in the boundary data and the geometry (corners). Sharp solution gradients near these singularities are accurately resolved by automatically placing a graded mesh within

these regions. In the cylindrical ICTA example, the adaptive method produced a solution using only 16% of the degrees of freedom required by a uniform-mesh solution having comparable accuracy.

3.2.2 Anisotropic Biphasic Theory of Tissue-Equivalent Mechanics

The entrapment of cells in a collagen gel to create an *in vitro* model TE was pioneered in the 1970s [31] and quickly emerged as an efficient and easily controllable method for studying cell behavior. TEs have been used extensively to study the contractile behavior of fibroblasts (cells in which fibrous tissue is formed) [64, 72, 98] because the volume change of the TE (a cell-induced contraction accompanied by the separating out of the fluid) provides a quantifiable measure of cell contractility. Figure 3.2.2 shows an example of the contractile behavior in the ICTA case which is discussed in §3.2.4.1.

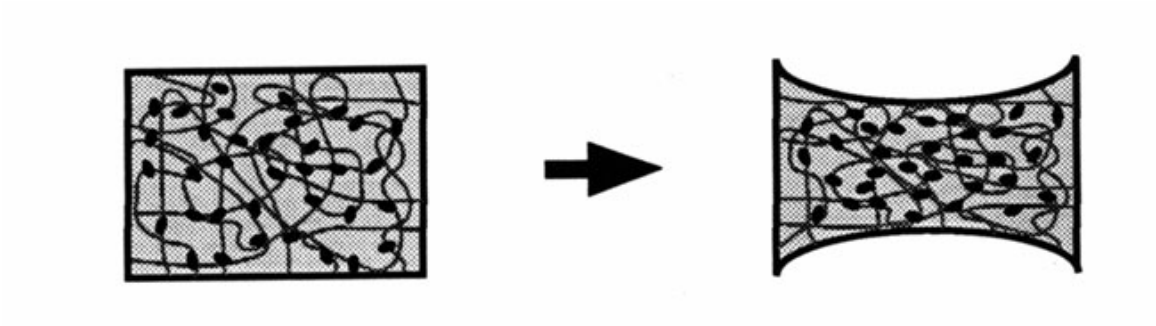

Figure 3.3: Illustration of contractile behavior in ICTA.

In order to facilitate the rational interpretation of *in vitro* contraction experiments and the design of artificial tissues, Barocas and Tranquillo [21] developed an ABT to describe the dynamic behavior of TEs that accounts for interaction between

- cell growth and migration;
- the biphasic nature of the TE, containing a collagen phase and an interstitial water phase;
- contractile stress exerted by the entrapped cells, driving the compaction of the TE [98];

- viscoelastic fluid behavior of the collagen matrix [19, 87];
- deformation-induced anisotropy in the collagen matrix; and
- contact guidance (the preferential alignment of cells in the characteristic direction of matrix alignment, which in turn leads to anisotropic cell migration and contractile stress [57]).

The mathematical statement of the model [21] is:

$$\frac{1}{2G}\dot{\boldsymbol{\sigma}} + \frac{1}{2\mu}\boldsymbol{\sigma} = \frac{1}{2}\left[\nabla\mathbf{v} + (\nabla\mathbf{v})^T\right] + \frac{\nu}{1-2\nu}(\nabla\cdot\mathbf{v})\mathbf{I}, \tag{3.1}$$

$$\nabla\cdot[\theta(\boldsymbol{\sigma} + \tau c\boldsymbol{\Omega}_c) - P\mathbf{I}] = \mathbf{0}, \tag{3.2}$$

$$\frac{Dc}{Dt} + c(\nabla\cdot\mathbf{v}) = \nabla\cdot(\mathcal{D}_0\boldsymbol{\Omega}_c\cdot\nabla c) + k_0 c, \tag{3.3}$$

$$\frac{D\theta}{Dt} + \theta(\nabla\cdot\mathbf{v}) = 0, \tag{3.4}$$

and

$$-\nabla\cdot\left[\frac{(1-\theta)}{\theta}\nabla P\right] + \varphi_0\nabla\cdot\mathbf{v} = 0, \tag{3.5}$$

where

$$\dot{\boldsymbol{\sigma}} \equiv \frac{D\boldsymbol{\sigma}}{Dt} - \nabla\mathbf{v}\cdot\boldsymbol{\sigma} - \boldsymbol{\sigma}\cdot(\nabla\mathbf{v})^T. \tag{3.6}$$

Here, D/Dt denotes a material derivative, $\mathbf{I}$ denotes the identity tensor, θ denotes the collagen network volume fraction, $\mathbf{v}$ denotes the network velocity, c denotes cell concentration, $\boldsymbol{\sigma}$ denotes the stress tensor, and P denotes pressure. Material properties associated with the collagen network are the modulus G, the viscosity μ, Poisson's ratio ν, and the interstitial flow resistance φ_0. Cell parameters are the migration (diffusivity) $\mathcal{D}_0$, the growth rate constant k_0, the cell orientation tensor $\boldsymbol{\Omega}_c$ [21], and the traction parameter τ

$$\tau \equiv \tau_0 \cdot \frac{t^{X_A}}{t^{X_A} + {X_B}^{X_A}} \cdot \frac{1}{1+\lambda c^2}, \tag{3.7}$$

where τ_0, X_A, X_B, and λ are constants. The cell orientation tensor is a complicated function of the deformation tensor [20]. To define the deformation tensor, first note that Ω changes in time. For a point $\mathbf{x}$ which starts at $\mathbf{x}_0$ at $t = 0$, let $\mathbf{U}$ be defined as $\mathbf{x} = \mathbf{x}_0 + \mathbf{U}$. Then define

$$\mathbf{D} = \frac{\partial \mathbf{x}}{\partial \mathbf{x}_0} = \mathbf{I} + \frac{\partial \mathbf{U}}{\partial \mathbf{x}_0}, \tag{3.8}$$

where the derivative with respect to $\mathbf{x}_0$ similar to the definition of gradient of a vector (A.2) except the derivative is respect to $\mathbf{x}_0$ instead of $\mathbf{x}$, and $\mathbf{I}$ is the identity matrix. The deformation tensor, $\mathbf{B}$, is then defined as $\mathbf{B} = \mathbf{D}^{-1}$.

Equation (3.1) with (3.6) is the linear viscoelastic constitutive equation of the collagen matrix, driving the compaction. Equation (3.2) preserves momentum balance. Equation (3.3) regulates cell concentration which changes by convection, diffusion, and growth. Equation (3.4) models the network volume which changes only by convection. Finally, equation (3.5) gives the pressure diffusion relation which balances the pressure and drag within the tissue on the network.

The first fraction in (3.7) accounts for the lag in cell traction stress due to the time needed to adjust to the change from suspension to tissue-equivalent. The cell traction force during this period has been correlated with cell spreading [19]. The second fraction accounts for so-called contact inhibition, the reduction in cell traction observed at high cell densities [98, 107].

The viscoelastic fluid momentum equation (3.5) cannot be eliminated by substitution into the stress balance equation (3.2); thus, unlike previous work involving elastic solid [99] or Newtonian fluid [51] models, the stress remains as a dependent variable.

A variational form of (3.1)-(3.5) was developed and solved by a mixed finite element method with piecewise biquadratic velocities and concentrations, piecewise-bilinear pressures, and discontinuous piecewise-biquadratic stresses [22].

The problems studied in this section involve axisymmetric three-dimensional motion, as shown in Figure 3.4 [20].

Thus, standard cylindrical coordinates [34] are used. The unit vectors are denoted as $\hat{\mathbf{r}}$, $\hat{\theta}$, and $\hat{\mathbf{z}}$. The variational problem is constructed by multiplying (3.1)-

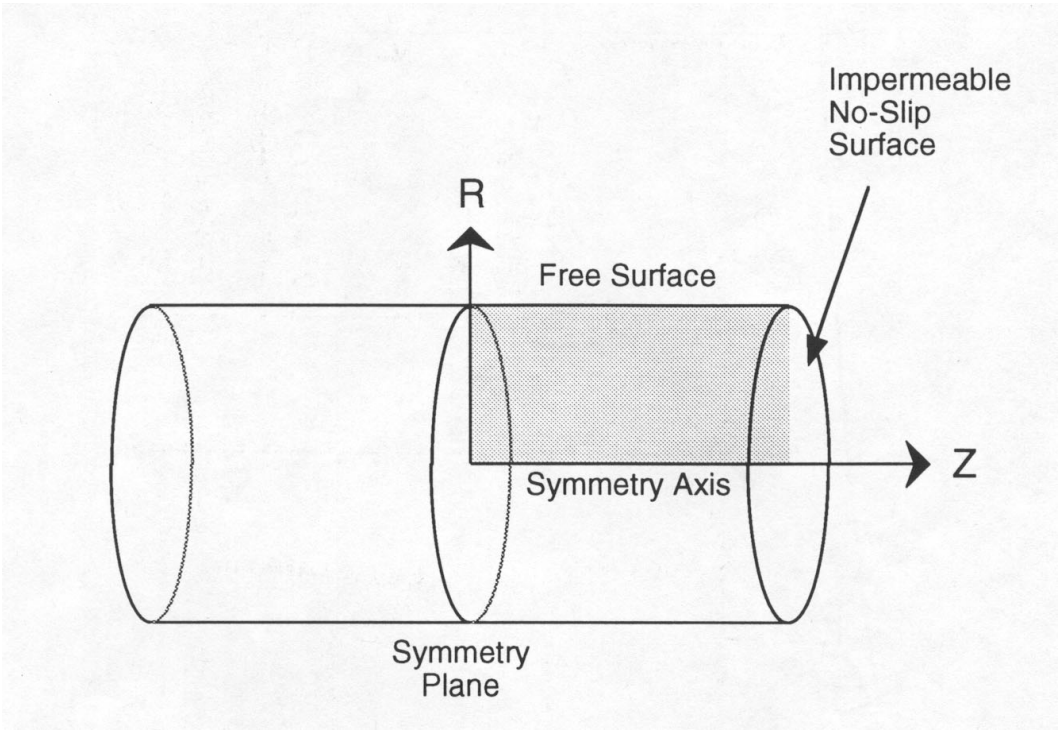

Figure 3.4: Axial symmetry of ICTA problems.

(3.5) by a test function, using integration by parts, and then integrating over Ω, to obtain

$$\left(\frac{D\theta}{Dt}, \phi_\theta\right) + (\theta(\nabla \cdot \mathbf{v}), \phi_\theta) = 0, \tag{3.9}$$

$$\left(\frac{Dc}{Dt}, \phi_c\right) + (c(\nabla \cdot \mathbf{v}), \phi_c) - \langle D_0 \boldsymbol{\Omega}_\mathbf{c} \cdot \nabla c, \phi_c \rangle + (D_0 \boldsymbol{\Omega}_\mathbf{c} \cdot \nabla c, \nabla \phi_c) - (k_0 c, \phi_c) = 0 \tag{3.10}$$

$$\langle \theta(\sigma + \tau_0 c \boldsymbol{\Omega}_\mathbf{c}), \phi_\mathbf{v} \rangle - (\theta(\sigma + \tau_0 c \boldsymbol{\Omega}_\mathbf{c}), \nabla \phi_\mathbf{v}) - (\nabla P, \phi_\mathbf{v}) = \mathbf{0}, \tag{3.11}$$

$$-\left\langle \frac{(1-\theta)}{\theta} \nabla P, \phi_P \right\rangle + \left(\frac{(1-\theta)}{\theta} \nabla P, \nabla \phi_P\right) + \varphi_0 \left(\nabla \cdot \mathbf{v}, \phi_P\right) = 0, \tag{3.12}$$

$$\left(\frac{1}{G}\dot{\sigma}, \phi_\sigma\right) + \left(\frac{1}{\mu}\sigma, \phi_\sigma\right) - \frac{1}{2}\left(\nabla \mathbf{v} + (\nabla \mathbf{v})^T, \phi_\sigma\right) - \frac{\nu}{1-2\nu}\left((\nabla \cdot \mathbf{v})\mathbf{I}, \phi_\sigma\right) = \mathbf{0}, \tag{3.13}$$

where ϕ_X is the test function associated with variable X. In cylindrical coordinates, for $\mathbf{v} = [v_r, v_\vartheta, v_z]^T$ [34]

$$\nabla \cdot \mathbf{v} = \frac{\partial v_r}{\partial r} + \frac{v_r}{r} + \frac{1}{r}\frac{\partial v_\vartheta}{\partial \vartheta} + \frac{\partial v_z}{\partial z} \tag{3.14}$$

and

$$\nabla \mathbf{v} = \begin{pmatrix} \frac{\partial v_r}{\partial r} & r\frac{\partial}{\partial r}\left(\frac{v_\vartheta}{r}\right) & \frac{\partial v_z}{\partial r} \\ \frac{1}{r}\frac{\partial v_r}{\partial \vartheta} & \frac{1}{r}\frac{\partial v_\vartheta}{\partial \vartheta} + \frac{v_r}{r} & \frac{1}{r}\frac{\partial v_z}{\partial \vartheta} \\ \frac{\partial v_r}{\partial z} & \frac{\partial v_\vartheta}{\partial z} & \frac{\partial v_z}{\partial z} \end{pmatrix} \tag{3.15}$$

and for a scalar function g,

$$\nabla g = \frac{\partial g}{\partial r}\hat{\mathbf{r}} + \frac{1}{r}\frac{\partial g}{\partial \vartheta}\hat{\vartheta} + \frac{\partial g}{\partial z}\hat{\mathbf{z}}. \tag{3.16}$$

With cylindrical symmetry, the stress tensor, σ, and cell orientation tensor, $\mathbf{\Omega}_c$, reduce to

$$\sigma \equiv \begin{pmatrix} \sigma_{rr} & 0 & \sigma_{rz} \\ 0 & \sigma_{\vartheta\vartheta} & 0 \\ \sigma_{rz} & 0 & \sigma_{zz} \end{pmatrix} \tag{3.17}$$

and

$$\mathbf{\Omega}_c \equiv \begin{pmatrix} \Omega_0 & 0 & \Omega_3 \\ 0 & \Omega_1 & 0 \\ \Omega_3 & 0 & \Omega_2 \end{pmatrix}, \tag{3.18}$$

where Ω_i, $i = 0, 1, 2, 3$, denotes the non-zero entries of $\mathbf{\Omega}_c$. Then, (3.9)-(3.13) become integrals whose integrands are (3.19)-(3.27).

$$F_\theta = \phi_\theta r\dot{\theta} + \phi_\theta r\theta\frac{\partial v_r}{\partial r} + \phi_\theta\theta v_r + \phi_\theta r\theta\frac{\partial v_z}{\partial z}, \tag{3.19}$$

$$\begin{aligned} F_c = \phi_c r\dot{c} + D_0\frac{\partial \phi_c}{\partial r}r\frac{\partial c}{\partial r}\Omega_0 + \\ D_0\frac{\partial \phi_c}{\partial z}r\frac{\partial c}{\partial z}\Omega_2 + D_0\left(\frac{\partial \phi_c}{\partial r}r\frac{\partial c}{\partial z} + \frac{\partial \phi_c}{\partial z}r\frac{\partial c}{\partial r}\right)\Omega_3 - \\ k_0\phi_c rc + \phi_c cr\frac{\partial v_r}{\partial r} + \phi_c cv_r + \phi_c rc\frac{\partial v_z}{\partial z}, \end{aligned} \tag{3.20}$$

$$\begin{aligned} F_{v_r} = \frac{\partial \phi_{v_r}}{\partial r}r\sigma_{rr}\theta + \frac{\partial \phi_{v_r}}{\partial r}r\tau_0 c\theta\Omega_0 + \phi_{v_r}\tau_0 c\theta\Omega_1 + \\ \phi_{v_r}\sigma_{\vartheta\vartheta}\theta + \frac{\partial \phi_{v_r}}{\partial z}r\sigma_{rz}\theta + \frac{\partial \phi_{v_r}}{\partial z}r\tau_0 c\theta\Omega_3 + \phi_{v_r}r\frac{\partial P}{\partial r}, \end{aligned} \tag{3.21}$$

$$F_{v_z} = \frac{\partial \phi_{v_z}}{\partial r} r \sigma_{rz} \theta + \frac{\partial \phi_{v_z}}{\partial r} r \tau_0 c \theta \Omega_3 + \frac{\partial \phi_{v_z}}{\partial z} r \tau_0 c \theta \Omega_2 + \frac{\partial \phi_{v_z}}{\partial z} r \sigma_{zz} \theta + \phi_{v_z} r \frac{\partial P}{\partial z}, \quad (3.22)$$

$$F_p = \varphi_0 \phi_P r \frac{\partial v_r}{\partial r} + \varphi_0 \phi_P v_r + \varphi_0 \phi_P r \frac{\partial v_z}{\partial z} + \left(\frac{1-\theta}{\theta} \right) \frac{\partial \phi_P}{\partial r} r \frac{\partial P}{\partial r} + \left(\frac{1-\theta}{\theta} \right) \frac{\partial \phi_P}{\partial r} r \frac{\partial P}{\partial z}, \quad (3.23)$$

$$F_{\sigma_{rr}} = \phi_{\sigma_{rr}} \frac{1}{\mu} \sigma_{rr} r + \phi_{\sigma_{rr}} \frac{1}{G} \dot{\sigma}_{rr} r - \left(1 + \frac{\nu}{1-2\nu} \right) \phi_{\sigma_{rr}} r \frac{\partial v_r}{\partial r} - \frac{\nu}{1-2\nu} \phi_{\sigma_{rr}} v_r - \frac{\nu}{1-2\nu} \phi_{\sigma_{rr}} r \frac{\partial v_z}{\partial z} - 2\phi_{\sigma_{rr}} r \frac{1}{G} \sigma_{rr} \frac{\partial v_r}{\partial r} - 2\phi_{\sigma_{rr}} r \frac{1}{G} \sigma_{rz} \frac{\partial v_z}{\partial r}, \quad (3.24)$$

$$F_{\sigma_{\vartheta\vartheta}} = \phi_{\sigma_{\vartheta\vartheta}} r \frac{1}{\mu} \sigma_{\vartheta\vartheta} + \phi_{\sigma_{\vartheta\vartheta}} r \frac{1}{G} \dot{\sigma}_{\vartheta\vartheta} - \frac{\nu}{1-2\nu} \phi_{\sigma_{\vartheta\vartheta}} r \frac{\partial v_r}{\partial r} - \left(1 + \frac{\nu}{1-2\nu} \right) \phi_{\sigma_{\vartheta\vartheta}} v_r - \frac{\nu}{1-2\nu} \phi_{\sigma_{\vartheta\vartheta}} r \frac{\partial v_z}{\partial z} - 2\phi_{\sigma_{\vartheta\vartheta}} \frac{1}{G} \sigma_{\vartheta\vartheta} v_r, \quad (3.25)$$

$$F_{\sigma_{zz}} = \phi_{\sigma_{zz}} r \frac{1}{\mu} \sigma_{zz} + \phi_{\sigma_{zz}} r \frac{1}{G} \dot{\sigma}_{zz} - \frac{\nu}{1-2\nu} \phi_{\sigma_{zz}} r \frac{\partial v_r}{\partial r} - \frac{\nu}{1-2\nu} \phi_{\sigma_{zz}} v_r - \left(1 + \frac{\nu}{1-2\nu} \right) \phi_{\sigma_{zz}} r \frac{\partial v_z}{\partial z} - 2\phi_{\sigma_{zz}} r \frac{1}{G} \sigma_{zz} \frac{\partial v_z}{\partial z} - 2\phi_{\sigma_{zz}} r \frac{1}{G} \sigma_{rz} \frac{\partial v_r}{\partial z}, \quad (3.26)$$

$$F_{\sigma_{rz}} = \phi_{\sigma_{rz}} \frac{1}{\mu} \sigma_{rz} r + \phi_{\sigma_{rz}} \frac{1}{G} \dot{\sigma}_{rz} r - \frac{1}{2} \phi_{\sigma_{rz}} r \frac{\partial v_z}{\partial r} - \frac{1}{2} \phi_{\sigma_{rz}} r \frac{\partial v_r}{\partial z} - \phi_{\sigma_{rz}} r \frac{1}{G} \sigma_{rr} \frac{\partial v_r}{\partial z} - \phi_{\sigma_{rz}} r \frac{1}{G} \sigma_{rz} \frac{\partial v_r}{\partial r} - \phi_{\sigma_{rz}} r \frac{1}{G} \sigma_{rz} \frac{\partial v_z}{\partial z} - \phi_{\sigma_{rz}} r \frac{1}{G} \sigma_{zz} \frac{\partial v_z}{\partial r}. \quad (3.27)$$

For each F_X, $\int_\Omega F_X \, d\Omega$ must equal 0. The boundary integrals are omitted from (3.19)-(3.27) since the imposed Neumann boundary conditions are always zero (*cf.* §3.2.4).

Although this formulation was successfully applied to several boundary value problems [21], the associated software lacked the mesh generation and adaptivity capabilities necessary to handle sharp gradients and irregular configurations. This difficulty provides the motivation for the current study.

3.2.3 Adaptive Finite Element Software

Initial-boundary value problems for the axisymmetric ABT equations (3.1)-(3.5) are solved by an extended version of an adaptive finite element software system, originally written by Aiffa [9], that addresses two-dimensional transient partial differential systems of the form (1.2). The software is generic and specific problems are addressed by supplying procedures to evaluate the n-dimensional functions $\mathbf{a}$, $\mathbf{f}$, and $\mathbf{g}$; defining the initial and boundary conditions; describing the domain Ω; and generating an initial mesh. For the ABT system (3.1)-(3.5), $\mathbf{u}$ is the nine-dimensional vector

$$\mathbf{u} = [\sigma_{rr}, \sigma_{\phi\phi}, \sigma_{zz}, \sigma_{rz}, v_r, v_z, c, \theta, P]^T \tag{3.28}$$

where σ_{rr}, $\sigma_{\phi\phi}$, σ_{zz}, and σ_{rz} are the axisymmetric components of $\boldsymbol{\sigma}$, and v_r and v_z are components of $\mathbf{v}$.

A spatially-discrete Galerkin form of (1.2) is solved by a method-of-lines formulation (§3.2.3.3) with a hierarchical finite element basis (§3.2.3.2) of arbitrary degree. The software has capabilities to

- perform arbitrary combinations of adaptive mesh refinement and coarsening (h-refinement), order variation (p-refinement), and mesh motion (r-refinement) (in §3.2.3.5);
- manipulate meshes of triangular elements on complex two-dimensional regions using an octree spatial decomposition [119] (in §3.2.3.1);
- provide a high-order representation of Ω (in §3.2.3.1);

- handle combinations of Dirichlet, Neumann, and Robin boundary conditions for different solution components on stationary and moving boundaries; and
- address mixed finite element formulations.

Herein, only combinations of adaptive h- and r-refinement have been used. Similar analyses have been performed on the oxidation of ceramic-matrix composites [4] and ceramic fiber coating by chemical vapor deposition [8]. Aiffa [9] presents results using adaptive p- and hp-refinement.

3.2.3.1 Mesh Structures

Prior to solution, the domain Ω is discretized by an octree spatial decomposition [119] and partitioned into triangular elements Ω_e, $e = 1, 2, \ldots, N$, by an external mesh generator. Exact representation of curved regions is possible with elements having curved sides; thus, $\Omega = \cup_{e=1}^{N} \Omega_e$. Polygonal representations of boundaries are also possible for use with low-order finite element approximation. Even in this case, adaptive h-refinement is performed on the curved boundary rather than its polygonal approximation. With an octree decomposition, the mesh will naturally be finer near curved boundaries.

Maintaining an exact representation of Ω when its boundary evolves in time is no longer possible. In this case, the moving boundary is approximated by a piecewise polynomial of degree q. Interpolating a curved boundary to degree q introduces an $O(h^{q+1/2})$ error in the H^1-norm of the finite element solution [136], where h is the length of the longest edge in the mesh. Without such an approximation, the error of the finite element solution with a piecewise polynomial basis of degree p is $O(h^p)$ in the H^1-norm [33]. Thus, the choice $q = p$ ensures that solution accuracy is not diminished by the approximation of the boundary.

The mesh is stored in a data structure called the *SCOREC Mesh Database* [27] with data divided into *entities* that are element vertices, edges, and faces (finite elements in two dimensions). The representation is hierarchical with faces having pointers to their bounding edges which, in turn, have pointers to their bounding vertices. Entities of a given dimension are linked to facilitate traversals. Operators (functions) exist to find bounding entities, such as the vertices bounding a face,

and to create and delete elements. Solution and other data is stored with the mesh entities with a goal of facilitating the development and maintenance of finite element software, particularly when using p-refinement. Entities may be queried for the solution or mesh data that they contain. Mesh data may be altered for use with h and r-refinement. The operators that create and delete elements eliminate most of the complex programming effort associated with adaptive h-refinement (in §3.2.3.5). A more modern version of this software, AOMD, is discussed in §4.1.6.

3.2.3.2 Piecewise Polynomial Basis

The finite element basis is a modification of the classical [130] hierarchical basis on a triangle. Thus, polynomial shape functions of degree $p+1$ are obtained as corrections to shape functions of degree p. Elements of the basis are associated with vertices (for $p \geq 1$), edges (for $p \geq 2$), and faces (for $p \geq 3$); thus, complementing the structure of the SCOREC mesh database. The new basis [9, 5] differs from the classical [130] one in that face functions (internal modes) satisfy an orthogonality condition in strain energy. When applied to the Laplacian operator, the new basis reduces the growth of the condition number of the stiffness matrix from exponential to nearly linear in p [9, 5]. Similar savings were recorded on problems with more complex stiffness matrices [5] relative to other hierarchical bases [43]. These hierarchical bases can be constructed to arbitrarily high orders; thus, facilitating adaptive p-refinement and *a posteriori* error estimation through adaptive p-refinement (§3.2.3.4).

Mixed methods, where different components of $\mathbf{u}$ have different order approximations, are permitted. A traditional reason for using a mixed finite element method is the satisfaction of the Babuška-Brezzi stability condition [38]. For an incompressible medium, this condition requires that the approximation space of the pressure be a subspace of that used for the velocity. This is often achieved by using a lower-order basis for the pressure than for the velocity. The pressure space may further be selected relative to a patch of elements (macro-elements) used for the velocity or made discontinuous relative to a continuous velocity space [38]. Our software has no capabilities for macro-elements. While discontinuous spaces are possible, the

computations of §3.2.4 used a piecewise quadratic polynomial for pressure and a piecewise cubic for all other variables in $\mathbf{u}$. This choice produced solutions without the spurious oscillations associated with a failure to satisfy the Babuška-Brezzi condition.

Elemental integrals associated with Galerkin inner products are evaluated by Gaussian quadrature of order $2p$ for a basis of degree p [60]. If the element has a curved edge corresponding to a piecewise polynomial approximation of the boundary to order q (§3.2.3.1), then quadrature is performed to order $2(p+q-1)$. This prescription maintains quadrature errors at a higher order than finite element interpolation errors [47, 66, 126, 127].

3.2.3.3 Temporal Solution Techniques

Approximating (1.2) by the finite element-Galerkin technique in space gives rise to a large system of either ordinary differential equations (ODEs) or, when the Jacobian of $\mathbf{g}$ with respect to $\partial_t \mathbf{u}$ is singular, a system of differential-algebraic equations (DAEs), which must be solved in time. Solving this system using ODE/DAE software is known as the *method of lines* and it is the approach used by our adaptive software. Separating the spatial and temporal discretization has the advantage that time integration procedures may be changed without affecting the larger finite element portion of the software.

We use the variable-order, variable-time-step, backward-difference formula package DASPK [37] to solve the ODE/DAE system. Nonlinear problems are solved by a damped Newton's method, and the linear system introduced at each Newton step is solved by sparse Gaussian elimination with a minimum degree reordering [73]. After each successful time step, solution coefficients are stored with the geometric entities corresponding to their basis elements.

3.2.3.4 Error Indication

A posteriori error estimates provide a measure of solution accuracy and a means of guiding the adaptive enrichment process. The hierarchical basis (§3.2.3.2) supplies an inexpensive (relative to the solution cost) way of estimating spatial discretization errors as the correction terms of the next polynomial in the sequence.

Thus, if the finite element approximation has degree p, a spatial error estimate can be obtained from the hierarchical correction of degree $p+1$. The effectiveness of this strategy depends on several factors including mesh uniformity and structure, the PDE system, and the computational procedure used to determine the error [128, 135]. The technique performs best on diffusion-dominated problems. Procedures typically localize the error computation to an element or a patch of elements [135] to further reduce the cost of obtaining estimates. These local finite element problems are closed by prescribing fluxes on element or patch boundaries. Performance differs greatly with flux prescriptions [10, 128]. A strategy of this type is offered within our software with fluxes at element boundaries prescribed as the average numerical flux across an edge [9].

Local finite element problems for the spatial error estimate consist of contributions from the element or patch of elements and from the fluxes across the element or patch boundary. Error estimates have been shown to converge under h-refinement in exactly the same manner as the true error for linear elliptic [143, 144] and parabolic [6, 7] problems on uniform rectangular-element meshes. These results further demonstrate that the elemental contribution dominates the flux contribution of the error for even-order finite element computations, while the opposite is true for odd-order finite element solutions. This, and the uncertainty of estimating errors of nonlinear mixed problems such as (3.1)-(3.5), motivates us to adapt a simpler alternative. Instead of computing an error estimate, we compute an error indicator E_i on Ω_i as

$$E_i = E_i^1 + E_i^2 + E_i^3 \tag{3.29}$$

where

$$E_i^j = \left|\partial\Omega_i^j\right| \int_{\partial\Omega_i^j} \left| \frac{\partial C^+}{\partial \mathbf{n}} - \frac{\partial C^-}{\partial \mathbf{n}} \right| \, dS, \qquad j = 1, 2, 3. \tag{3.30}$$

Here, $\partial\Omega_i^j$ is the $j\ th$ edge of the boundary $\partial\Omega_i$ of Ω_i, $\mathbf{n}$ is the unit outward normal vector to $\partial\Omega_i$, E_i^j is the error indicator on edge j, $\left|\partial\Omega_i^j\right|$ is the length of $\partial\Omega_i^j$, $C(t, \mathbf{x})$ is the finite element approximation of the concentration $c(t, \mathbf{x})$, and superscripts $+$ and $-$ denote values on the exterior and interior of Ω_i. Thus, the error indicator on Ω_i is the absolute jump in the concentration gradient across the boundary of Ω_i

scaled by the length of the boundary. The concentration gradient was chosen to control adaptivity because c varies significantly (relative to other components of $\mathbf{u}$) and high gradients in $\mathbf{u}$ are reflected by a high gradient in c.

Theoretical results [6, 7] implying that the spatial error of odd-order finite element solutions are proportional to flux jumps are only available for rectangular-element meshes. Nevertheless, computational evidence [82] suggests that they hold on triangles. With the concentration approximated by a piecewise cubic polynomial, the error indicator (3.29)-(3.30) might be proportional to the true error; however, with the complexities of the ABT system, this remains unverified. Since we choose the concentration, which is approximated by a cubic polynomial, as the variable to use for spatial error estimation (*cf.* §3.2.3.2), (3.30) is applicable.

Control of local temporal errors is done by procedures within DASPK. Strategies for controlling global errors by maintaining the local temporal error at a small fraction of global spatial error have been described for stiff differential systems arising from the discretization of parabolic problems [92]. If the error indicator (3.29)-(3.30) were a true error estimate, this strategy might give an appraisal of the global discretization error; however, once again, the ABT system is too involved to characterize as either being stiff or parabolic.

3.2.3.5 Adaptivity

When choosing adaptive hr-refinement to solve the ABT system (3.1)-(3.5), our aims were accurate resolutions of moving boundaries by r-refinement and of sharp gradients by h-refinement. Mesh coarsening is needed in addition to refinement since high-gradient locations may move with time.

Adaptive r-refinement can concentrate a mesh within high-error regions and follow these regions as they evolve. This mesh motion is often far less expensive than h-refinement [3] for a given accuracy. With parallel computation, r-refinement avoids the need for dynamic load balancing which must be done with h-refinement. Unfortunately, two- and three-dimensional meshes tend to tangle with r-refinement and there is no effective way of controlling the discretization error.

R-refinement may be regarded as a dynamic coordinate transform and, as

such, the software permits mappings from the physical $\mathbf{x}$-plane to a computational $\boldsymbol{\xi}$-plane of the form

$$\mathbf{x} = \mathbf{x}(t, \boldsymbol{\xi}, \mathbf{u}). \tag{3.31}$$

Typically, only the vertices of the triangular elements are mapped by this transformation. Element edges remain straight, at least for those elements that are not on a moving curved boundary or interface. In this analysis, we follow a simpler course and move mesh vertices according to their material positions, *i.e.*,

$$\frac{d\mathbf{x_j}}{dt} = \mathbf{v}(t, \mathbf{x}_j), \qquad j = 1, 2, \ldots, N_V, \tag{3.32}$$

where $\mathbf{x}_j$ is the coordinate of vertex j and N_V is the number of vertices. Curved element sections on moving boundaries are tracked with greater precision. If, as described in §3.2.3.1, a curved element edge is approximated by a polynomial of degree q, then the $q + 1$ interpolation points describing the segment are moved according to (3.32). The ODE system (3.32) may either be coupled with the finite element system for integration by DASPK or, with less precision, solved separately in an explicit manner. The former course was taken in this application.

The temporal integration is halted after each time step, error indicators are calculated according to (3.30), and adaptive h-refinement is performed, if necessary. In particular, edge j of element i is marked for refinement when $E_i^j \geq \alpha_R \overline{E}$ and marked for coarsening when $E_i^j \leq \alpha_C \overline{E}$, where α_R and α_C are, prescribed refinement and coarsening tolerances, respectively, and $\overline{E}$ is the average of E_i^j over the edges of the mesh. Normally, error indicators would be checked after four to five time steps [9]; however, these error indicators are so inexpensive that checking after each time step incurs a negligible cost and provides additional reliability. Given the uncertainties of the initial mesh, backtracking is performed to ensure its adequacy. Thus, the initial time step is repeated until every edge of the mesh satisfies $E_i^j \in (\alpha_R, \alpha_C)\overline{E}$.

Elements having edges that are marked for refinement are refined by templates as shown in Figures 3.2.3.5 and 3.2.3.5. Dashed lines indicates the edges created by refinement. Marked edges scheduled for bisection are shown as darker lines. The

choice of the refinement that is performed when three edges are marked in Figure 3.2.3.5 is determined to minimize the difference in the angles of the resulting four triangles. This decision is based on a desire to avoid elements with small or large angles for accuracy considerations.

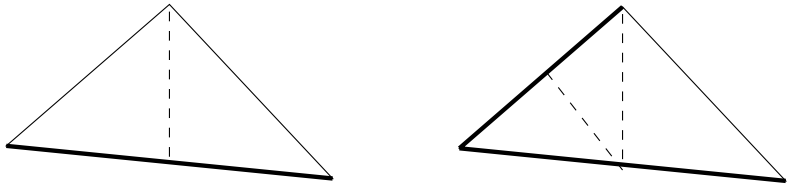

Figure 3.5: Refinement templates for a triangle with one (left) and two (right) marked edges.

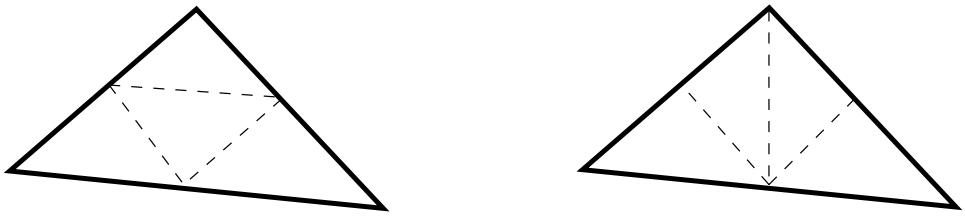

Figure 3.6: Refinement templates for a triangle with three marked edges.

Coarsening is done by edge collapsing, *i.e.*, one of the vertices bounding an edge is moved to the other to remove the edge. An edge marked for removal on a patch of elements is shown dark on the left of Figure 3.2.3.5. This edge is removed by collapsing the upper vertex to the lower one as shown on the right of Figure 3.2.3.5.

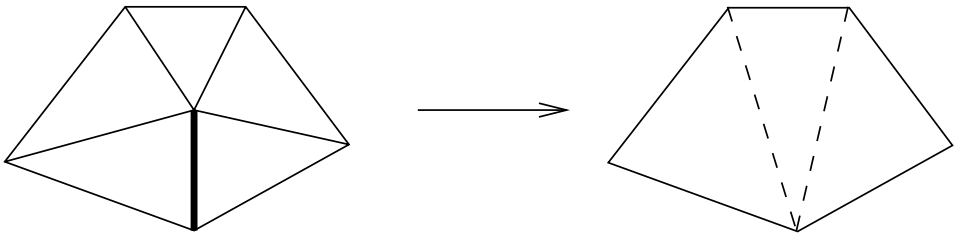

Figure 3.7: An edge (shown dark on the left) is removed by collapsing its upper vertex to the lower one (right).

The dashed lines indicate the new elements of the patch. Other complications arise, particularly with simultaneous h- and r-refinement. The major problem is

ensuring that the angles of elements remain bounded away from zero and π radians during enrichment. Failure to do so renders the coordinate transform between the physical and canonical elements singular. Techniques used to address these and other h-refinement issues are described elsewhere [9].

Finite element solutions must be available on the newly refined or coarsened elements before the time integration can continue and we obtain these by a local L^2 projection. Consider an element or patch of elements that have been affected by refinement (Figures 3.2.3.5 and 3.2.3.5) or coarsening (Figure 3.2.3.5) and construct a finite element basis relative to the new mesh. The finite element solution $\mathbf{U}^{new}$ is determined on the new element or patch Ω_{new} by solving the L^2 problem

$$\int_{\Omega_{new}} (\mathbf{V}^{new})^T [\mathbf{U}^{new} - \mathbf{U}] \, dV = 0, \qquad \forall \mathbf{V}^{new} \in S(\Omega_{new}), \tag{3.33}$$

where $\mathbf{U}$ is the finite element solution prior to h-refinement and $\mathbf{V}^{new}$ is a test function relative to the finite element space S^{new} on the new mesh. Structures within the SCOREC mesh database (§3.2.3.1) facilitate the solution of (3.33) since all solution and shape function data are stored with the appropriate mesh entities.

3.2.4 Simulations

We apply the adaptive finite element software to three initial-boundary value problems for the ABT system (3.1)-(3.5): an ICTA [22], a prototypical bioartificial artery (BAA) [93, 133], and a BAA with thicker ends. Collagen parameter values [21] and cell parameter values for the ICTA [22] and BAA [93, 133] problems were obtained from the literature.

The types of boundary conditions involved with each problem are:

- A *free surface* where the pressure, the normal component of the total stress, and the diffusive cell flux vanish, *i.e.*,

$$P = 0, \qquad (\boldsymbol{\sigma} + c\boldsymbol{\Omega}_c) \cdot \mathbf{n} = \mathbf{0}, \qquad (\boldsymbol{\Omega}_c \cdot \nabla c) \cdot \mathbf{n} = 0. \tag{3.34}$$

- A *plane of symmetry* or a *free-slip surface* where the tangential component of the total stress, the interstitial flow, the diffusive cell flux, and the velocity of

the network normal to the surface vanish, *i.e.*,

$$\mathbf{n}\cdot(\boldsymbol{\sigma}+c\boldsymbol{\Omega}_c)\cdot\mathbf{t}=\mathbf{0}, \qquad \nabla P\cdot\mathbf{n}=0, \qquad (\boldsymbol{\Omega}_c\cdot\nabla c)\cdot\mathbf{n}=0, \qquad \mathbf{v}\cdot\mathbf{n}=0, \tag{3.35}$$

where $\mathbf{t}$ is a unit tangent vector. A free-slip surface represents a surface that is impenetrable but does not adhere to the TE. It is functionally equivalent to a plane of symmetry.

- A *no-slip surface* has the impenetrability conditions of the free-slip surface, but the tangential component of the velocity vanishes instead of the tangential stress, *i.e.*,

$$\nabla P\cdot\mathbf{n}=0, \qquad (\boldsymbol{\Omega}_c\cdot\nabla c)\cdot\mathbf{n}=0, \qquad \mathbf{v}=0. \tag{3.36}$$

In all experiments, the initial solution vector $\mathbf{u}(0,\mathbf{x})$, $\mathbf{x}\in\Omega$, is zero except $c=\theta=1$. These conditions correspond to a stress-free state of equilibrium with the domain filled with a collagen network having a unit concentration and no solution phase volume.

3.2.4.1 Isometric Cell Traction Assay

The ICTA has been used in various forms [88] to measure the force generated by cells in a TE. ICTA problems have been solved on a slab [88]; however, we solve axisymmetric problems on the cylinder $\{(r,\phi,z)\mid 0<r<1,\ 0\le\phi<2\pi,\ 0<z<1\}$. Symmetry conditions (3.35) are imposed on $r=0$; $z=0$ is a plane of symmetry (3.35); $r=1$ is a free surface (3.34); and $z=1$ represents a no-slip surface (3.36). The parameters are chosen as $\tau_0=0.0455$, $X_A=1.6$, $X_B=0.21$, $\theta_0=0.003$, $G=\mu=1/2$, $\varphi_0=0.0193$, $D_0=0$, $k=0$, $\nu/(1-2\nu)=1/3$, and $\lambda=2.5$.

As the cells contract, the force to maintain a constant length increases, and the circumference at the midplane ($z=0$) decreases. The prior analysis of this problem [22], without adaptivity, had difficulty resolving sharp gradients where the free and free-slip surfaces meet. We show that the adaptive hr-refinement procedure will alleviate this.

Concentration and the axial stress contours are shown in Figures 3.8 and 3.9, respectively, at four times with the computational meshes superimposed. Here, and

in all similar illustrations, the radial (r) direction is shown horizontal and the axial (z) direction is vertical. There is a singularity, at the juncture of the no-slip and free-surface boundaries. This singularity strengthens as time increases. The mesh is concentrated in this region and graded to resolve the solution adequately. The axial stress obtained in a prior study [22] had an anomalous oscillatory behavior that disappeared with (uniform) mesh refinement. Axial stress components obtained by adaptive *hr*-refinement (Figure 3.9) have no spurious behavior, presumably because the singularity is being adequately resolved on the graded mesh.

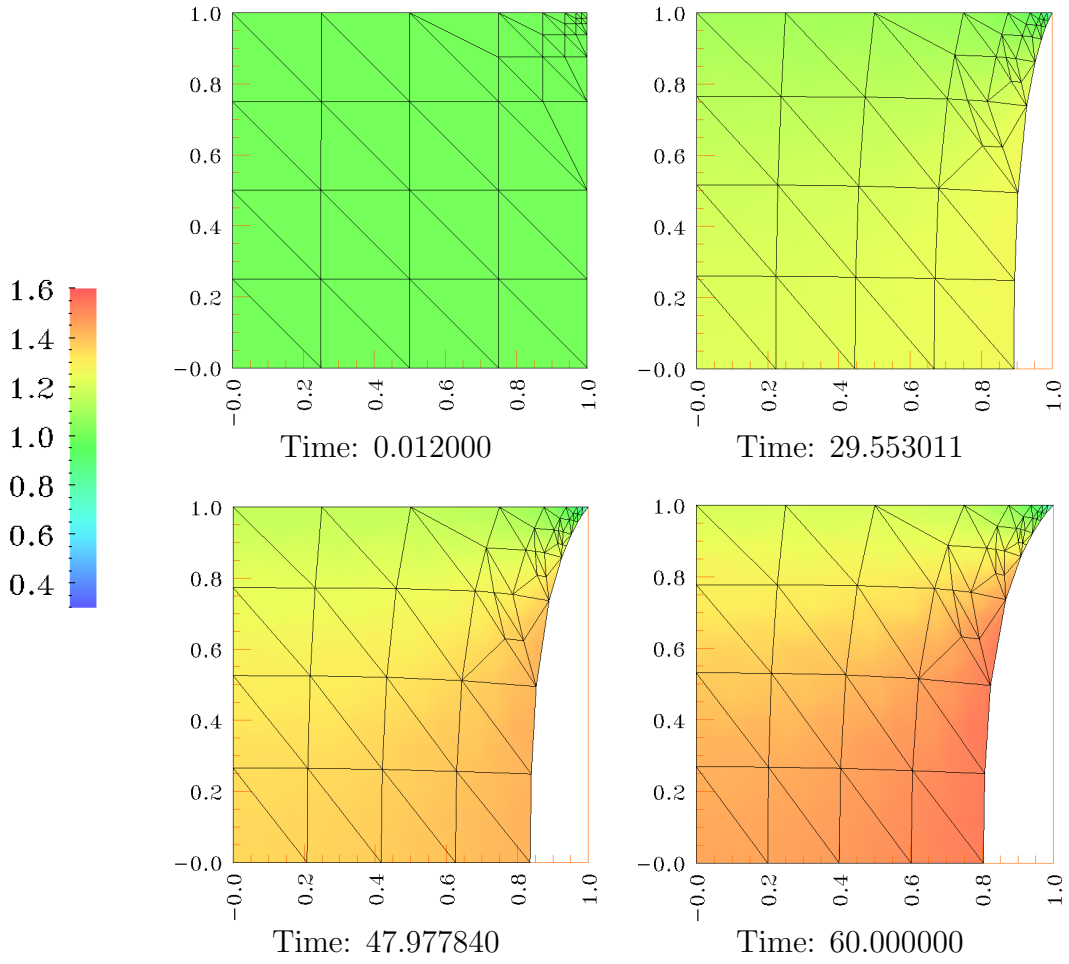

Figure 3.8: ICTA concentration at four times with adaptive meshes superimposed.

As an indication of the convergence rate, we generated a sequence of solutions

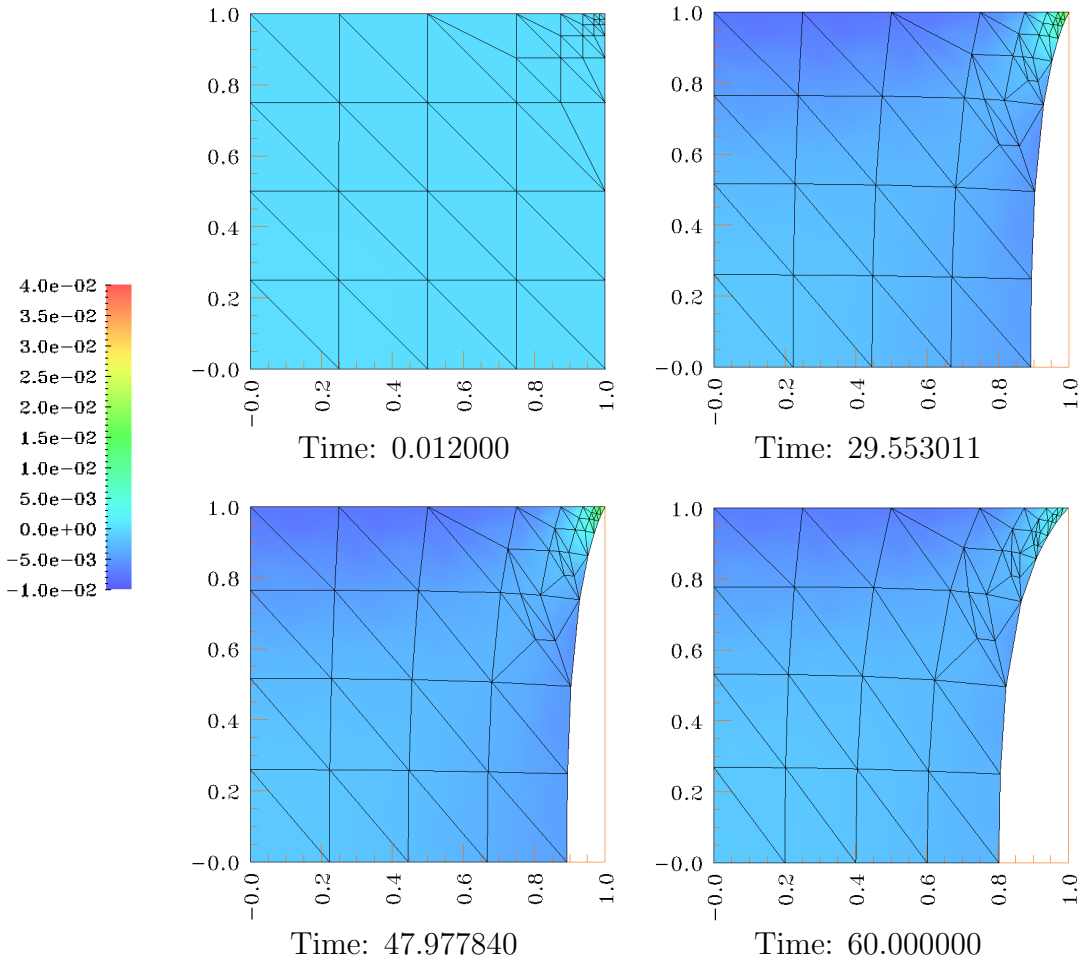

Figure 3.9: ICTA axial stress at four times with adaptive meshes superimposed.

on meshes obtained by uniform refinement of an initial mesh. In Figure 3.10, we show $\|c(t,\cdot)\|_1$ for these solutions at $t = 60$ as a function of the average mesh spacing h and the total degrees of freedom (DOF), which is the time integral of the spatial DOF. This sequence indicates that the concentration is converging at an approximate rate of $h^{1.2}$. This is less than the $O(h^2)$ rate for smooth problems because of the singularity at the intersection of the no-slip and free-slip surfaces. The adaptive computation of Figures 3.8 and 3.9 is indicated by a $\diamond$ on Figure 3.10. For comparable accuracy, the solution by adaptive *hr*-refinement required approximately 16% of the total DOF of that with a uniform mesh. The improvement is largely

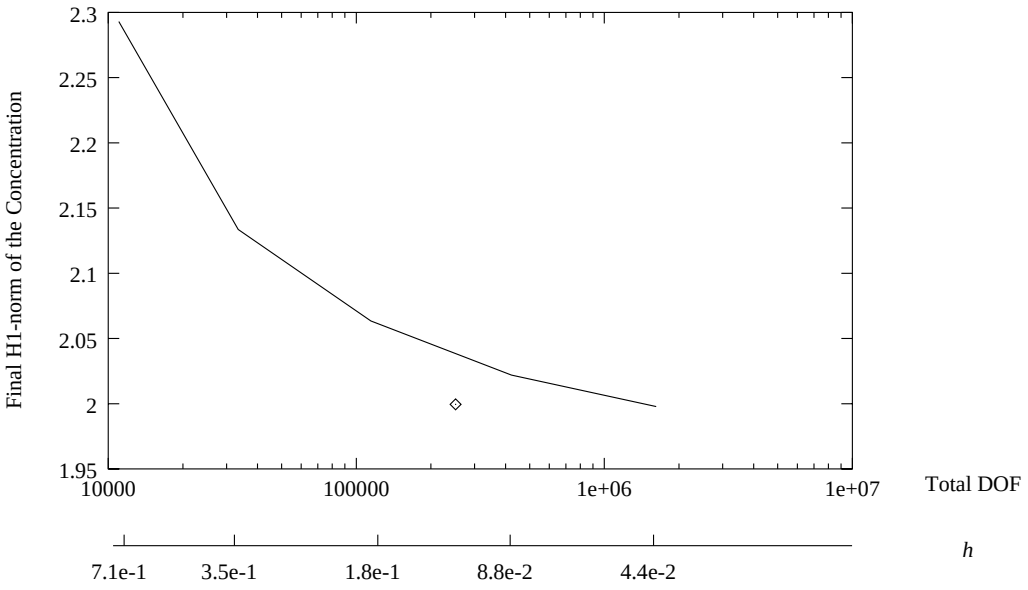

Figure 3.10: ICTA convergence as a function of the total DOF or h for uniform (solid curve) and adaptive ($\diamond$) mesh refinement.

due to the greater resolution of sharp gradients near the singularity provided by adaptivity. The excellent performance of the adaptive method further suggests that the error indicator (3.30) is a good measure of solution accuracy.

3.2.4.2 Bioartificial Artery

A prototypical BAA may be formed by allowing a tubular TE to contract on a rigid, impermeable, free-slip mandrel [133, 93]. Using the initial geometry of the cylinder $\{(r, \phi, z) \mid 2 < r < 3,\ 0 \leq \phi < 2\pi,\ 0 < z < 1\}$, this axisymmetric study features free surface conditions (3.34) at $z = 1$ and $r = 3$; a plane of symmetry (3.35) at $z = 0$; and a free-slip surface (3.35) at $r = 2$. Concentration contours are shown at four times in Figure 3.11. These adaptive results compare well with prior results [22, 23] obtained by standard finite element techniques. Since there are no sharp gradients, adaptive hr-refinement is needed less than with the ICTA problem. Mesh refinement is fairly uniform and is redistributed as the medium contracts.

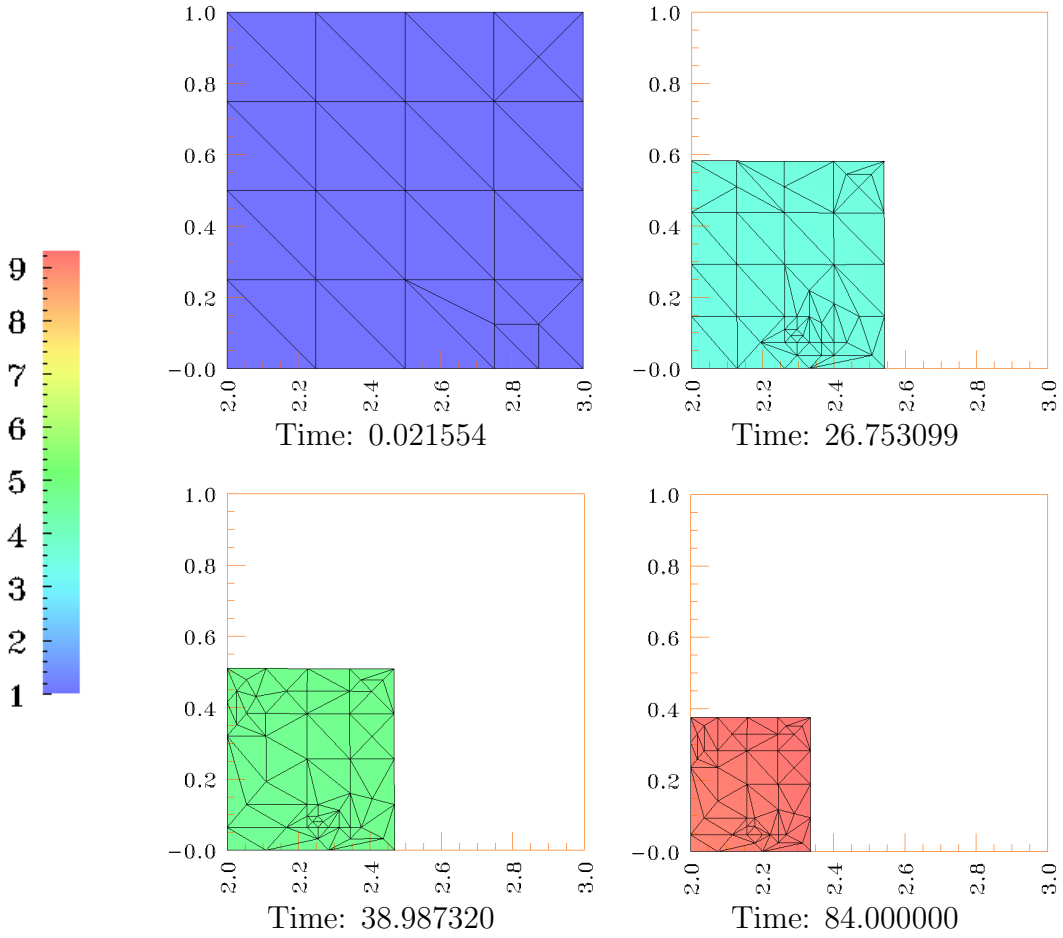

Figure 3.11: Concentration in a BAA at four times with adaptive meshes superimposed.

3.2.4.3 Reinforced Bioartificial Artery

Although we have not identified an appropriate objective function for optimal BAA design, we know that the initial shape of the BAA is an important (and controllable) design parameter. As a preliminary study, we analyze a BAA with reinforced ends and boundary conditions as shown in Figure 3.12. The volume of the reinforced BAA has been selected to be the same as that of the standard BAA (§3.2.4.2). Our hope was that this design will lead to a BAA with significantly higher collagen and cell content near the ends, which could facilitate suturing the BAA at the implant site. The geometry also provides a demanding test of the finite element software because of the singularity at the reentrant corner.

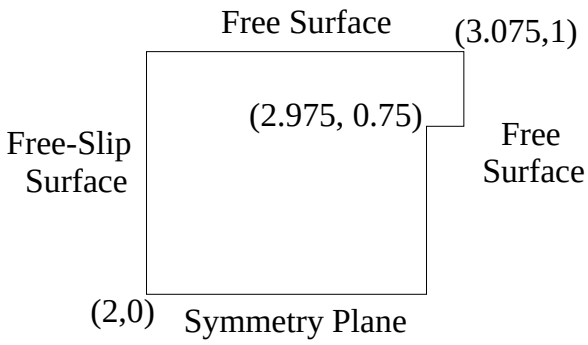

Figure 3.12: Geometry and boundary conditions of a reinforced BAA.

The parameters are the same as in the standard BAA problem (§3.2.4.2). Once again, concentration contours are shown at four times in Figure 3.13. As conjectured, the mesh is refined near the reentrant corner to maintain accuracy in the presence of a singularity. The mesh does not change significantly after some initial refinement. This is likely due to the self-similar behavior of the solution near the singularity.

The three-dimensional object the BAA represents in Figure 3.13 is shown in Figure 3.14. The representation shows an artery shunt, due to the small length along the axis relative to its radial length.

3.2.4.4 Discussion

An adaptive finite element software system has been used to solve the ABT system (3.1)-(3.5). Solutions of ICTA and BAA problems duplicate prior results [22]; however, the adaptive solutions have comparable accuracy with fewer total DOF. With adaptive *hr*-refinement, we were able to locate singularities and automatically refine the mesh in their vicinity. No *a priori* solution knowledge was necessary. The software is able to solve problems on arbitrary two-dimensional regions; hence, it should provide a useful tool for the investigation of optimal design and other problems involving complex geometry. Future problems are noted in §5.3

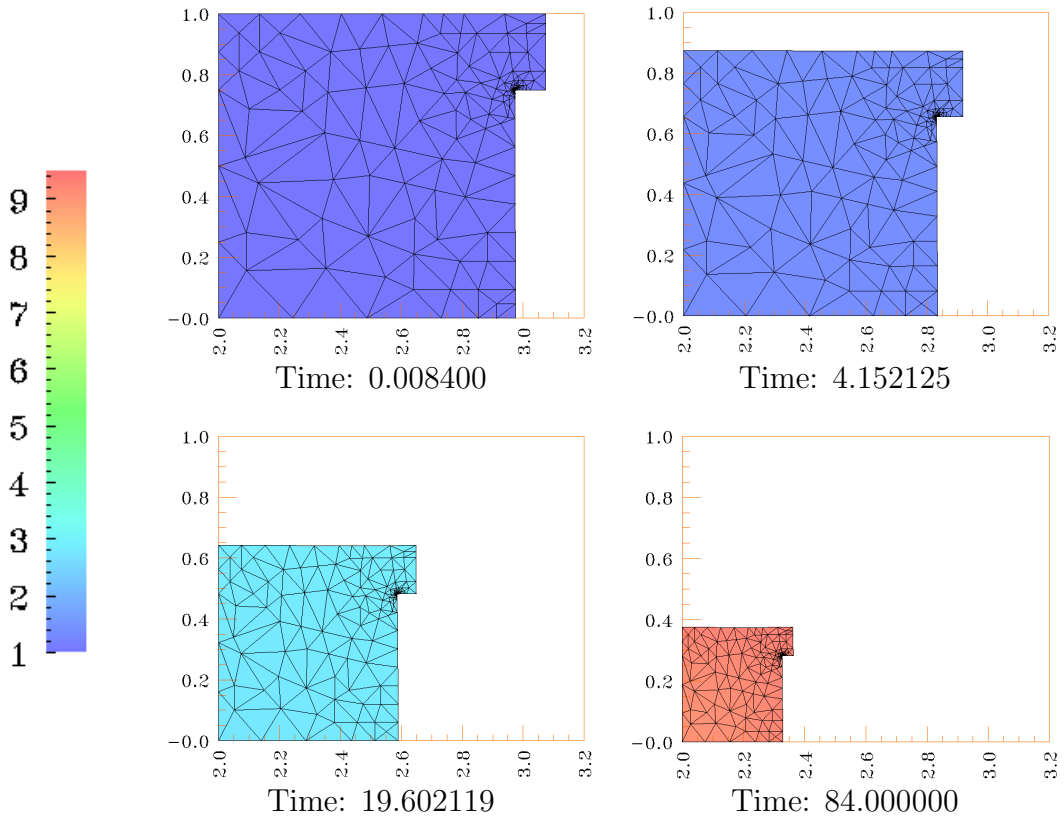

Figure 3.13: Concentration in a reinforced BAA at four times with adaptive meshes superimposed.

3.3 Design of Microfluidic Devices via Finite Element Simulation

3.3.1 Introduction

In this section, the flow of fluid through a micro-pump is simulated, by solving the Navier-Stokes equations in a moving domain using the prototype PDESE from the previous section. Re-entrant corners create a sharp gradient in the solution variables, thus requiring adaptive procedures. Since the domain in which the problem is solved is "thin" and the domain compresses as well as expands in time, elements deform and sometimes become badly shaped. Procedures are used to correct such elements.

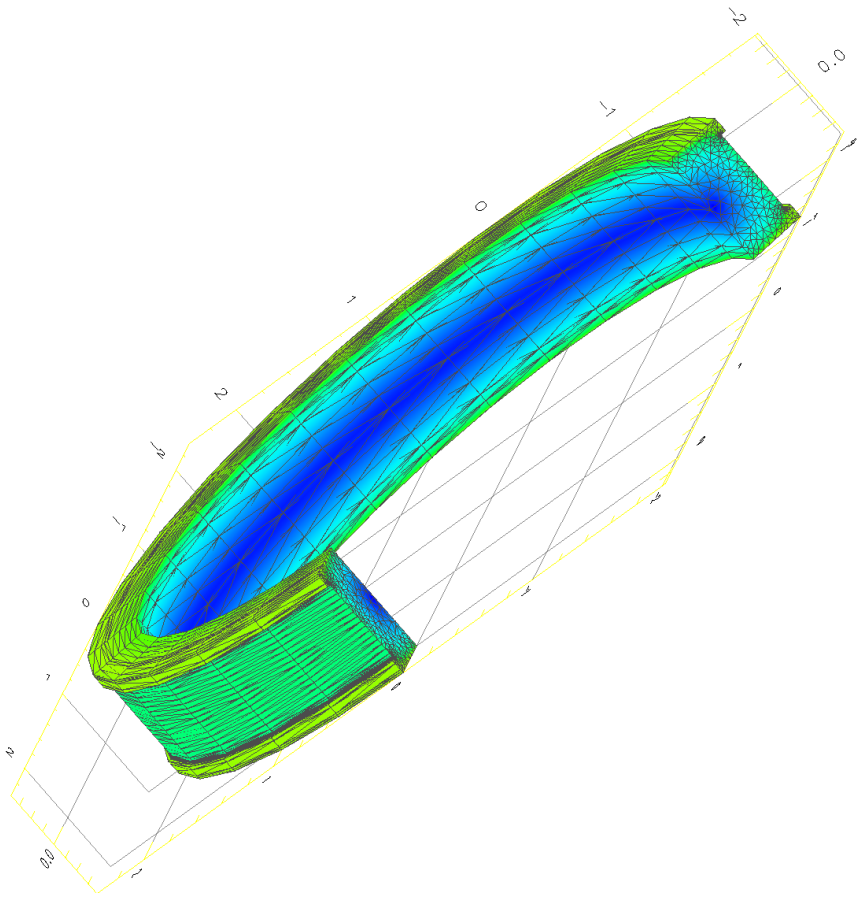

Figure 3.14: Three-dimensional representations of the reinforced BAA.

3.3.1.1 Microfluidics and Their Applications

Micro-Electo-Mechanical Systems (MEMS) technology has enabled the development of microscale chemical processes with broad potential application, including sensing [61, 140, 142] and chemical analysis (*e.g.*, "polymerase chain reaction (PCR) on a Chip" [89]). Microchemical processes often rely on multiple distinct subunits within an overall process and thus require a mechanism to move fluids (liquids or gases) between the subunits or in some cases (*e.g.*, microscale gas chromatography [139]) to move fluid through a subunit. Although there are passive mechanisms to move fluid through a MEMS device, the more common way to do so is by means of a pump.

3.3.1.2 Microfluidic Devices

There are a wide range of microfluidic devices (*e.g.*, [44, 50, 124]) with different mechanisms and performance characteristics. Our goal of a generally-usable, fluid-independent pump requires that we not rely on chemical methods (such as electro-osmotic pumps) but prefer a purely mechanical pump. In collaboration with Sandia National Laboratories, we have recently developed a surface-micromachined peristaltic pump (SMPP, [70]) that operates by creating an electrostatically-driven peristaltic wave. The SMPP is small (micron-scale), directly on chip (and thus suitable for connection to other on-chip units), and easily produced in bulk by Sandia's surface micromachining techniques. In this section, we focus on design of the SMPP, noting that the simulation methods described herein would be broadly applicable to other pump designs.

3.3.1.3 Design tools

One must choose not only the type of pump to use in a process but also the operating parameters of the pump. Like traditional chemical process design, the design of the SMPP (or of any microscopic process) can be reduced to a constrained optimization problem: minimize the cost of the device subject to various constraints such as required flow rate, maximum allowable shear, and maximum available power. In order to solve the optimization problem, one needs a mathematical model of the pump operation capable of predicting the performance of the pump as a function of its operating parameters. Further, since optimization in an inherently iterative procedure, the model must be solved rapidly and accurately to allow efficient determination of the optimal parameters.

3.3.1.4 Adaptive Methods

Adaptive finite element mesh refinement methods (§3.2.3.5) refine meshes in high error regions where the solution changes rapidly and maintain a coarse discretization everywhere else. As such, they are typically much more efficient than traditional methods, which utilize uniform mesh refinement. With solutions having steep gradients, an adaptive mesh refinement method will often yield solutions

having the same accuracy as traditional methods with far fewer degrees of freedom (DOF), such as in §3.2.

3.3.1.5 History and Motivation

The history of using FEMs to solve the Navier-Stokes equations is long and is not covered here as the reader is referred to the text by Zienkiewicz and Taylor [147] and references within. Previous work on simulating fluid dynamics problems with moving boundaries use non-adaptive approaches[132]; however, no work on adaptive moving boundary simulations of MEMS are known.

We investigate the solution of the Navier-Stokes equation for a MEMS application in two-dimensions. Due to the geometry of the domain, a sharp gradient is formed in the y component of the velocity. Adaptivity is shown to provide a marked increase in efficiency for this problem.

3.3.1.6 Organization

We use a moving-boundary Navier-Stokes model with a sinusoidal wave to approximate the motion of the moving membrane to describe the motion of the SMPP (§3.3.2). A brief description of the adaptive finite-element modeling framework used to solve the flow problem follows in §3.3.3. We set the design variables and calculate flow and energy versus a pressure rise at the output for a sample design study of the SMPP (§3.3.5). We present the results of this study and an analysis of the pump along with a performance appraisal of adaptive methods used for design calculations.

3.3.2 Model

3.3.2.1 Device

The basic design of the SMPP is depicted in Figure 3.15. It consists of three major components: (1) the fluid inlet and the outlet ports, (2) the pump chamber whose floor is composed of a series of actuation pads enclosed by a flexible polysilicon membrane, and (3) the electrical leads to the actuation pads. The actuation pads are electrically isolated from the pump wall, and each electrical lead is connected to only one pad. The pump sidewalls support the overhanging membrane which, when in use, is electrically grounded. This allows an electrostatic force to be generated

between the membrane and the actuation pad so that the membrane is pulled toward the pump chamber floor.

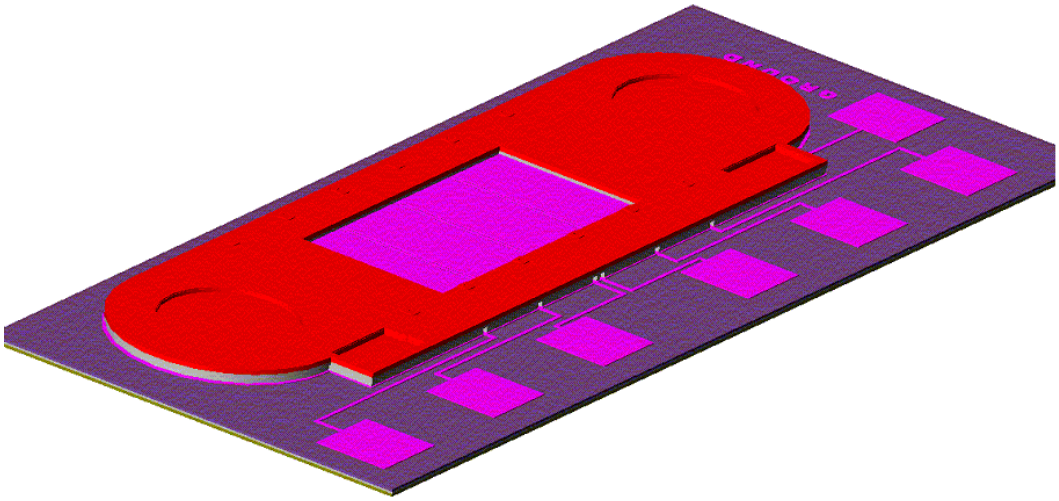

Figure 3.15: A three-dimensional drawing of the Surface Micromachined Peristaltic Pump (SMPP) generated by a geometric extrusion of the AutoCAD photolithography masks. A portion of the polysilicon membrane has been cut away to show the actuation pads that line the channel floor. The pump works by generating an electrical potential between each pad and the grounded membrane that covers the channel. Fluid enters and leaves through the backside of the silicon wafer via Bosch etched holes.

The pump works by developing a wave in the polysilicon membrane to propel fluid down the length of the channel from the inlet to outlet port. Manipulating the dynamics of the voltage potential between each pad and membrane creates the wave in the solid membrane. Voltage at the first actuation pad will pull the membrane down toward that segment of the floor. When the voltage at the first pad is turned off and applied to the second pad, the membrane will be moved toward the second actuation pad. Repeating this procedure for all of the actuation pads that line the channel floor establishes a peristaltic wave as shown in Figure 3.16.

3.3.2.2 Mathematical Model

A rigorous model of the SMPP would have to include the electrostatic attraction between the actuators and the upper membrane as well as the mechanical

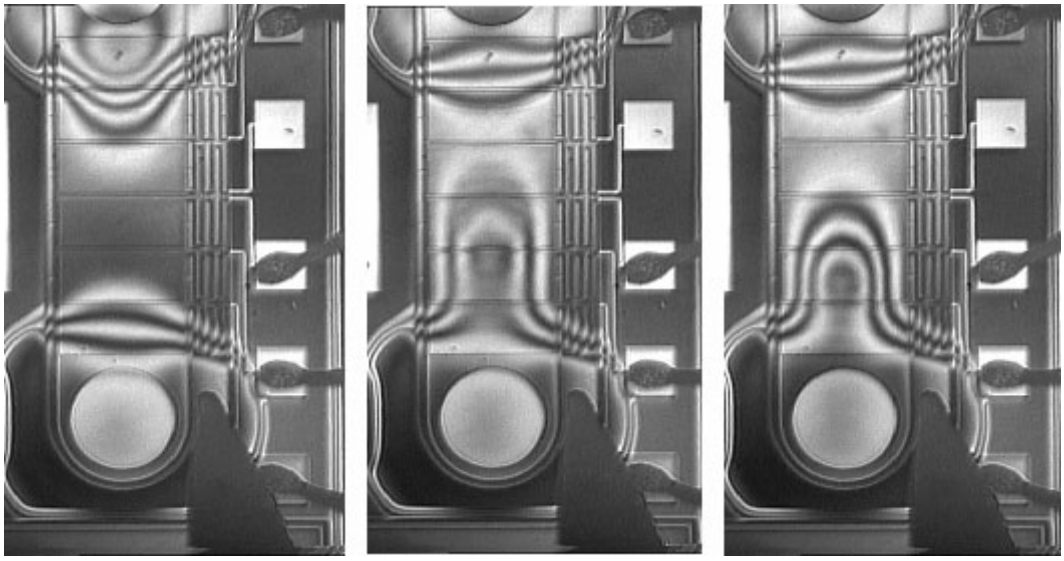

Figure 3.16: A packaged SMPP driving air with a membrane frequency of 1 Hz. The figure depicts three snapshots of interferometric patterns, off the surface of the pump membrane, indicating that the SMPP is generating a traveling wave. At each actuation pad, a potential of 70 volts is required to pull the membrane to the channel floor, approximately 3 microns.

coupling between the fluid stress and the shape of the membrane. We present and solve a simplified model where we assume that the membrane moves in a prescribed waveform, allowing us to solve the fluid dynamics problem independently of the membrane mechanics problem. The domain of the model is shown in Figure 3.17.

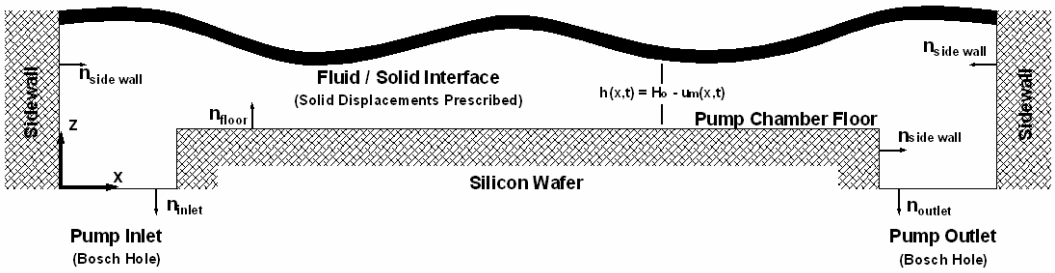

Figure 3.17: The two-dimensional Navier-Stokes model domain includes the Bosch inlet and outlet holes out the backside of the wafer.

The height of the channel is 8 microns, the length of the channel is 1175 microns, the height of the inlet and outlet (measured from the top of the silicon wafer to the bottom of the inlet) is 150 microns, and the width of the inlet and

outlet are 200 microns. Unlike in Figure 3.17, the inlet and outlet channels are offset from the outer walls by 30 microns. Notice that there is a significant scaling difference between the width and height of the channel. This is addressed in §3.3.3.1.

The fluid flow within the pump is governed by the incompressible Navier-Stokes equations [46]

$$\begin{gathered}\rho\frac{d\mathbf{v}}{dt}+\rho v\cdot\nabla\mathbf{v}=v\nabla\cdot\left[(\nabla\mathbf{v})+(\nabla\mathbf{v})^{\mathrm{T}}\right]-\nabla P\\ \nabla\cdot\mathbf{v}=0\end{gathered}\tag{3.37}$$

where the variables are velocity $\mathbf{v}=[v_x,v_y]^T$ and pressure P. We did not make the common simplification

$$\nabla\cdot\left[(\nabla\mathbf{v})^{\mathrm{T}}\right]=\nabla\left(\nabla\cdot\mathbf{v}\right)=0\tag{3.38}$$

for incompressible fluids in order to preserve the total stress within the divergence to facilitate incorporation of total stress boundary conditions in the finite element formulation. In the finite element weak formulation, integration by parts is performed on the divergence of the symmetric gradient of $\mathbf{v}$. For the purpose of this study, we assumed that the SMPP was pumping water. Therefore, the constants were set at $\rho=1$ and $\nu=0.01$.

The boundary in Figure 3.17 was divided into four sections: *rigid, inlet, outlet,* and *membrane.* The rigid portions of the boundary are those where the fluid is in contact with a solid, stationary portion of the SMPP satisfying the no-slip condition

$$\mathbf{v}=\mathbf{0},\quad (x,y)\in\Gamma_{rigid}.\tag{3.39}$$

The *inlet* and *outlet* boundaries represent connections between the SMPP and fluid reservoirs. For incompressible fluids, pressure is only determined to an arbitrary constant. We determined it by specifying the total normal stress at the inlet as

$$\left[\nabla\mathbf{v}+(\nabla\mathbf{v})^{\mathrm{T}}\right]_{yy}-P=0,\quad (x,y)\in\Gamma_{inlet}\tag{3.40}$$

We further assume that the flow is well developed at the inlet and outlet, so

$$v_x=0,\quad (x,y)\in\Gamma_{inlet}\tag{3.41}$$

At the outlet, we allow the SMPP to work against a pressure rise, and, thus, prescribe

$$\left.\begin{array}{c}\left[\nabla\mathbf{v}+(\nabla\mathbf{v})^{\mathrm{T}}\right]_{yy}-P=\Delta P_0\\ v_x=0\end{array}\right\},\ (x,y)\in\Gamma_{outlet} \tag{3.42}$$

where ΔP_0 is the pressure difference between the inlet and outlet reservoirs. Since many microfluidic applications require pumping through narrow channels, the required pressure head may be significant.

The fluid must move with the membrane and its motion is restricted to the vertical direction; thus, we have

$$\left.\begin{array}{c}v_x=0\\ v_y=\frac{4A\pi c}{\lambda}\sin\left(\frac{4\pi}{\lambda}\left(x-ct\right)\right)\end{array}\right\},\ (x,y)\in\Gamma_{\text{membrane}} \tag{3.43}$$

where the constant A is set so that the membrane's deflection is at most one third of the channel's height (Figure 3.17). Since the oscillations must taper at the end of the membrane, v_y is multiplied by a sine squared term if x is less than the taper length, which is 12.5% of the length of the membrane away from the ends. Similarly, for the first period, (3.43) is multiplied by a sine cubed term, so that the velocity is "ramped up" smoothly from zero to the maximum amplitude. In §3.3.4, the calculated quantities are computed without using the initial ramped solution. The height of the membrane may be obtained by integrating (3.43) in time, yielding a sinusoidal form.

3.3.3 Solution Method

3.3.3.1 Rescaling

Since the channel of the domain has a length much longer than the height, the equations (3.37) is rescaled to a dimensionless form. Letting L be the length of the channel, H the height of the channel, c the wave speed, λ the wave length,

$Re = cL/\nu$, $a = H/L$, and $z = \lambda/L$, then with the scalings

$$x' = \frac{x}{L}, \qquad y' = \frac{y}{H}, \qquad t' = \frac{c}{\lambda a}t \tag{3.44}$$

$$v_x' = \frac{1}{c}v_x, \qquad v_y' = \frac{1}{ca}v_y, \tag{3.45}$$

$$P' = \frac{1}{C_p}P, \tag{3.46}$$

where the primed variables are the scaled variables, (3.37) scales to

$$\frac{\partial v_x'}{\partial t'} + zav_x'\frac{\partial v_x'}{\partial x'} + zav_y'\frac{\partial v_x'}{\partial y'} = \frac{1}{Re}\left[2\frac{za}{\rho}\frac{\partial^2 v_x'}{\partial x'^2} + \frac{za}{\rho}\frac{\partial^2 v_y'}{\partial x'\partial y'} + \frac{z}{\rho a}\frac{\partial^2 v_x'}{\partial y'^2}\right] - \frac{C_p za}{\rho c^2}\frac{\partial P'}{\partial x'} \tag{3.47}$$

$$\frac{\partial v_y'}{\partial t'} + zav_x'\frac{\partial v_y'}{\partial x'} + zav_y'\frac{\partial v_y'}{\partial y'} = \frac{1}{Re}\left[\frac{za}{\rho}\frac{\partial^2 v_y'}{\partial x'^2} + \frac{z}{\rho a}\frac{\partial^2 v_x'}{\partial x'\partial y'} + 2\frac{z}{\rho a}\frac{\partial^2 v_y'}{\partial y'^2}\right] - \frac{C_p z}{a\rho c^2}\frac{\partial P'}{\partial y'} \tag{3.48}$$

$$\frac{\partial v_x'}{\partial x'} + \frac{\partial v_y'}{\partial y'} = 0. \tag{3.49}$$

C_p may be arbitrarily chosen since the pressure only appears as a gradient. We have chosen $C_p = c^2/(Re \times a^2)$. When using the above scalings, the dimensions of the channel are 1×1. In the weak form integration by parts is performed on the terms with second derivatives. The rest of the terms are untouched in the weak form. For the rest of this section, the physical variables are used, but all of the computations were done with the scaled form.

3.3.3.2 Adaptive Finite Element Software

The initial-boundary value problem (3.37)-(3.43) is solved by an adaptive finite element software system [9, 105] which was described in §3.2.3 with

$$\mathbf{u} = [P, v_x, v_y]^{\mathrm{T}} \tag{3.50}$$

and appropriate substitution of $\mathbf{f}$, $\mathbf{g}$, and $\mathbf{a}$ by the incompressible Navier-Stokes equations. Only combinations of adaptive h- and r-refinement have been used. Since a sharp gradient in the v_y variable is encountered at the upper corners of the silicon wafer in Figure 3.17, the V_y variable (the finite element approximation of v_y) is used in (3.30) instead of C. Since the domain moves in time due to (3.43), mesh motion is used.

3.3.3.3 Mesh Reshaping

Mesh motion (*cf.* §3.2.3.5) may distort elements to the point where very acute angles are created. This occurs in the thin membrane where elements are compressed in response to the membrane motion. Such sliver elements are close to singularity and create numerical difficulties in the limit as angles approach zero. To avoid these difficulties, the smallest angle of each element is measured at every time step and the element is reshaped if the angle is smaller than a tolerance. The sliver may be merged with a neighboring element by collapsing its shortest edge as shown in Figure 3.18.

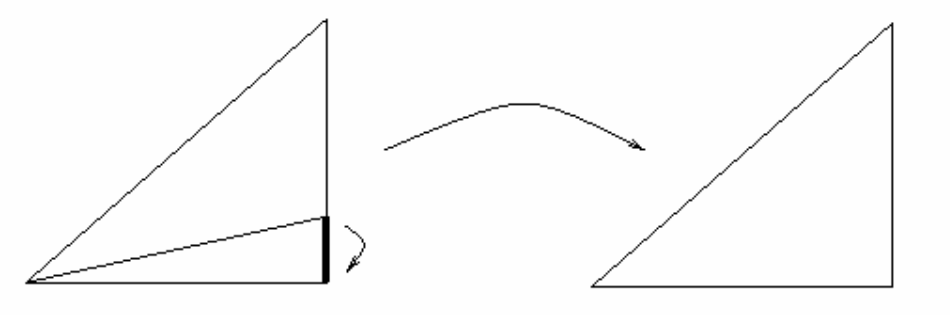

Figure 3.18: Collapse of a small edge.

The dark line indicates the edge to be collapsed. After collapsing all elements, those elements with a side opposite to a large angle are split using the template shown in Figure 3.19.

Edge swapping [9] is an alternative that was not used, where the edge between two elements bisects a small angle and a small edge is replaced by an edge which bisects a large angle and large edge. Thus, the dashed line in Figure 3.20 is replaced by the bold line in Figure 3.20.

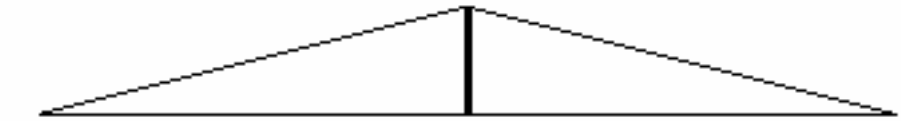

Figure 3.19: Edge splitting.

Figure 3.20: Edge swapping.

3.3.4 Design

3.3.4.1 Design Variables

There are many variables that can be changed in the design of the SMPP. In this study, we chose to consider the wave amplitude A and the pressure rise across the pump, ΔP_0, which is a key operating variable set by the pump's environment.

3.3.4.2 Base Case

The base for the study and the range of variables are given in Table 3.1.

Variable	**Base Value**	**Other Values**
Wave frequency, ω, Hz	1	1000
Wave amplitude A, % of height	33	0-33
Pressure Rise ΔP_0	0	0-ΔP^* (where $Q = 0$)

Table 3.1: Pump parameters.

We consider $\omega = 1$KHz. Because of the risk of "snap-through," in which the membrane gets to close to the actuator pads and is electrostatically pulled into the pad and shorts the system, only amplitudes of less than one third of the SMPP chamber height were considered. We explored all pressures in the range from zero (pump working against no pressure rise) and ΔP^*, the pressure rise at which the pump is no longer able to generate any net flow.

3.3.4.3 Calculated Quantities

In order for simulations to be of use in design, one must be able to derive quantities of practical significance from the results. One such quantity is the total flow through the pump. Since the flow is time-dependent over the pump cycle, we define the time-averaged total flow rate Q as

$$Q \equiv \frac{1}{T} \int_{t_0}^{t_0+T} \int_{\Gamma_{inlet}} v_y \, dx \, dt \tag{3.51}$$

where T is the period of the pump cycle, and t_0 is an arbitrary time chosen after a periodic solution has been established.

Other important derived quantities are the power requirement and the efficiency of the SMPP. Dimensional analysis of the mechanical energy balance shows that the dominant energy loss in the fluid is by viscous dissipation,

$$E_{viscous} \equiv \frac{1}{T} \int_{t_0}^{t_0+T} \iint_{\Omega} \mu \left[\nabla \mathbf{v} + (\nabla \mathbf{v})^{\mathrm{T}} \right] : \nabla \mathbf{v} \; dx \, dy \, dt \tag{3.52}$$

The other significant destination for energy is in the pressurization of the fluid (if $\Delta P_0 > 0$). The pressurization power is given by

$$E_{pressure} \equiv \frac{1}{T} \int_{t_0}^{t_0+T} \int_{\Gamma_{outlet}} v_y \, \Delta P_0 \; dx \, dt. \tag{3.53}$$

The total power required for the SMPP is then

$$E = E_{viscous} + E_{pressure}. \tag{3.54}$$

The efficiency of the pump is a measure of how much of the power supplied to the pump is converted to fluid pressurization (and thus presumably recoverable) rather than to viscous dissipation

$$\eta = \frac{E_{pressure}}{E_{viscous} + E_{pressure}}. \tag{3.55}$$

3.3.5 Results

3.3.5.1 Adaptive *versus* Non-adaptive Solution

Using the methods of 3.2.3.5, we are able to compute solutions with fewer DOF than by uniformly refining the mesh. Comparing Figures 3.21 and 3.22, we see that the adaptive mesh has been refined near the reentrant corner to a greater degree than the uniform mesh. The vertical scale has been stretched by a factor of 8 for clarity.

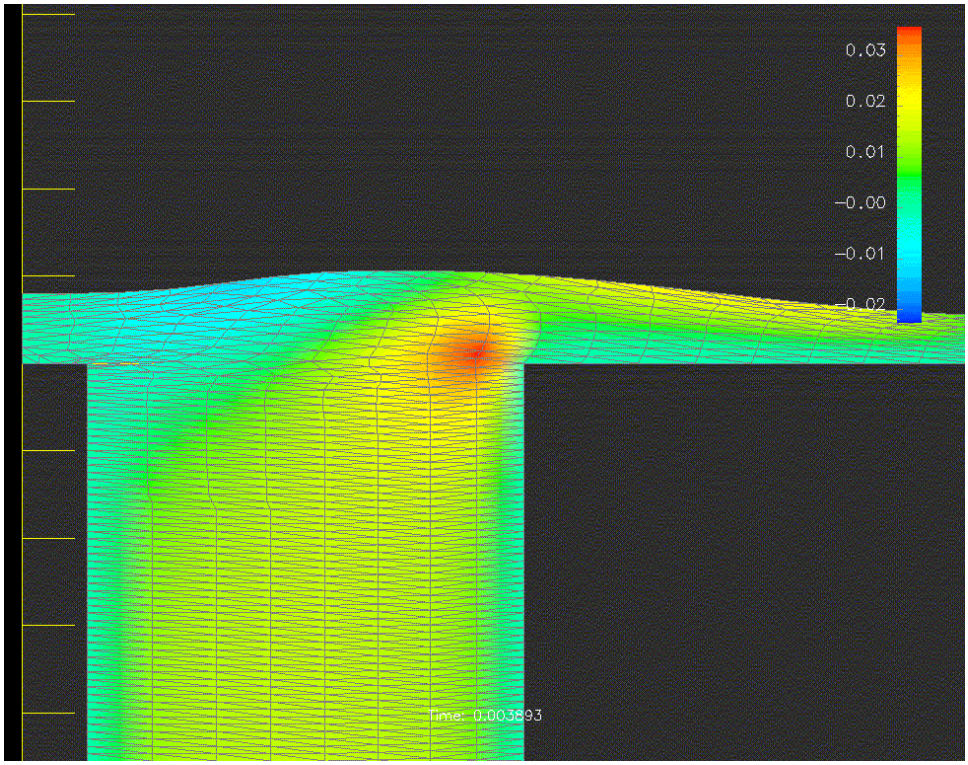

Figure 3.21: Non-adaptive mesh.

In Figure 3.23, we plot the final L^2-norm of v_y for both adaptive and successive uniform mesh refinements. If one linearly extrapolates the uniform refinement results, then one may compute a comparable solution using adaptive refinement with only 0.6% of the number of DOF.

3.3.5.2 Base Case

Figures 3.24 and 3.25 show typical velocity and pressure profiles for $\omega = 1$ KHz. Since the wave is moving from left to right, the fluid is pushed forward by the

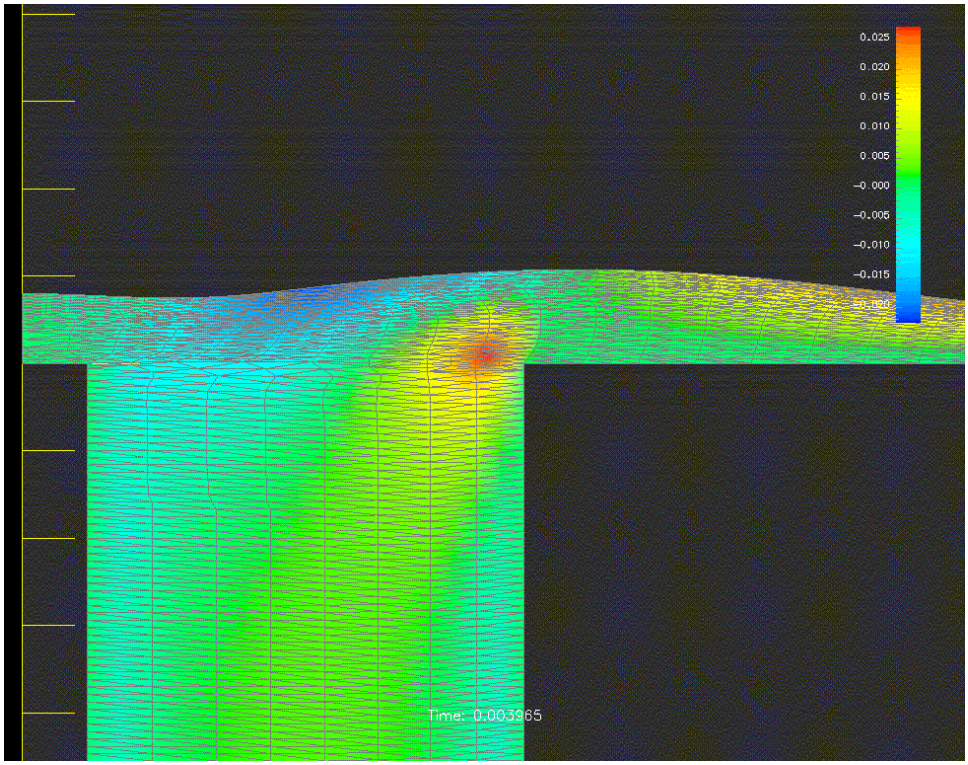

Figure 3.22: Adaptive mesh.

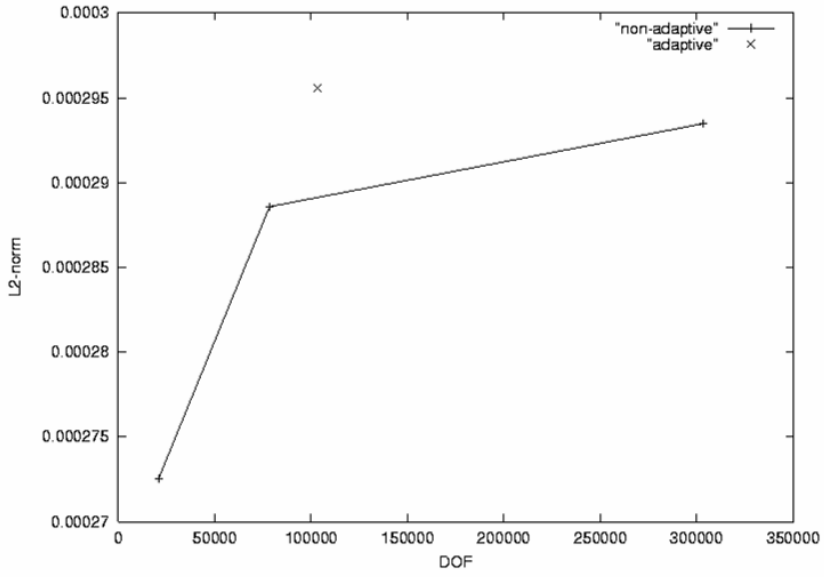

Figure 3.23: Adaptive versus non-adaptive methods.

upper membrane and also pulled into the pump (*i.e.* there is a net positive vertical velocity in the inlet channel). The minimum pressure in the chamber is at the trailing edge of the wave, as expected. Both forward flow (from behind the pressure minimum) and backward flow (from ahead of the pressure minimum) are seen, but the net flow is positive because of the smaller channel height and, therefore, greater resistance to backward flow.

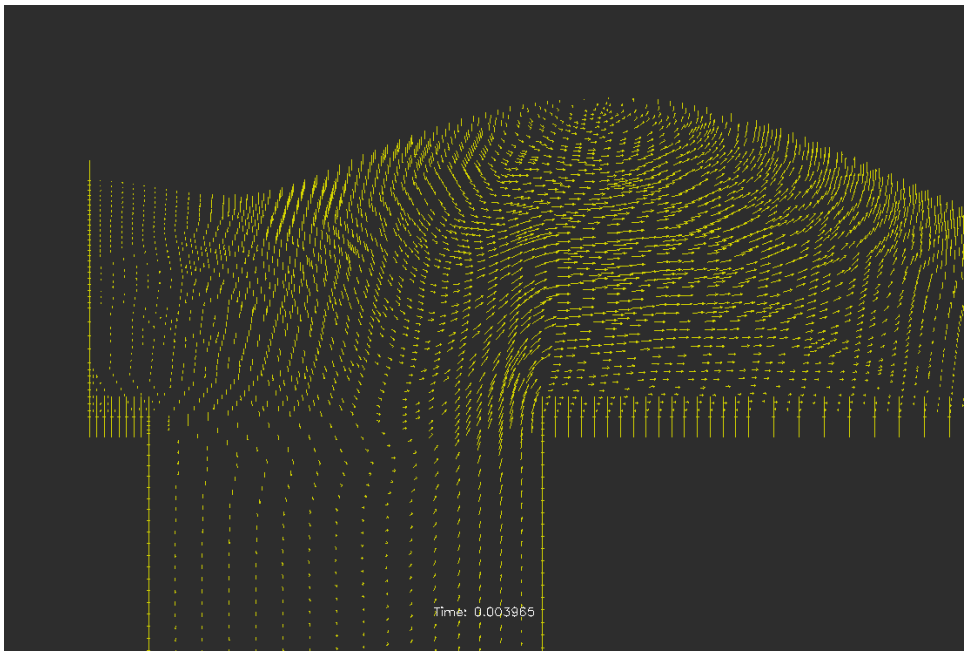

Figure 3.24: Velocity profile.

3.3.5.3 Calculated Quantities with a Pressure Rise

The results of Figures 3.24 and 3.25 were facilitated by our adaptive FE method. The reentrant corner where the inlet joins the chamber is known to lead to an integrable singularity in the velocity gradient, leading to significantly greater error for the same size mesh. The adaptive code recognizes this and automatically refines aggressively near the corner, whereas the non-adaptive mesh can only be adjusted by a uniform refinement. The effect of adaptivity was evaluated by computing the L^2-norm of the vertical velocity.

With the superior performance of the adaptive methods, we were able to do multiple runs to simulate SMPP performance under various operating conditions.

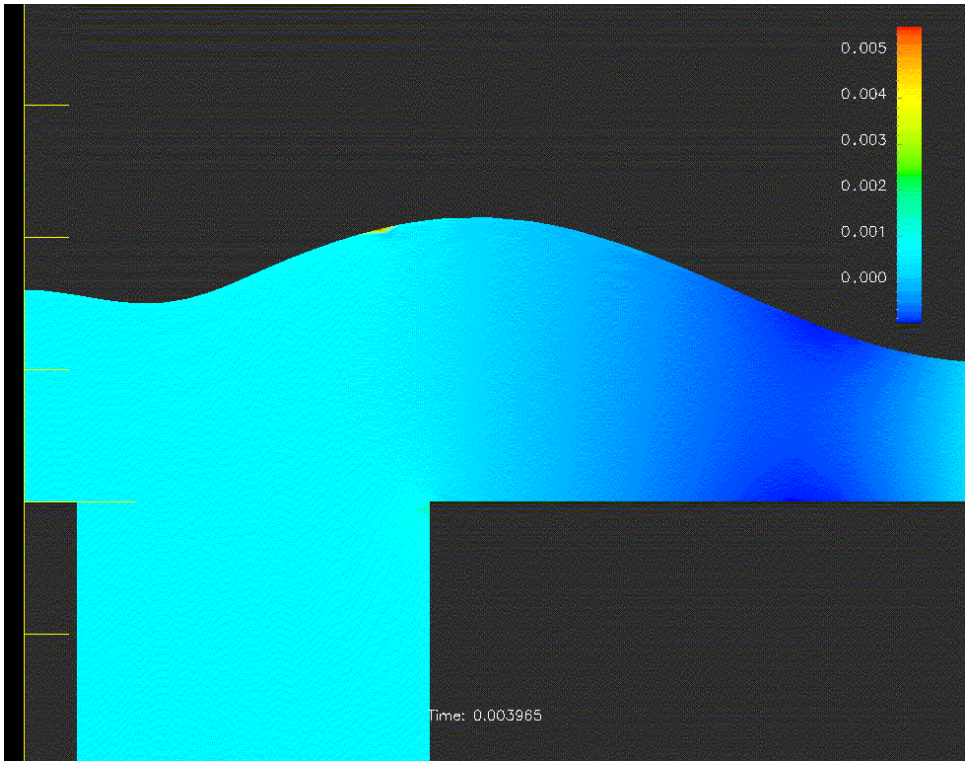

Figure 3.25: Pressure profile.

Figure 3.26 shows a classical "pump curve" as drawn for an SMPP operating according to the parameters of Table 3.1.

As the pressure rise opposing the pump is increased, the flow rate through the pump decreases linearly until the pump can no longer overcome the pressure and the flow rate drops to zero as shown in Figure 3.27.

The efficiency of the pump is zero when there is no opposing pressure (no energy is transferred to the fluid, which enters and leaves with the same velocity and pressure), and it is zero when there is no net flow. For the design specifications used, the maximum pressure rise that the pump could overcome is approximately 1.6×10^{-12}, from the data in Figure 3.27. The efficiency of the pump is extremely low, consistent with the large viscous losses at small scales. The efficiency versus the pressure drop is plotted in Figure 3.28.

The total viscous losses in all simulations were on the order of $10^{-4} ergs/s$. The viscous loss versus the pressure drop is plotted in Figure 3.29.

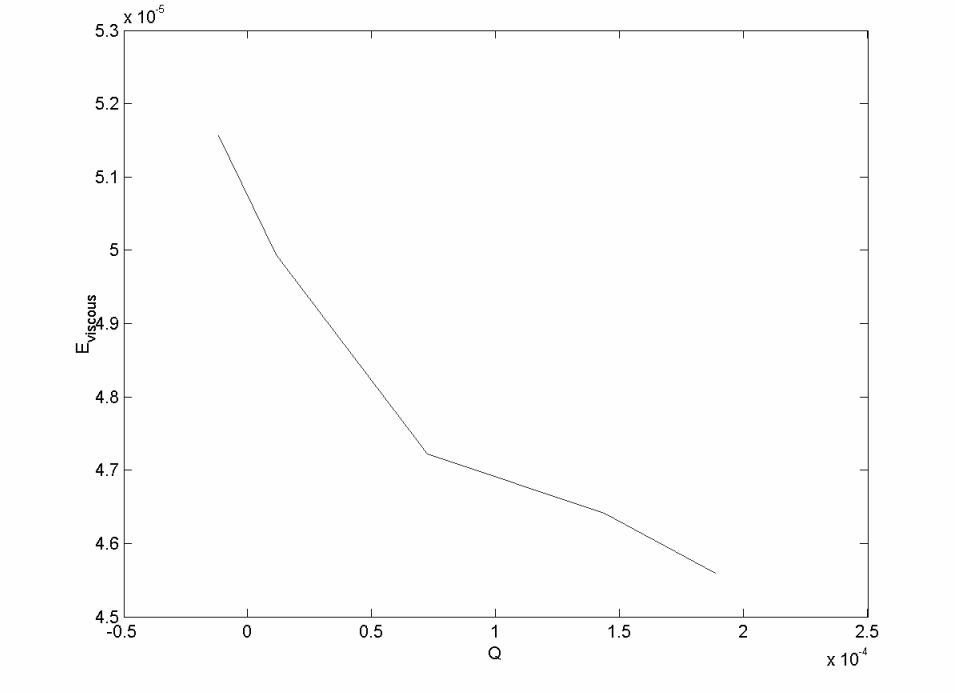

Figure 3.26: Pump flow versus viscous energy.

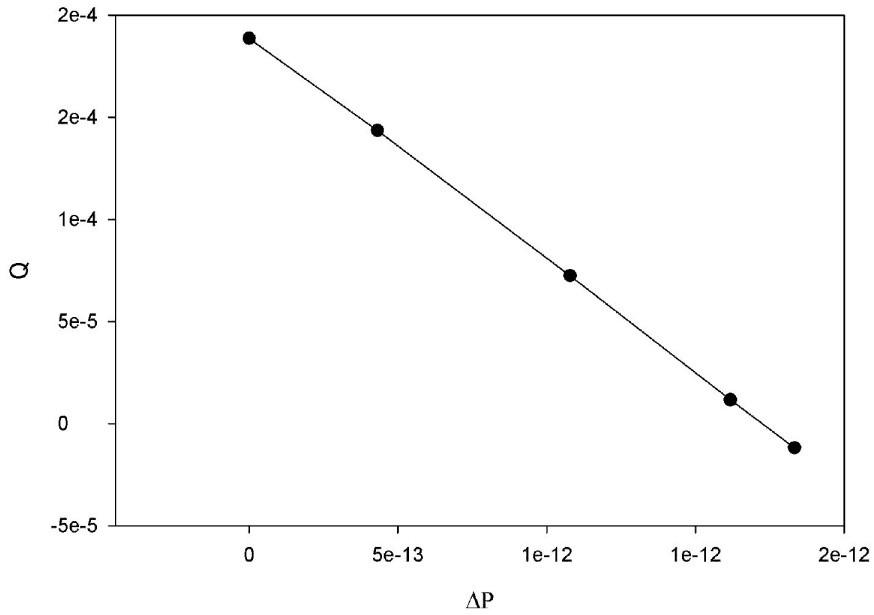

Figure 3.27: Pressure drop versus flow rate

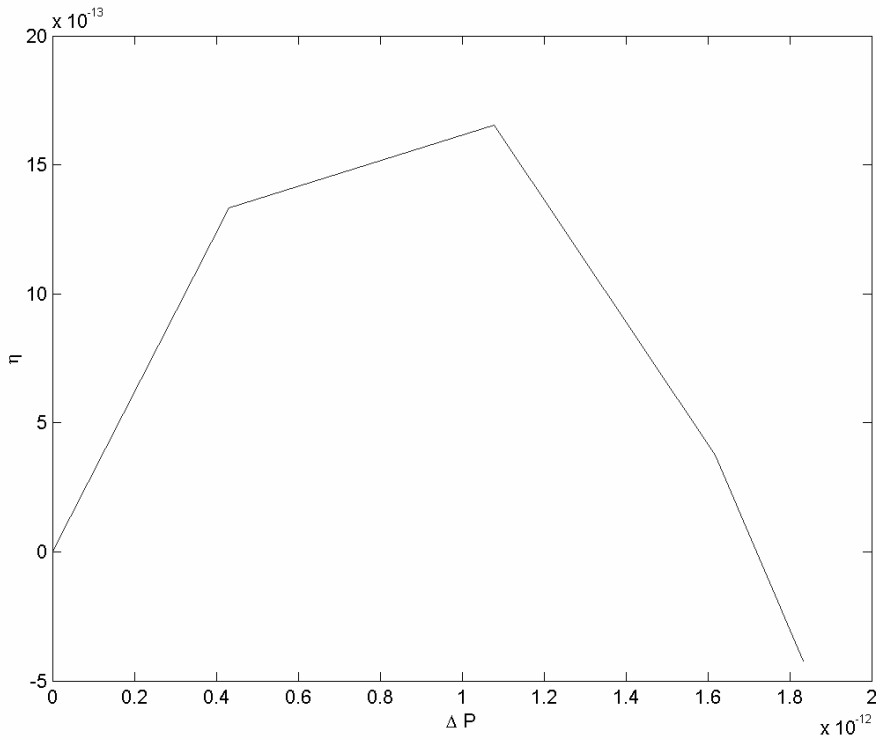

Figure 3.28: Pressure versus efficiency (1 KHz).

3.3.5.4 Calculated Quantities with Amplitude Changes

The final study was the effect of varying amplitude on the performance of the pump when working without a pressure rise. Both the flow rate and viscous dissipation increase quadratically with the amplitude, as can be seen in Figures 3.30 and 3.31.

3.3.6 Discussion

It was also shown that using *hr*-adaptive methods, the solutions may be obtained in a much more efficient manner, using as little as 0.06% DOF as a similar non-adaptive computation. *H*-adaptive methods can handle singularities in the gradient of the solution more efficiently than without. Using simulations, it was discovered that given the boundary conditions (3.43), the maximum pressure drop that can be used with this configuration is 1.6×10^{-12} and that the maximal pump efficiency occurs when ΔP_0 is approximately 1.0×10^{-12}. This shows that the prototype PDESE may be used to study and design a micropump and find parameters

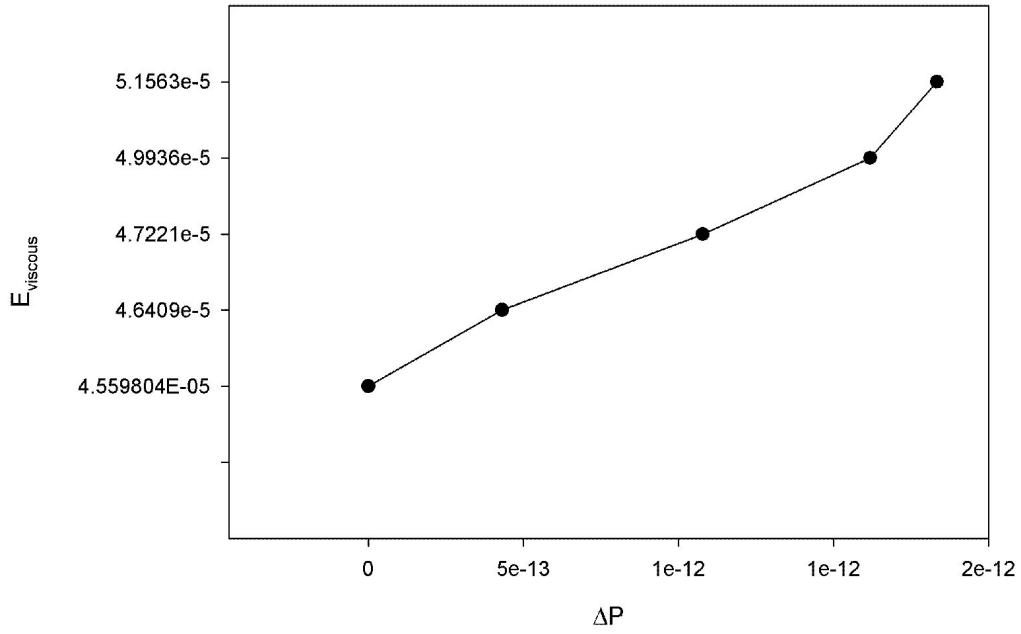

Figure 3.29: Pressure versus viscous energy dissipation.

for optimal efficiency. While an optimization package has not yet been incorporated into the prototype PDESE, the results indicate a good starting point for finding the optimal parameters in laboratory experiments. Similar studies for different configurations, such as different forcing functions for the membrane movement, may also be done without having to resort to expensive laboratory experiments.

3.4 Stage-Structured Infection Transmission and a Spatial Epidemic: A Model for Lyme Disease

3.4.1 Introduction and the Lyme Disease Life Cycle

In this section, we use the adaptive PDESE prototype software to study the spread of an epidemic of Lyme disease. The Lyme disease cycle is shown in Figure 3.32 [45]. Lyme disease in the Northeast U.S. involves interactions between no fewer than four species [106, 86]. The pathogen is a spirochetal bacterium *Borrelia burgdorferi*. The hematophagous vector is the black-legged tick *Ixodes scapularis*. Most newly hatched larvae attack white-footed mice *Peromyscus leucopus*; a few larvae attack other hosts [123]. Replete larvae molt and winter as nymphs. At the

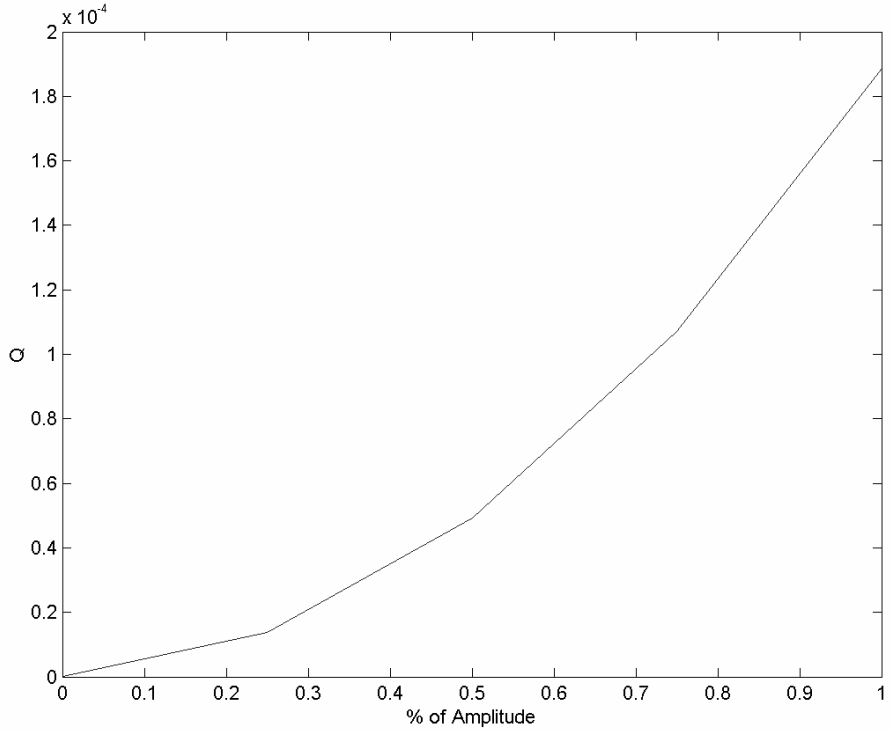

Figure 3.30: Percent amplitude versus average flow.

start of the second year, surviving nymphs "quest" for a second blood meal. Most successful nymphs again attack mice, but some infectious nymphs bite humans. After feeding on mice, nymphs quickly mature as adults. Adult ticks feed preferentially on white-tailed deer *Odocoileus virginianus*, and mating occurs on deer.

The spread of the disease is modeled using a nonlinear reaction-diffusion system (§3.4.2) which is solved adaptively (§3.4.5). Due to a sharp gradient in the initial conditions, adaptivity is required for an efficient and accurate solution. The model is new (and incorporates vector dynamics), so no previous results are available.

3.4.2 A Reaction-Diffusion Model for Lyme Disease

The reaction-diffusion model [40] used to study the propagation of Lyme disease, unlike other models [63], accounts for vector dynamics, such as vector mortality. The parameters and variables used in this study are listed in Tables 3.4.2 and 3.4.2 respectively. The parameters are assumed to be independent of spatial location.

Mice reproduce in a density-dependent manner, and incur density-independent

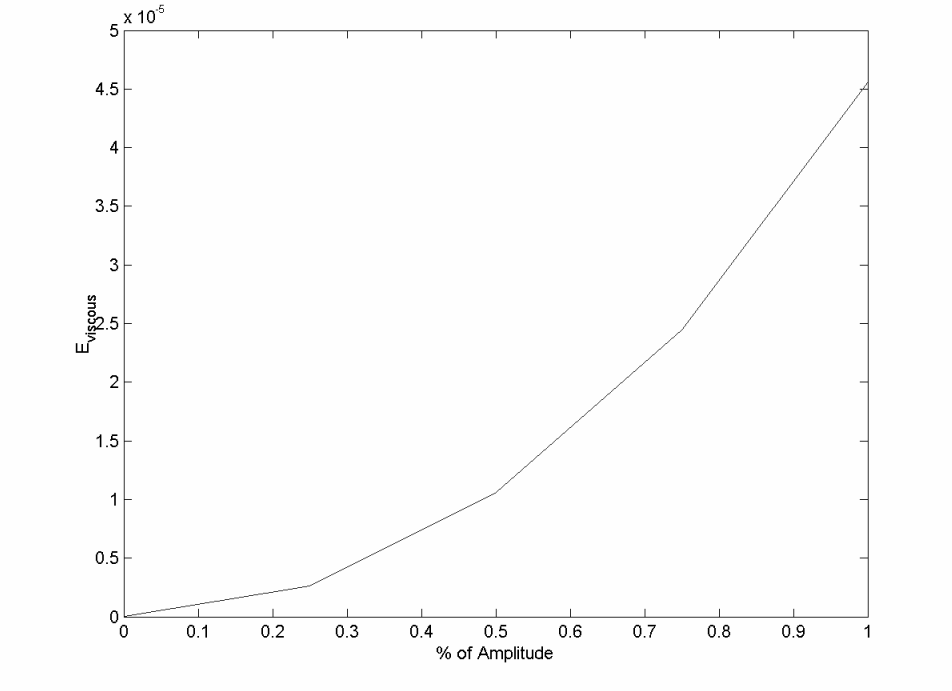

Figure 3.31: Percent amplitude versus viscous dissipation.

Parameter	Definition
r_M	Maximal individual birth rate in mice
K_M	Carrying capacity for mice
μ_M	Mortality rate per mouse
D_M	Diffusion coefficient for mice
D_H	Diffusion coefficient for deer
r	Maximal individual birth rate in ticks
c	Crowding coefficient in ticks
μ_L	Mortality rate per questing tick larva
μ_V	Mortality rate per feeding tick larva
μ_N	Mortality rate per tick nymph
μ_A	Mortality rate per adult tick
α	Attack rate, juvenile ticks on mice
γ	Attack rate, tick nymphs on humans
β	Susceptibility to infection in mice $(0 < \beta < 1)$
β_T	Susceptibility to infection in ticks $(0 < \beta_T < 1)$
σ	Rate at which larvae complete first blood meal

Table 3.2: Lyme disease model parameters.

mortality. Since mice are born uninfected, the equation for susceptible-mouse density $M(x, y, t)$ includes birth, death, acquisition of the spirochete from infectious-

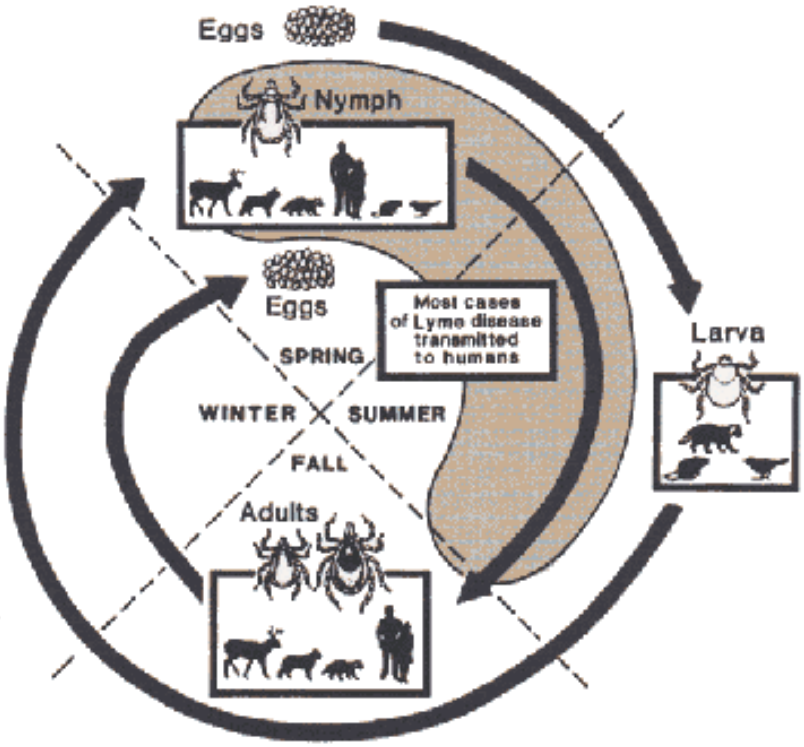

Figure 3.32: Life cycle of Lyme disease ticks.

Variable	Definition
M	Density of susceptible mice
m	Density of pathogen-infected mice
L	Density of questing larvae
V	Density of larvae infesting susceptible mice
v	Density of larvae infesting pathogen-infected mice
N	Density of susceptible questing nymphs
n	Density of questing infectious nymphs
A	Density of uninfested adult ticks
a	Density of pathogen-infected adult ticks

Table 3.3: Lyme disease model variables.

nymph bites, and dispersal:

$$\frac{\partial M\left(x,y,t\right)}{\partial t} = r_M\left(M+m\right)\left(1-\frac{M+m}{K_M}\right) - \mu_M M - \alpha\beta M n + D_M\left(\frac{\partial^2 M}{\partial x^2}+\frac{\partial^2 M}{\partial y^2}\right). \quad (3.56)$$

The density of pathogen-infected mice $m(x, y, t)$ increases as susceptible mice are bitten by infectious nymphs, and decreases through mortality. The equation for infected mice includes infection, death and dispersal:

$$\frac{\partial m\left(x, y, t\right)}{\partial t} = \alpha\beta M n - \mu_M m + D_M \left(\frac{\partial^2 m}{\partial x^2} + \frac{\partial^2 m}{\partial y^2}\right). \tag{3.57}$$

We assume that larval hatching rate depends nonlinearly on adult tick density [129, 110, 81]. The density of questing larvae declines through mortality and attacks on mice. Then,

$$\frac{dL}{dt} = r\left(A + a\right)\left[1 - c\left(A + a\right)\right] - \mu_L L - \alpha L\left(M + m\right). \tag{3.58}$$

Larvae must hatch at a positive rate when $A + a > 0$, so we assume c is small.

Given densities of questing larvae, the density of larvae infesting susceptible mice, $V(x, y, t)$, changes through successful attack, completion of the first blood meal, death, and dispersal while they infest mice:

$$\frac{\partial V\left(x, y, t\right)}{\partial t} = \alpha M L - V\left(\sigma + \mu_V\right) + D_M \left(\frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2}\right). \tag{3.59}$$

Since the duration of a larval meal seldom exceeds a few days [18], $\sigma > \mu_V$.

The assumptions concerning the density of larvae infesting pathogen-infected mice, $v(x, y, t)$, are similar. We substitute the density of infectious mice (m) for susceptible-mouse density (M) to obtain

$$\frac{\partial v\left(x, y, t\right)}{\partial t} = \alpha m L - v\left(\sigma + \mu_V\right) + D_M \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right). \tag{3.60}$$

The density of susceptible nymphs, $N(x, y, t)$ increases as larvae complete their first meal without acquiring the spirochete. The larvae may have infested a susceptible mouse, or attacked an infectious mouse and avoided infection. $N(x, y, t)$ decreases as they die, bite humans, and attack mice. Combining processes yields

$$\frac{dN}{dt} = \sigma\left[V + \left(1 - \beta_T\right) v\right] - N\left[\gamma + \alpha\left(M + m\right) + \mu_N\right]. \tag{3.61}$$

Infectious nymphs must have attacked a mouse infected with *Borrelia* as larvae, and then acquired the pathogen. Their density varies as

$$\frac{dn}{dt} = \beta_T \sigma v - n\left[\gamma + \alpha\left(M + m\right) + \mu_N\right]. \tag{3.62}$$

The term γ *n(x,y,t)* represents the local risk of Lyme disease to humans. We assume that nymphs biting humans do not feed long enough to mature as adults.

Given densities of susceptible nymphs, $N(x, y, t)$, the density of uninfected adult ticks, $A(x, y, t)$, changes through attacks of those nymphs on mice, death of adults and dispersal

$$\frac{\partial A\left(x, y, t\right)}{\partial t} = \alpha N\left[M + \left(1 - \beta_T\right) m\right] - \mu_A A + D_H\left(\frac{\partial^2 A}{\partial x^2} + \frac{\partial^2 A}{\partial y^2}\right). \tag{3.63}$$

Given densities of infectious nymphs $n(x, y, t)$, the density of pathogen-infected adult ticks $a(x, y, t)$ increases as infected nymphs attack any mouse, and as susceptible nymphs attack infected mice and acquire *Borrelia* during their second blood meal. Adding death and dispersal, we have

$$\frac{\partial a\left(x, y, t\right)}{\partial t} = \alpha\left[\left(M + m\right) n + \beta_T m N\right] - \mu_A a + D_H\left(\frac{\partial^2 a}{\partial x^2} + \frac{\partial^2 a}{\partial y^2}\right). \tag{3.64}$$

3.4.3 Traveling Wave Conjecture

In one-dimension, suppose a population exhibits both growth, self-regulation, and diffusion. Let the carrying capacity be K. Then if the initial density is zero everywhere except at one location, the population will increase to K expanding as a traveling wave [40]. The wave has an asymptotic speed of $\sqrt{4\rho D}$, where ρ is the intrinsic rate of increase and D is the diffusion coefficient [100, 78, 62]. Given the parameters in Table 3.4.2, one may compute the velocity ξ. In §3.4.5, ξ_0 is used, where

$$\xi_0 = \frac{\xi}{2}\sqrt{\frac{E[T]}{D_M}}, \tag{3.65}$$

where

$$E[T] \equiv \left(\gamma + \alpha Q + \mu_N\right)^{-1} + {\mu_M}^{-1}. \tag{3.66}$$

3.4.4 Reaction Equilibria

To compute the aspatial equilibria in one dimension, the diffusion terms were deleted. The positive abundance of ticks and mice in the absence of the spirochete and proportional infection of both mice and ticks equilibrium was chosen. Then, the equilibrium densities (the respective variable with a superscript asterisk) are as follows [40]:

$$Q \equiv K_M \left(1 - \frac{\mu_M}{r_M}\right), \tag{3.67}$$

$$A^* + a^* = \frac{1}{c} - \frac{(\alpha Q + \mu_L)(\sigma + \mu_V)(\gamma + \alpha Q + \mu_N)\mu_A}{rc\sigma(\alpha Q)^2}, \tag{3.68}$$

with $c < 1$,

$$m^* = Q\frac{\beta\beta_T\mu_A(A^* + a^*) - \mu_M Q}{\beta\beta_T\mu_A(A^* + a^*)}, \tag{3.69}$$

with $m^* > 0$,

$$n^* = \frac{\beta\beta_T\mu_A(A^* + a^*) - \mu_M Q}{\beta\alpha Q}, \tag{3.70}$$

$$M^* = Q - m^* \tag{3.71}$$

$$L^* = \frac{(\sigma + \mu_V)(\gamma + \alpha Q + \mu_N)\mu_A}{\sigma(\alpha Q)^2}(A^* + a^*), \tag{3.72}$$

$$v^* = \frac{\gamma + \alpha Q + \mu_N}{\sigma\alpha Q}\frac{\beta\beta_T\mu_A(A^* + a^*) - \mu_M}{\beta\beta_T}, \tag{3.73}$$

$$V^* = \frac{\mu_M(\gamma + \alpha Q + \mu_N)}{\beta\beta_T\alpha\sigma}, \tag{3.74}$$

$$N^* = \frac{\mu_A(1 - \beta_T)}{\alpha Q}(A^* + a^*) + \frac{\mu_M}{\alpha\beta}, \tag{3.75}$$

$$a^* = \frac{\alpha}{\mu_A}(Qn^* + \beta_T m^* N^*), \tag{3.76}$$

$$A^* = \frac{\alpha}{\mu_A}N^*(Q - \beta_T m^*). \tag{3.77}$$

3.4.5 Results

The parameter values used in the study are listed in Table 3.4.5 with $D_H = 0.1 = 10D_M$.

Parameter	Value (run 1)	Value (run 2)	Value (run 3)
r_M	0.04	0.008	0.008
K_M	6.0	4.0	4.0
μ_M	0.02	0.006	0.006
r	0.35	0.015	0.015
c	0.0003	0.0025	0.0025
μ_L	0.09	0.01	0.01
μ_V	0.08	0.01	0.01
μ_N	0.06	0.008	0.008
α	0.02	0.02	0.02
γ	0.005	0.001	0.001
β	0.3	0.35	0.35
β_T	0.5	0.45	0.45
σ	0.25	0.2	0.2

Table 3.4: Lyme disease model parameters values.

The initial conditions of the variables are listed in Table 3.4.5 where

$$g = \exp\left(-\frac{(x-x_0)^2 + (y-y_0)^2}{\epsilon^2}\right) \tag{3.78}$$

with $\epsilon = 0.001$, and (x_0, y_0) representing the center of the domain, $\left(\frac{1}{2}, \frac{1}{2}\right)$.

Variable	Initial Value
M	$3 - g$
m	g
V	1.669
v	363.4
A	735.7
a	2180
L	80.32
N	29.29
n	29.03

Table 3.5: Lyme disease initial values.

We compared the equilibria of the model, discussed in §3.4.4, with simulation results. (3.56)-(3.64) were solved using the adaptive software discussed in §3.2.3. The results are tabulated in Table 3.4.5.

μ_A	0.002	0.025	0.045
m^*	2.81	2.97	2.93
$m(s)$	2.81	2.97	2.93
n^*	50.04	350.09	141.76
$n(s)$	50.04	350.17	141.69
N^*	56.71	356.76	148.42
$N(s)$	56.71	356.84	148.36
ξ_0	2.31	4.04	3.22
$\xi_0(s)$	1.99	2.68	2.33

Table 3.6: Simulation results are given in rows with "(s)".

3.4.6 Discussion

The analysis in §3.4.5 using the adaptive PDESE prototype software generally matches closely with the expected values. The traveling wave conjecture was verified numerically. H-adaptivity was required due to the sharp initial condition (3.78). One may now consider simulations with various initial densities of Lyme disease as well as various parameters to determine how the disease will spread.

CHAPTER 4
APPLICATIONS IN THREE DIMENSIONS

4.1 Trellis

4.1.1 Introduction

In this section, the simulation framework developed by the Scientific Computation Research Center (SCOREC) at Rensselaer Polytechnic Institute, Trellis, is described. Trellis is an object-oriented *hr*-adaptive PDE solver, written by J.-F. Remacle and O. Klaas but with contributions from D. Datta, S. Kocak, K. Pinchedez, and the author of this thesis, which currently uses Galerkin finite element methods using polynomial basis functions in one, two, and three dimensions. Work is presently underway to expand and parallelize the framework. Trellis is a major step towards the PDESE ideal described in §1.1. The original Trellis framework described by Beall [29] forms the basis for the current version although most of the code has been rewritten.

4.1.2 Motivation

There have been many attempts to create an object-oriented finite element framework [12, 75, 30, 91]. Unfortunately, all of them have serious deficiencies which make them unsuitable for solving problems in this thesis, such as the three-dimensional biphasic sections in §4.2. For example, some object-oriented finite element frameworks [12, 75] cannot solve time-dependent problems. Kaskade [30] does not allow for parallelism nor does it interface to different external software packages, such as DASPK, nor does it allow for interfaces to modelers, such as Parasolid. A project closer to Trellis in capabilities is Diffpack [91]. Unfortunately, Diffpack does not support advanced DAE solvers, such as DASPK, and like the other cited software, does not provide support for moving mesh problems.

The prototype PDESE by Aiffa [9], used in §3.2 and §3.3, does not extend to three dimensions nor does it do warm restarts using DASPK (§4.1.8). Furthermore, the original version of SCOREC's Mesh Database has been replaced by a much

faster successor, AOMD [111]. The lack of a framework design approach [58] (or the presence of software antipatterns [39]) in the prototype PDESE is known to be conducive to a lack of code extendibility.

To address the deficiencies of previous codes, a new structure for a simulation framework is needed. In particular, the structure must be able to accommodate adaptivity at all levels (mathematical model, discretization, solution techniques, *etc.*). The code must be able to handle a large class of general problems, rather than specific types of problems. The code must also be able to import models representing problem domains from a number of geometric modeling systems, such as Parasolid. Coupling between the geometric modeling system and the framework is needed to facilitate geometry-based problem specification. This is critical for p-refinement since curved element approximations must be used for high-order accuracy [54]. Furthermore, the adaptive process cannot accurately refine the mesh on the boundary without knowing the shape. However, the geometric problem specification must be independent of the solution code.

Finally, Trellis must have a rich data structure supporting the adaptive process that is also efficient.

All of the above requirements are addressed by Trellis.

4.1.3 Trellis Overview

4.1.4 Unified Modeling Language Nomenclature

The Unified Modeling Language (UML) [36] is used to describe the structure of Trellis. A *class*, analogous to a C++ class, is denoted as shown in Figure 4.1, where the top section represents the name of the class, the middle section contains the attributes (variables in C++), and the bottom section contains the operators (member functions in C++).

The names of attributes and operators may be preceded by a $+$, $\#$, or $-$, depending on whether it is public, protected, or private, respectively. For clarity, particularly when showing relationships, the attributes and operators may be omitted. The namespace of the classes may be omitted when it is clear from the context.

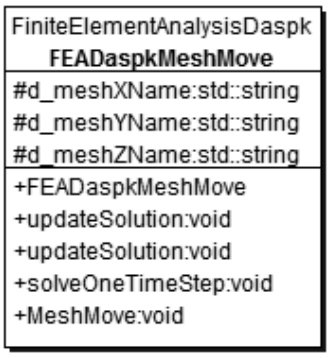

Figure 4.1: UML class diagram.

Dependencies, such as one class using another class in a function argument, return type, or an attribute that is of a class type, is denoted with an arrow as in Figure 4.2. Here, `mAttachableEntity` depends on `mEntity`, since there is a pointer attribute to `mEntity` in `mAttachableEntity`.

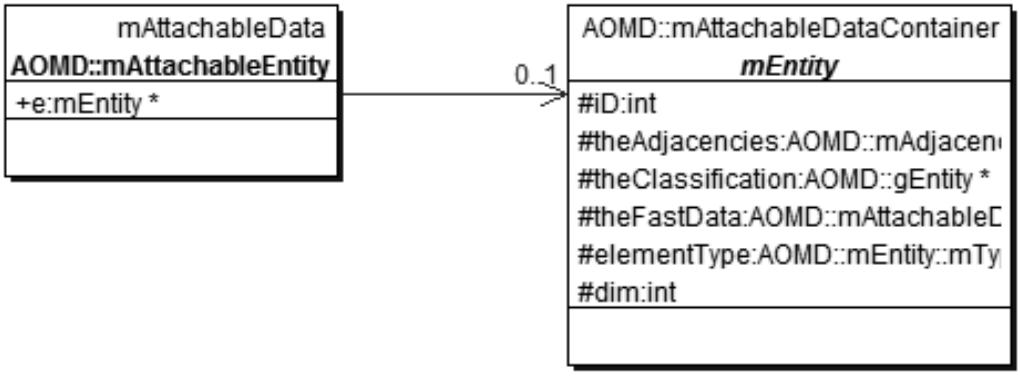

Figure 4.2: UML dependency.

One can also have a dependency if a member operator has a class as an argument of the operator. A *generalization*, or "is-a-kind-of", is a relationship where the child class may be used anywhere that the parent(s) classes may be used, but not the reverse. In C++, a generalization refers to base class(es) from which a class is derived. A generalization is denoted by an arrow with a triangular head as in Figure 4.3, where `SpoolesLinearSystemSolver` is a class derived from `LinearSystemSolver`.

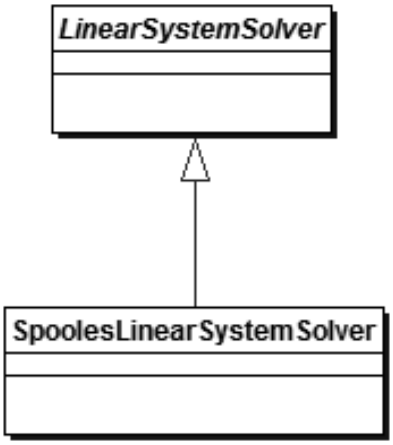

Figure 4.3: UML generalization.

An *aggregation* is a special case of a dependency such that one or more instances of a class is contained in the attribute section of a class. For example, in Figure 4.4, the class `BezierMapping` contains a vector of `mPoint` as a member variable, thus giving rise to an aggregation.

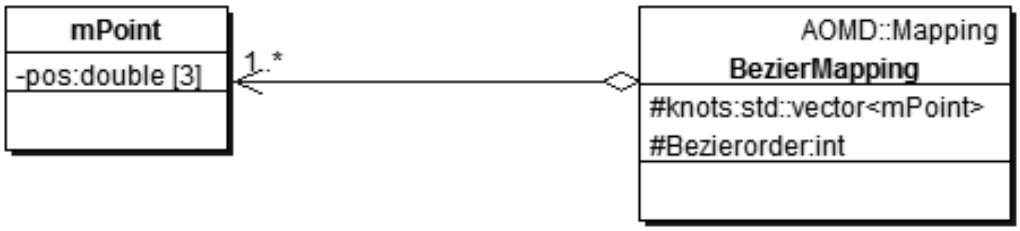

Figure 4.4: UML aggregation.

In the case of dependencies and aggregation, multiplicity may be denoted at the ends of the arrow. For example, in the above case of aggregation, `BezierMapping` contains 1 or more instances of `mPoint`. Finally, a collection of objects, typically classes, is called a *package*. A package is purely conceptual; however, sources for separate packages are usually stored in separate subdirectories.

4.1.5 Package Structure

Trellis has three main packages (and one minor package) and many subpackages:

1. `Applications` This package contains all major applications that Trellis can do (or applications using large portions of Trellis).

    - `DG:` This is a discontinuous Galerkin package to solve hyperbolic conservation laws [114].
    - `BarocasBiphasic:` This package contains the analysis code to solve the ABT equations in three-dimensions, discussed in §4.2.

2. `Meshing`

    - `MeshAdapt:` The procedures to do refinement, coarsening, and other mesh modifications.
    - `MeshTools:` This is a collection of routines to operate on a mesh, such as creating or deleting a region or face, computing measures (such as the ratio of the longest side over the shortest sides) on mesh entities, such as edges, faces, and regions, and other miscellaneous tools.
    - `templateRefine:` This package contains the templates by which the mesh is refined. It is similar to the templates in §3.2.3.5, adding templates in three-dimensions.

3. `Trellis`

    - `AOMD:` This package contains the data structure and tools to store a mesh. It also includes a subpackage to perform quadrature over an edge, face, or region. This package is described in more detail in §4.1.6.
    - `Core:` The main finite element analysis classes are located in this package, with routines to handle global matrix assembly; a generic interface to the `Solver` package; a parser for files that specify boundary conditions, initial conditions, and other parameters; time integrators; and an error estimator and adaptivity routines.

- `Field:` Routines pertaining to solution fields, such as the shape functions, a class to manage the DOF, gradient and divergence operators and the like, operators to form material laws, and output, are located in this package.
- `Solver:` This package contains interfaces to various linear algebraic packages, such as PETSc [17] and SPOOLES [14].

4. `Util:` This package is minor in that it contains only machine and compiler specific customizations to the `Makefile`.

4.1.6 Geometry Based Framework and AOMD

The next generation of analysis tools must be able to support automatic discretization and adaptivity while remaining flexible regarding available solution techniques. They must be able to solve a wide variety of problems, where both the equations and domain can be easily described.

For the domain and the parameters to be used on the domain to be readily described, the problem must be defined in a manner that corresponds to the physical simulation. Typically, the domain, or *model*, is created using a computer-aided design (CAD) program (or modeler), such as Gmsh, Parasolid, Shapes, AutoCAD, or ACIS, whose output is a model. Thus, the framework must be able to interface with these modelers to obtain domain information. Access to the geometry is needed since a refined mesh must correctly approximate the domain [54]. The domain is usually created in sections, where parameters and/or boundary conditions are uniquely specified. It is advantageous to have a framework where parameters and boundary conditions are specified on a particular section, known as a *model entity*. A model entity may be a region, face, edge, vertex, *etc.*, as shown in Figure 4.5.

Non-trivial boundary representations are needed since complex boundaries, such as Figure 4.6, can arise.

Given a model, Trellis assumes a corresponding mesh either exists and is loaded, or is generated. As part of the goal of providing a PDESE, it is emphasized that Trellis can call an automatic mesh generator to create a mesh, given a model. Then, mesh entities, *e.g.,* regions (elements in a three-dimensional mesh),

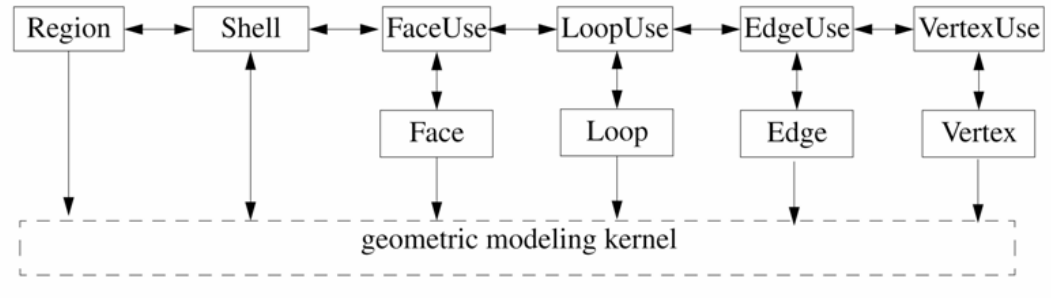

Figure 4.5: Non-manifold boundary representation.

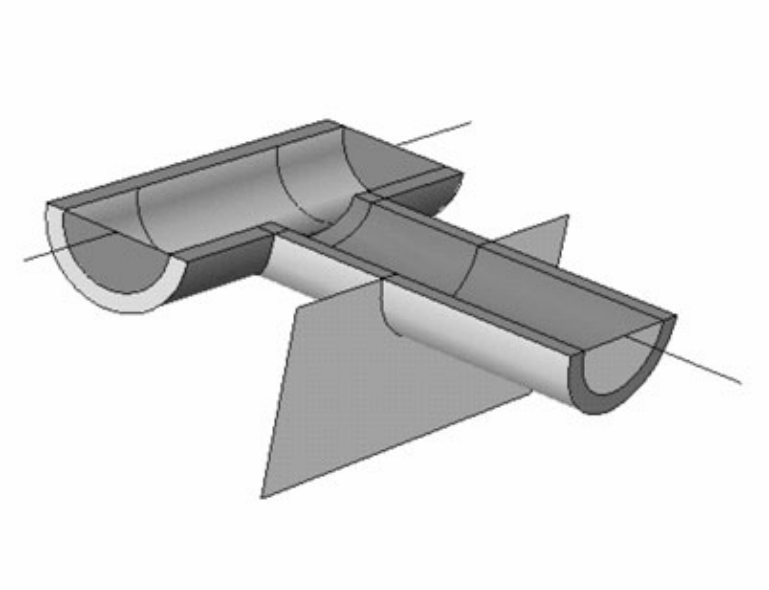

Figure 4.6: An example of a non-manifold model.

faces, edges, and vertices, are associated (or classified) to model entities, as shown in Figure 4.7.

For example, a face in a three-dimensional mesh may be associated with a model face if it lies on the surface or in a model region if the face is in the interior. Thus, for example, operations specified on a model face are actually done on each mesh entity (regions, faces, edges, vertices) that is associated with that model face. It is assumed that an associated mesh exists for a given model. Separating the model input from the mesh input, it is possible to support a wide variety of mesh and model input formats. It is important, however, that the choice of modeler be independent of the rest of the analysis. Trellis is a component based software system

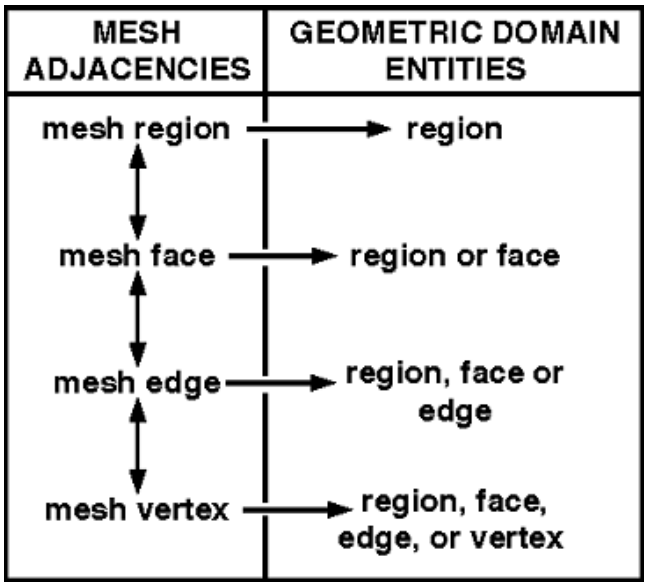

Figure 4.7: Mesh and model entities associations.

to implement such specifications using AOMD, as described below.

A rich and efficient data structure for storing the mesh and model is required to support the adaptive process. Thus, we turn our attention to Algorithm-Oriented Mesh Database (AOMD) [111], written by J.-F. Remacle, which is a new version of the SCOREC Mesh Database [27] discussed in §3.2.3.1. This software is not a database in the traditional sense, but rather a persistent object-oriented data structure. Again, like the Mesh Database, AOMD allows one to attach data, `mAttachableData`, to mesh entities, `mEntity`, although AOMD uses an object-oriented paradigm rather than using `void` pointers, which are type-unsafe. A skeleton UML structure of AOMD, illustrated in Figure 4.8, shows clearly the relationship between the attachable data types and mesh entities.

Other than the efficient use of C++ templates for speed [25, 134], AOMD is also memory efficient. The original SCOREC Mesh Database stores pointers from regions to all their vertices, edges, and faces, and similarly for pointers from faces to their edges and vertices, *etc.* Thus, there is a large number of pointers per element. However, one need not store all of the pointers. Rather, for example, one may store only the pointers to the $(d-1)$-dimensional entities (and optionally $(d+1)$-dimensional entities with an additional pointer from the 0-dimensional entities to

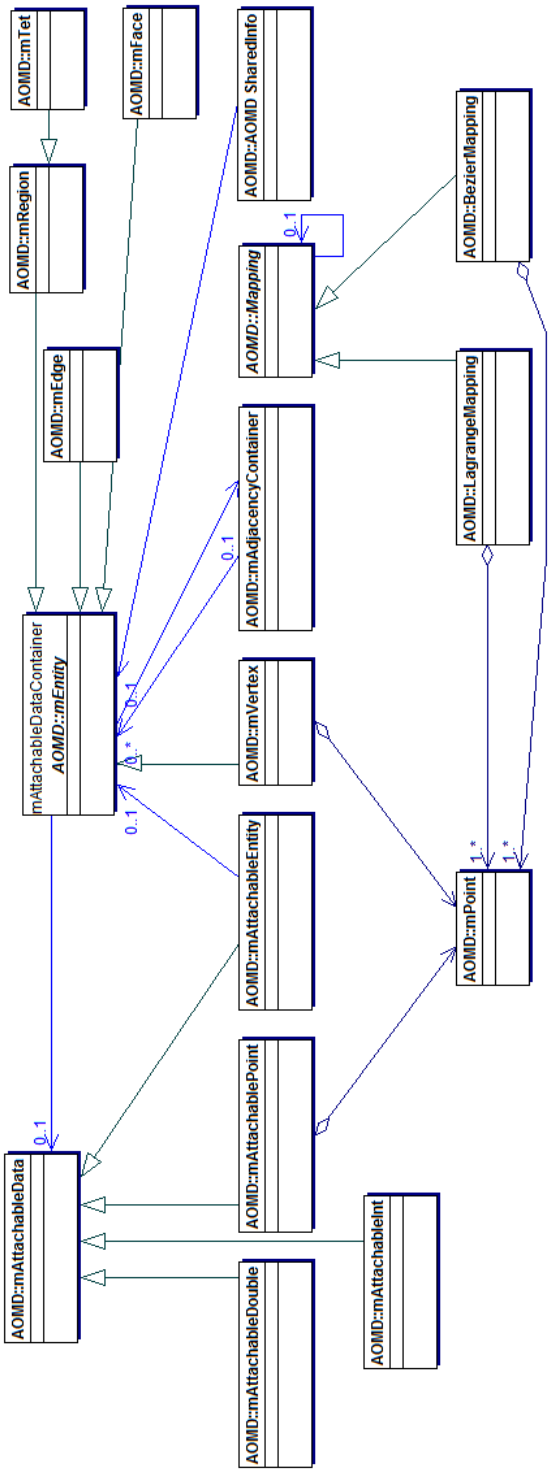

Figure 4.8: UML skeleton diagram of AOMD. Many AOMD classes were omitted for clarity.

the d-dimensional entities of the problem), where d is the dimension of the entity in question. This concept is shown in Figures 4.9 and 4.10.

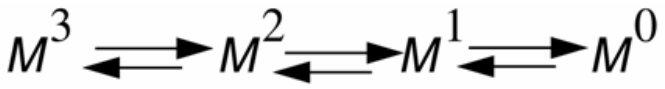

Figure 4.9: One level of adjacency pointers.

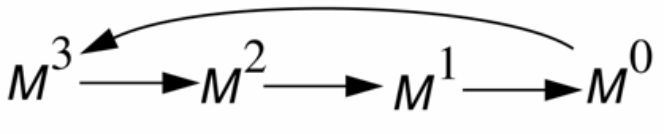

Figure 4.10: Circular adjacency pointers.

Storing just one level of adjacency pointers is sufficient, since one may loop over lower-dimensional entities to get the same information. For example, given a face, if one wishes to know what vertices are associated with the face, one loops over the list of edges, then for each edge, one can obtain the vertices on the edges. AOMD allows one to customize the adjacency information that is stored, thus greatly reducing memory [111].

AOMD not only stores and operates (such as creating and deleting elements) on meshes, it also has interfaces to various modelers, as noted at the beginning of this section. The modules for input of the mesh and model are separate from the rest of AOMD, so the interfaces for querying and manipulating the mesh are independent of the modeler used to create the domain. At this time, AOMD has only used meshes as fitting the exact geometry; however, work has progressed on interfacing AOMD to Parasolid and other modelers so that one may query the model entity containing a mesh entity, and *vice versa*.

4.1.7 Trellis Structure

Using AOMD as the data structure by which Trellis stores and interfaces to the mesh and model, we focus our attention on the structure of the analysis portion of Trellis. The solution field is coded in the subpackage `Field`, whose UML structure is shown in Figure 4.11.

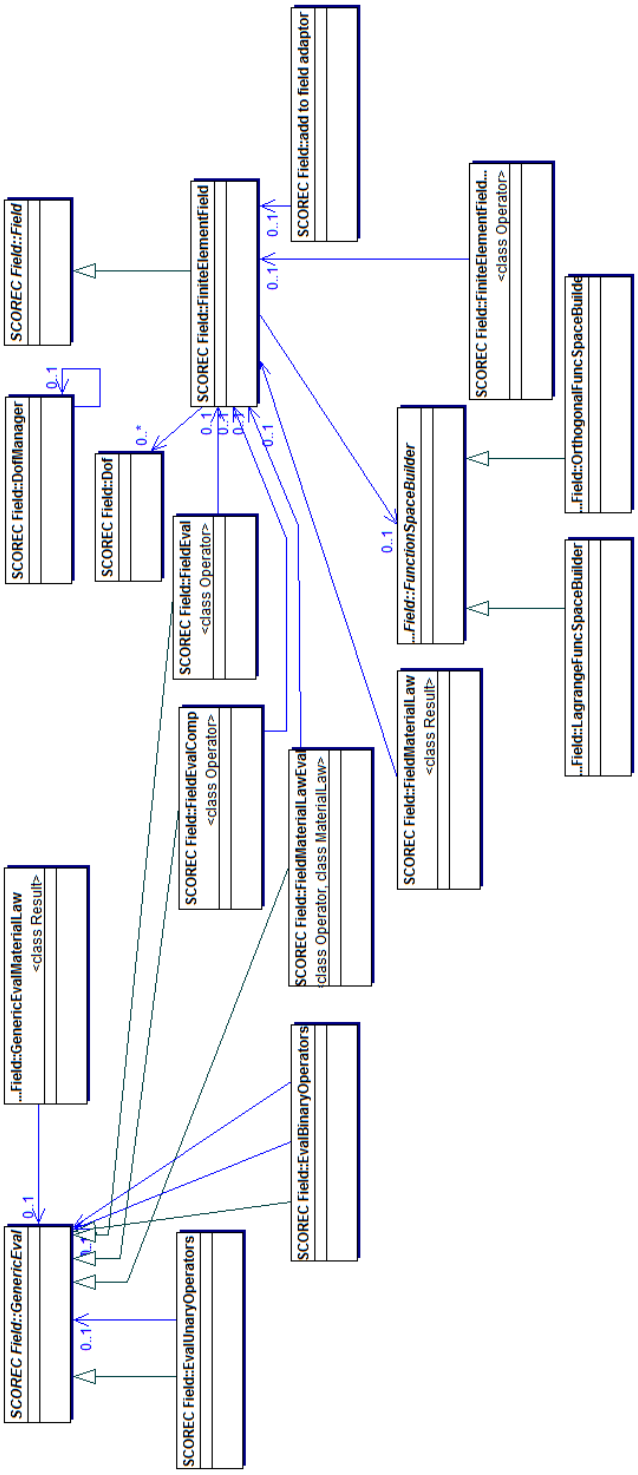

Figure 4.11: FiniteElementField UML diagram.

The FiniteElementField class manages an interpolating solution field. The classes derived from FunctionSpaceBuilder create the function space from a particular basis function set derived from the class FunctionSpace as shown in Figure 4.12.

All of the Builder classes are of the design pattern Builder found in [67]. The classes derived from GenericEval evaluate the field or operate on them in some way. For example, note how FieldMaterialLawEval (which is used to evaluate FieldMaterialLaw) is derived from GenericEval, but must also have a dependency on the FiniteElementField class.

The DofManager class, Figure 4.11, is of Singleton type [67] and manages the DOF in Trellis, keeping track of the number of DOF on each mesh entity. It also keeps track of the time history of the DOF, so that warm restarts, covered in §4.1.8, are possible. Notice that the FiniteElementField class depends on instances of the class Dof.

The connection between the Field package and AOMD is provided by the fact that the FiniteElementField depends on the class AOMD::MappingBuilder, which creates instances of classes derived from AOMD::Mapping. (This connection is not shown in the figures.) Thus, with the mapping defined over mesh entities, the finite element field is defined over the same entities.

The packages to do the analysis and solution are the Core and Solver modules, which are shown in Figure 4.13.

The classes derived from Assembler assemble the global matrices and vectors. The MatrixAssemblerDT class assembles the global matrix pertaining to terms of the form $(\partial u/\partial t, \phi_u)_\Omega$. This class is needed since the method by which one performs time integration determines how the time derivative term is approximated in the global matrix. The class MatrixAssemblerDT provides a backward Euler formulation. The class MatrixAssemblerDTDaspk, covered in §4.1.8, provides an interface to the DAE solver DASPK [37]. All of the assemblers depend on LinearSystemSolver, since it provides the top-level interface to various linear algebraic solving packages. Two package interfaces are shown in Figure 4.13, FullLU and SPOOLES, the latter described further in §4.1.8. Other linear algebraic packages to which there is an

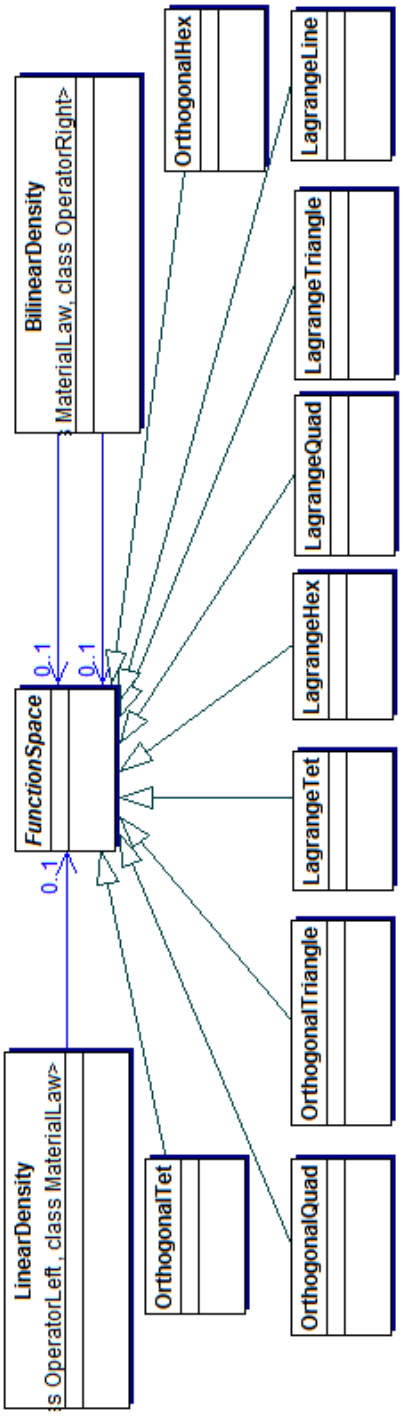

Figure 4.12: FunctionSpace UML diagram.

interface are Sparskit [117] and PETSc [17].

The `FiniteElementAnalysis` class contains the main routines to control the

solution of the finite element problem over one time step. The user must provide a routine in the `main` program that loops through time. Not shown in the diagram are routines to perform a nonlinear Newton solve. However, the call to this solver may be overridden, such as is done in the derived class `FiniteElementAnalysisDaspk`, to use the nonlinear solver in DASPK instead of Trellis'. However, regardless of the solver used, the `AlgebraicSystem` (and, therefore, the `Assembler`) interface is used to provide a consistent interface to the linear algebraic packages.

The connection between `Core` and `Field` is given by the association that `FiniteElementAnalysis` has an aggregate of `Field::Fields` classes, which contain a [C++ STL] `map` of class `Field::Field`, from which `FiniteElementField` classes are derived. (Not shown in the Figures.) There is also a connection (not shown) between `Core` and `AOMD`, since the `FiniteElementAnalysis` class has pointers to the mesh(es).

4.1.8 DASPK Incorporation, Mesh Motion, and Warm Restarts

Trellis was originally coded with only the backward Euler method for time integration. The incorporation of DASPK, which solves not only systems of stiff ODEs but also system of DAEs, while adaptively adjusting the time step and order of the method, is a significant upgrade to Trellis' capabilities. Unlike the backward Euler method where one specifies the time step, DASPK's input is the error tolerance, which is a more natural parameter.

Referring to Figure 4.13, `FiniteElementAnalysisDaspk`, derived from `FiniteElementAnalysis`, overrides the member functions `updateSolution` and `solveOneTimeStep` to interface with DASPK. The function `updateSolution` is overridden since DASPK provides the approximation of the time derivative of the solution as well as the solution itself, while the backward Euler method provides the approximate time derivative using the solution at the current and previous timestep. The `FiniteElementAnalysisDaspk` class also provides interfaces between DASPK and Trellis' linear algebraic solver and residual evaluation routines. The class `SimpleAlgebraicSystemDaspk` (name cut off in the Figure), derived from `SimpleAlgebraicSystem`, was needed so that `MatrixAssemblerDtDaspk`

is used instead of `MatrixAssemblerDt`. `MatrixAssemblerDtDaspk` is needed to pass DASPK's approximation of the time derivative of the solution rather than using a finite difference approximation as was done in `MatrixAssemblerDt` for the backward Euler method.

To improve the efficiency relative to DASPK, we modified the `Solver` package's `LinearSystemSolver` interface so that linear algebraic operations may be separated into two phases. Typically, the direct numerical solution of linear algebraic systems proceeds by first factoring (often in LU form) the system matrix then performing a forward-backward substitution to obtain the solution [69]. Recall that for a $n \times n$ matrix, the factorization time is $O(n^3)$ while the substitution time is $O(n^2)$. DASPK's nonlinear solver does not need to refactor the [global] linear algebraic system every time a solution is needed. Rather, DASPK waits until the rate of convergence of the nonlinear solver drops below a prescribed tolerance before refactoring the algebraic system. This ability to not have to factor the algebraic system at every iteration of the nonlinear solver greatly speeds up the computation. We have implemented an interface which transparently allows packages having the capability to separate out the factorization and substitution phase, such as SPOOLES (a sparse direct pivoting linear algebraic solver) [14], to take advantage of DASPK's factoring behavior. Other packages, such as Sparskit [117] which do not have this capability, will factor the algebraic system every time. In §4.2, the computations are performed using DASPK with SPOOLES.

One of the major improvements of Trellis relative to the initial version of Trellis [29], is its ability to store the solution at multiple times and project the solutions at all of the stored times from the old mesh onto an enriched mesh. Because of this, it was possible to implement warm restarts in DASPK. Warm restarts, investigated by Wang [137] in one- and two-dimensions for parabolic problems, are shown to greatly increase the solution method efficiency.

The idea of a warm restart is based on DASPK starting with a first-order ODE integration method and increasing the order as it moves forward in time. Recalling the backward differentiation formulas (BDF) method [13], higher-order methods require that the solution be known at previous time steps. Using higher-

order methods, such as BDF, is advantageous since for a given error tolerance, larger time steps may be used because the error for an order k BDF formula is $O\left(\Delta t^k\right)$. When performing $h-$ (or $p-$) adaptivity, the number of DOF changes. If the solutions at all the previous times are not transferred to the new mesh (or the solution is not projected onto the enriched solution space), then DASPK must start the next solution step using the first-order method with a much smaller time step rather than continuing with the current order method. Restarting from a first-order method is called a *cold restart*, while projecting the solution history and continuing with the current order method is called a *warm restart*. The prototype PDESE code (§3.2.3) was able to perform only cold restarts. However, in the implementation of the current version of Trellis, the DOF manager, class `DofManager`, has the ability to store the history of the solution vectors and to pass them to the projection routine during mesh and order enrichment, allows for warm restarts. It will be shown in §4.2.1 that using warm restarts improves the running time of one example by approximately a factor of four.

The class `FEADaspkMeshMove` derived from `FiniteElementAnalysisDaspk` adds mesh motion to the capabilities of Trellis. The user specifies which fields determine the coordinates of the domain. These field coordinates are used to update the mesh each time the solution is evaluated. Thus, the member function `updateSolution` is enhanced to call the mesh motion routine. A field is used to specify the deformation of the domain in time, so that the equations of motion, which are a part of the global system of equations which, are solved to the same accuracy in time as the rest of the system of PDEs.

4.1.9 Problem Specification

One of the requirements of a PDESE is the ease of specifying a problem, here a system of PDEs. This is readily accomplished in Trellis. For example, the forced heat problem (in weak form),

$$\left(\frac{\partial T}{\partial t}, \phi_T\right) + (\nabla T, \nabla \phi_T) + (sourceTerm, \phi_T) = 0, \tag{4.1}$$

where T is the temperature, may be specified in three lines, in order of the terms appearing in (4.1):

```
BilinearForm(M,...,Identity(),ScalarMaterialLaw(1.0),Identity(),
                    T, T, FIRST_DERIVATIVE,...);
BilinearForm(M,...,Grad(),ScalarMaterialLaw(1.0),Grad(),
                    T, T, ZERO_DERIVATIVE,...);
LinearForm(M,...,Identity(),sourceTerm,
                  T, ZERO_DERIVATIVE,...);
```

where extraneous parameters and namespaces were omitted. `M` specifies the model entity number on which the forms are applied, while `T` represents the temperature field. The `Identity()` and `Grad()` operators act on field `T`. The specification of zeroth and first derivatives indicates whether the form is acting on a time derivative of `T`. Comparisons between the code and the equations yields a self-evident mapping. Specifications of nonlinear terms in Trellis, hitherto not attempted, are not presented in this thesis, but examples of them are in the package `BarocasAnalysis` (*cf.* §4.2). Future work on problem specifications is mentioned in §5.3.

4.1.10 Class Interaction Sequence

A skeleton UML sequence diagram is shown in Figure 4.14. The diagram shows the interaction of classes (appearing at the top) in time going from top to bottom with the name of the call at the tail of the arrow. The call to `assembleSystem` in `updateRHS` has been omitted for clarity. The diagram is intended to show that over one time step, Trellis follows a traditional form of setting up the DOFs, imposing Dirichlet boundary conditions, numbering the DOF, setting up the algebraic system, getting the previous time step's solution, calling an ODE solver (DASPK) which in turns calls residual and Jacobian (`DMATD`) functions which assemble the global systems, updating the solution, and performing adaptivity. The `DofManager` class provides a simple interface for looping over the `Dof` class instantiations. Not shown are the calls through the matrix and vector assembler classes to class `Spooles` that provide a uniform interface to all available linear algebraic solvers.

4.2 Three-Dimensional Biomechanical Simulations

There is a lack of work on the solution of fully three-dimensional nonlinear biphasic equations for large deformations. Previously, the anisotropic biphasic equations of Barocas and Tranquillo were solved adaptively only in two dimensions [105].

We solve the biphasic equations of Barocas and Tranquillo (3.1)-(3.5) in three dimensions using Trellis. An analysis class derived from `FEADaspkMeshMove`, `BarocasBiphasic`, provides the equation specifications for the ABT equations. No symmetries are assumed so the computation is performed in Cartesian coordinates. Two problems are solved, an ICTA problem in a hexahedral domain and a wound problem, using Trellis. These problems cannot be reduced to a two-dimensional problem by geometric symmetries.

Currently, the only error indicator coded in Trellis estimates the error on an element Ω_e as

$$\eta_e = \int_{\Omega_e} \left(\sigma_{\Omega_e} - \nabla U\right)^2 \, d|\mathbf{x}| \tag{4.2}$$

where Ω_e is an element of the mesh, U is the finite element solution, and σ_{Ω_e} is the recovered gradient of the solution by a least squares fit of the gradient of the finite element solution over a nodal patch. This error indicator is known as the ZZ error estimator [145]. Similarly to §3.2, the finite element approximation of the concentration, C, is used as the solution variable U. The element size field, $h(\mathbf{x})$, is computed as the solution to the minimization problem

$$\min \left\{ \int_\Omega \frac{1}{h^d} \, d|\mathbf{x}| + \lambda \left(Tolerance + \sum_{e=1}^{N_\Delta} \int_{\Omega_e} h^d \frac{{\eta_e}^2}{{h_{old}}^d} \, d|\mathbf{x}| \right) \right\}, \tag{4.3}$$

where d is the dimension of the problem, λ is the Lagrange multiplier, $Tolerance$ is the desired tolerance, and h_{old} is the size of the length of the longest edge of element e in the original mesh. Given the size field h, the current mesh is then locally refined. Solving the minimization problem, it is theorized that this minimizes the number of elements given the error tolerance and that it equidistributes the error in the mesh. Due to concerns of an over-aggressive coarsening strategy in Trellis and that (4.2) is only a primitive error indicator, mesh coarsening was not performed in these calculations, unlike in §3.2.

4.2.1 Three-Dimensional ICTA Problem

In this section we solve a similar problem to that of §3.2.4.1, except that instead of a cylindrical domain, we start with a hexahedral domain. The initial domain of this problem is shown in Figure 4.15.

Similar to the cylindrical domain, the ends are fixed (no-slip face) while the rest of the faces are considered free faces. All of the parameters remain the same, except that we use the identity tensor in place of $\mathbf{\Omega}_c$.

The concentration is plotted at four representative times in Figures 4.16 to 4.19.

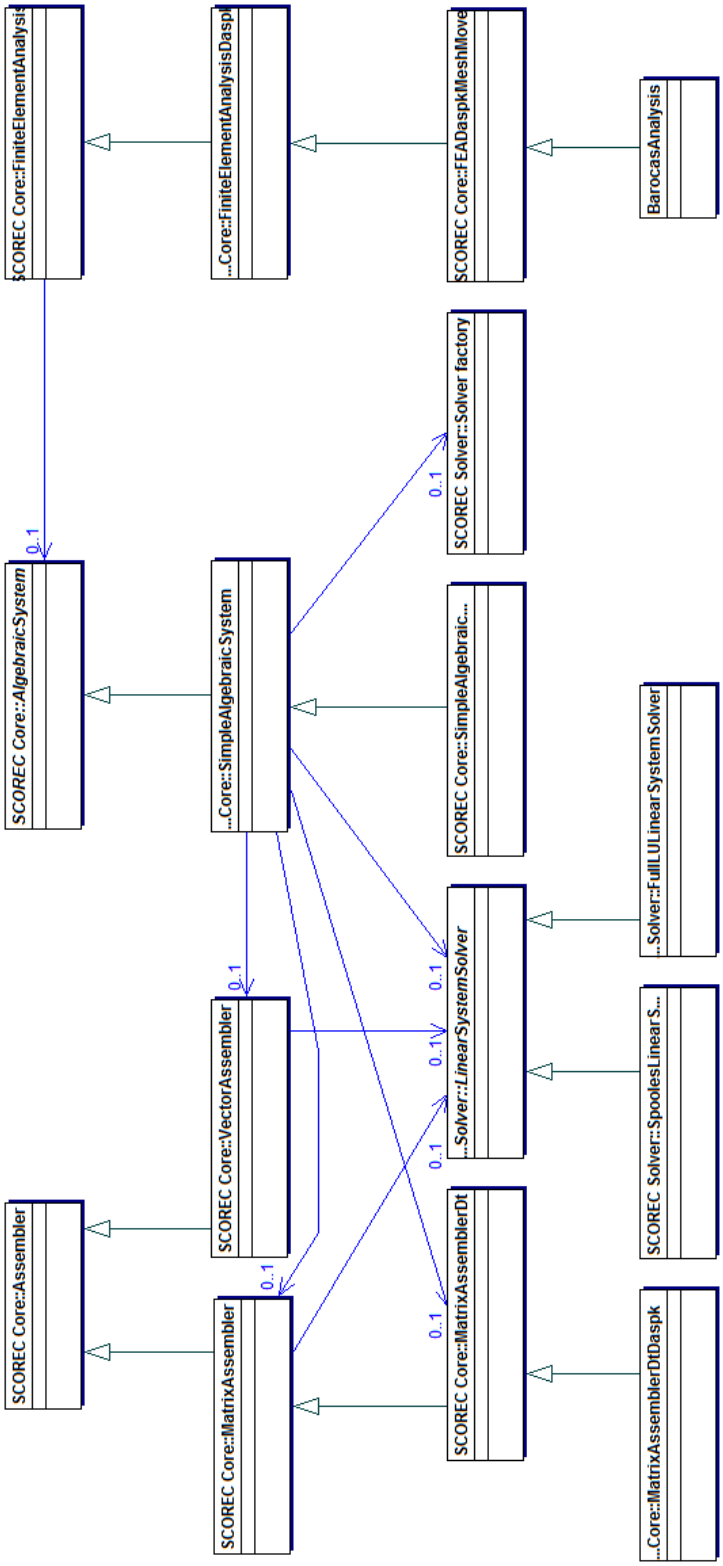

Figure 4.13: `Core` and `Solver` **UML diagram.**

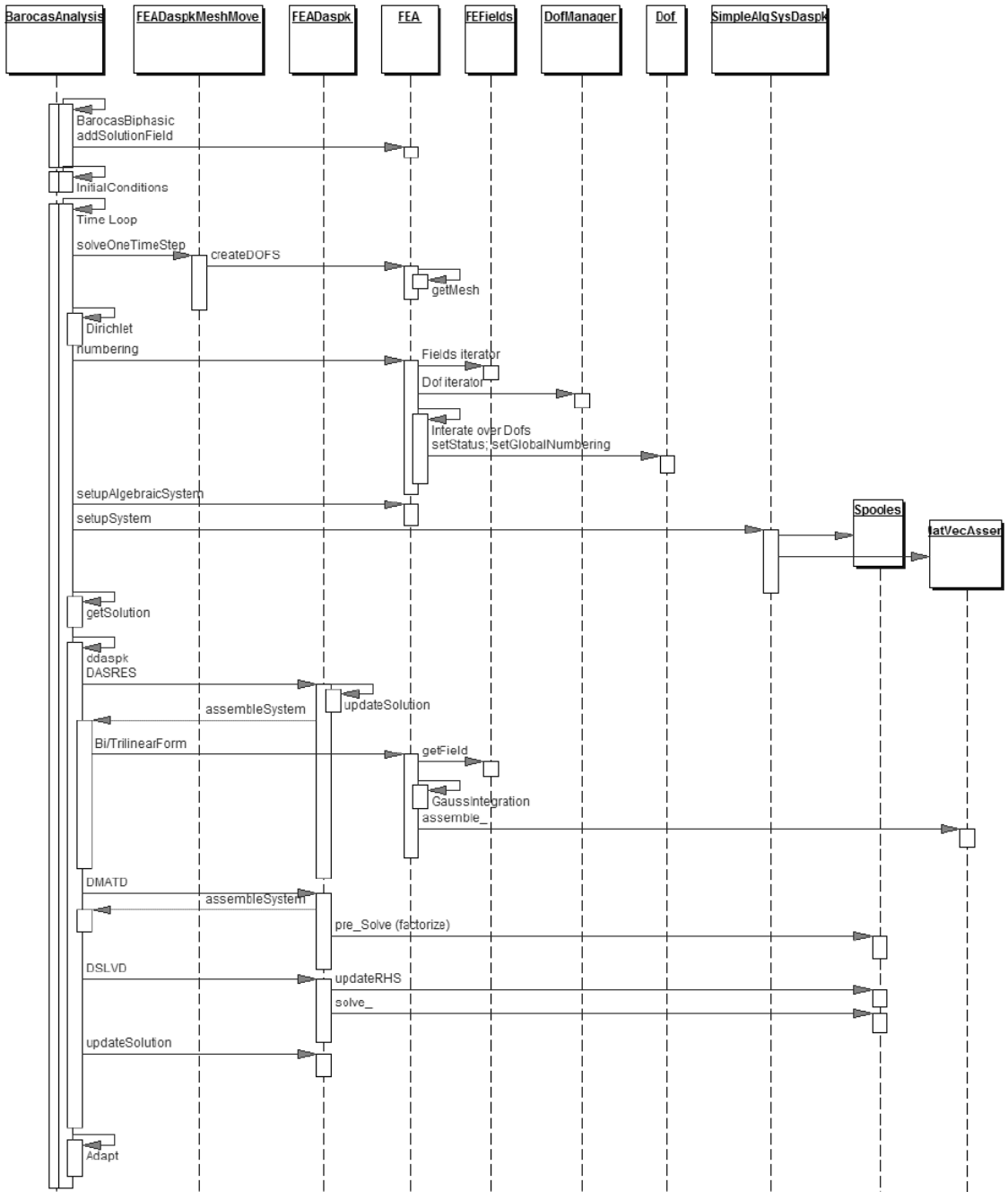

Figure 4.14: UML sequence diagram.

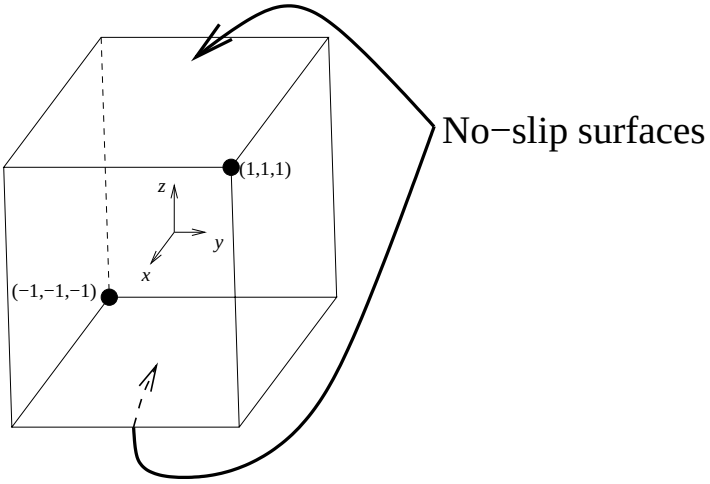

Figure 4.15: Hexahedral three-dimensional ICTA problem domain.

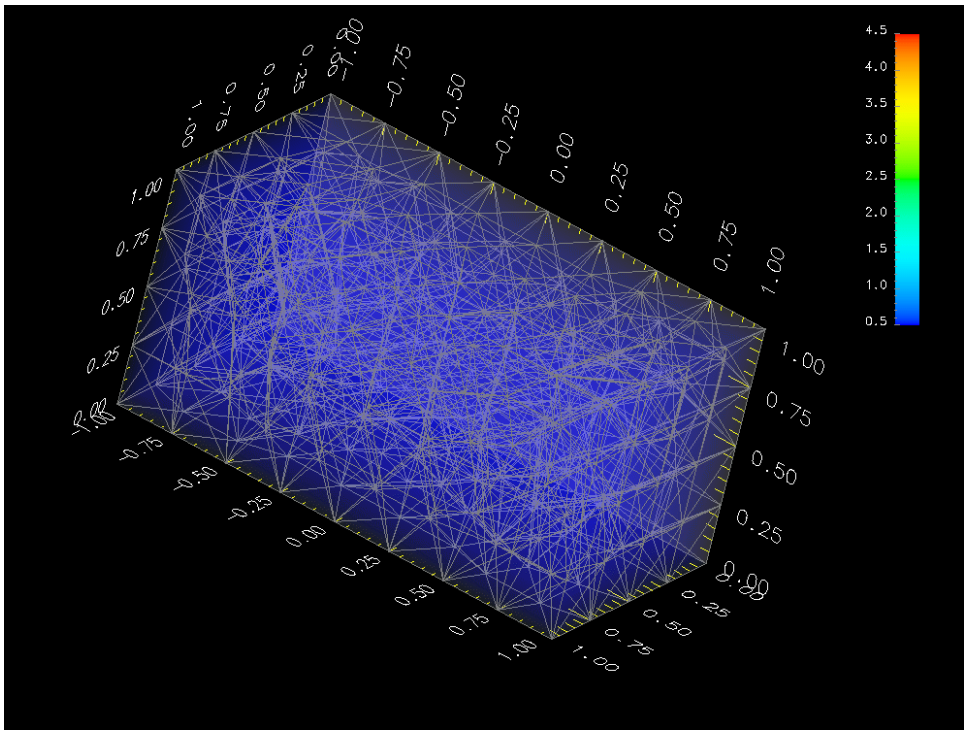

Figure 4.16: $t = 0$

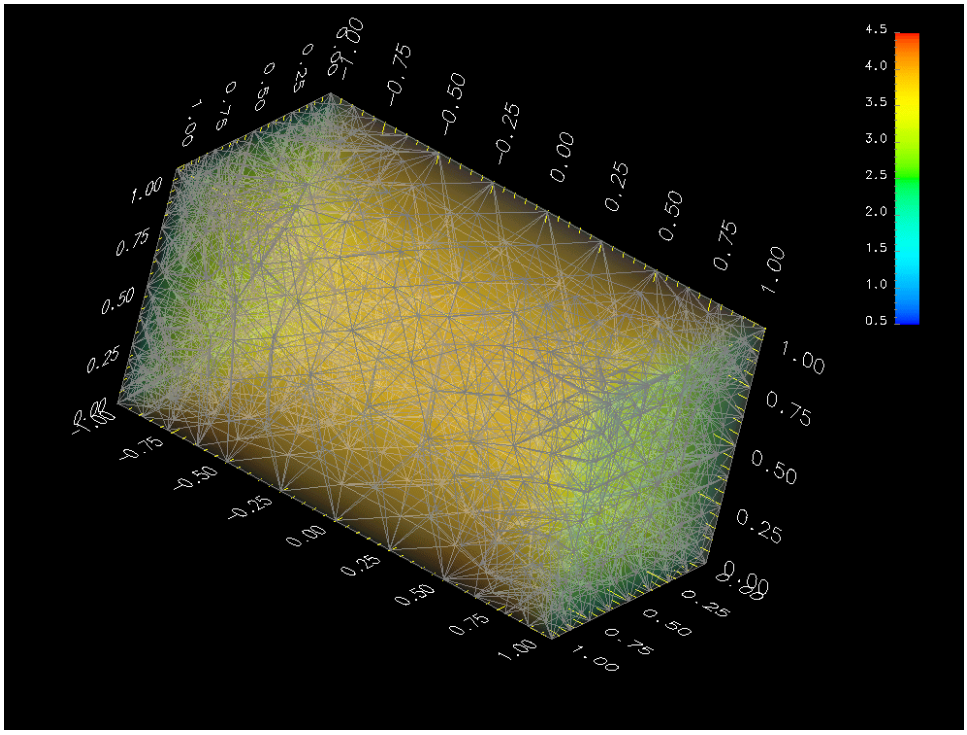

Figure 4.17: $t = 0.9$

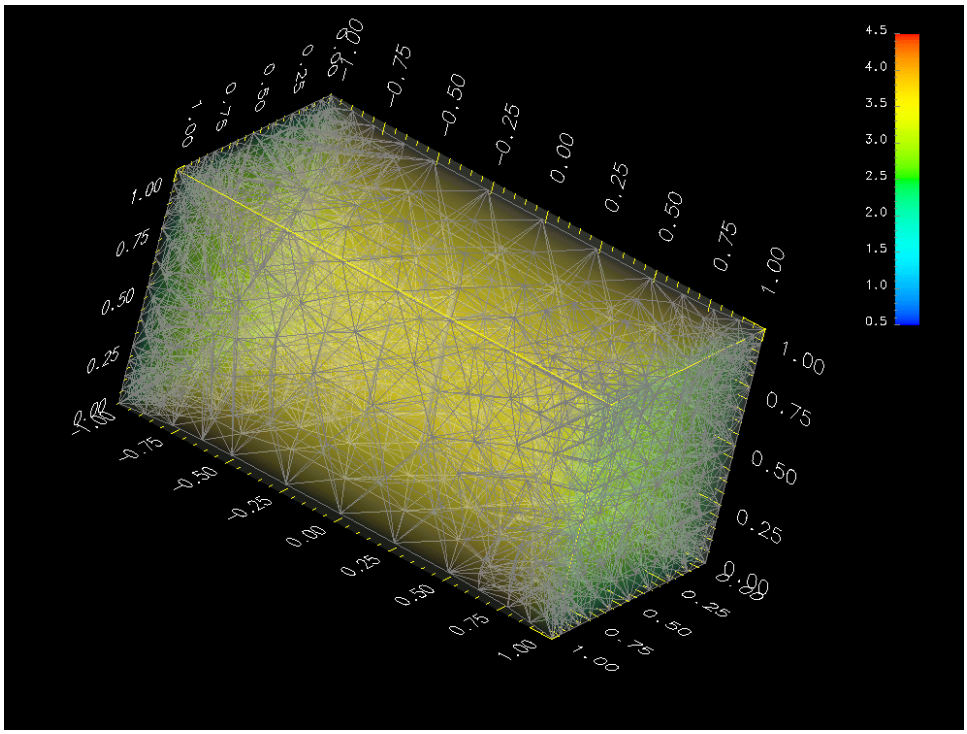

Figure 4.18: $t = 30.66$

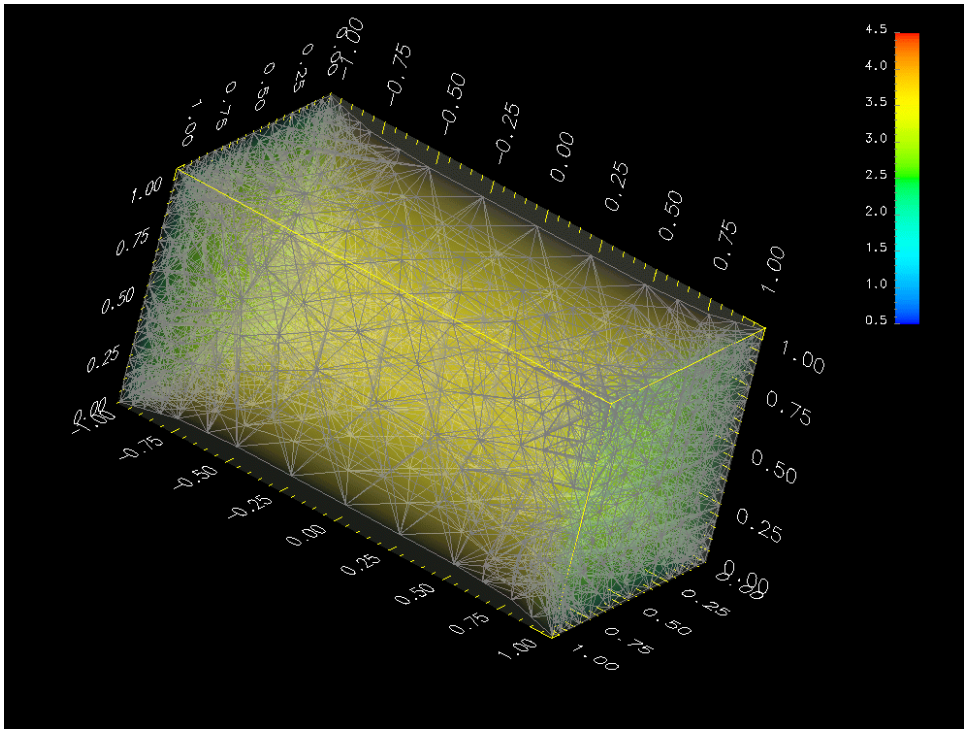

Figure 4.19: $t = 60$

As in the two-dimensional case, the mesh is progressively refined near the model faces that have no-slip boundary conditions.

We now examine the advantage of using warm restarts using the above problem. Since the frequency at which adaptivity occurs would be different, we set up the error tolerances for the warm restart run so that on average it would have the same or higher number of DOF as the cold restart run. The number of DOF versus t for both the cold restart and warm restart runs is plotted in Figure 4.20.

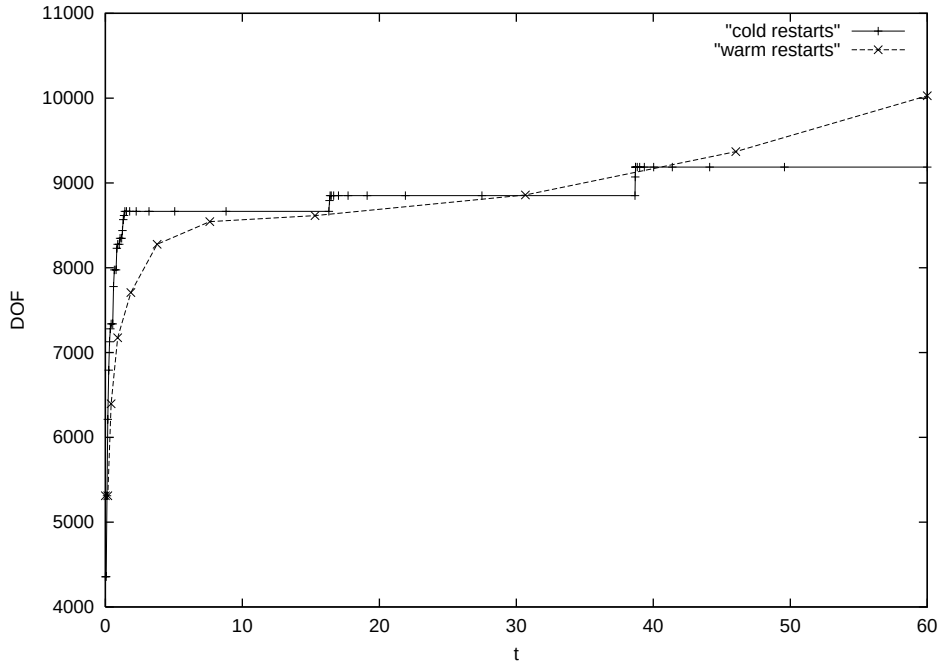

Figure 4.20: DOF versus time.

The time step of warm restarting is typically much larger than that of a cold restart run, because the cold restart run must reduce the time step to a small value after every mesh enrichment. The time step is plotted versus time in Figure 4.21.

The large time steps translate to a significant speed up as shown in Table 4.1.

Type	$\int_0^{60} DOF\, dt$	Average # DOF	Running Time
cold restart	533575	8893	100 minutes
warm restart	536044	8934	27 minutes

Table 4.1: Warm restart results.

The average number of DOF is the second column divided by 60. As indicated,

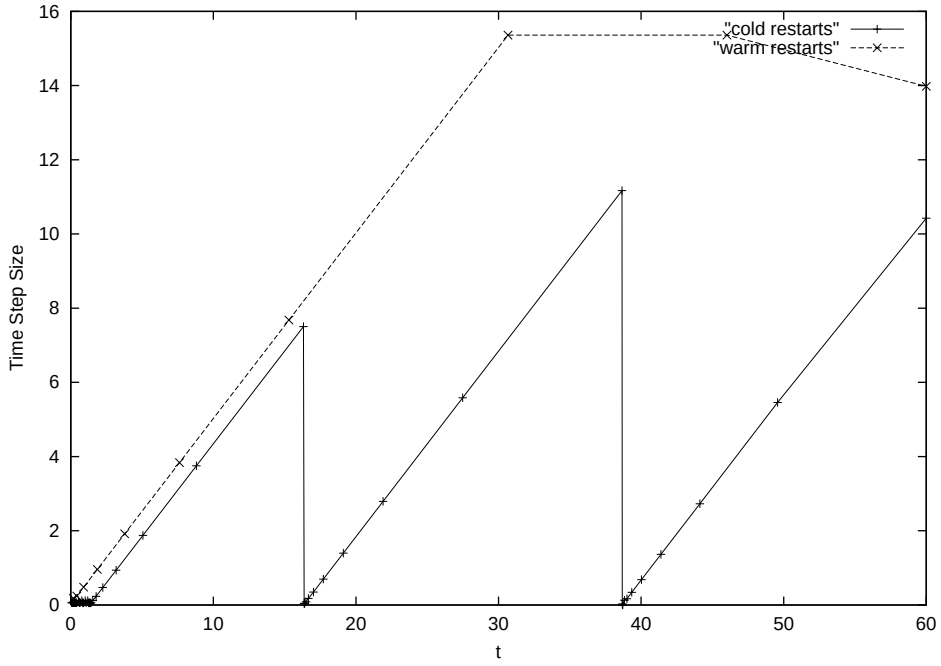

Figure 4.21: Time step versus time.

even with slightly more DOF, warm restarting took only almost one quarter the time of a run using cold restarts.

4.2.2 Three-Dimensional Wound Problem

We now solve the ABT equations, (3.1)-(3.5), for a new problem, simulating the healing of a wound [24]. The domain of this problem [24] is shown in Figure 4.22. Again, the same parameters were used with $\mathbf{\Omega}_c = \mathbf{I}$. However, the initial condition of c and θ was set to be

$$\begin{aligned} dist &= \left(x - \frac{3}{4}\right)^2 + \left(y - \frac{3}{4}\right)^2 \\ dist_{max} &= 2\left(\frac{3}{4}\right)^2 = \frac{9}{8} \\ c = \theta &= \frac{1}{2} + \frac{1}{\pi}\tan^{-1}(5(dist - dist_{max}/3)) \end{aligned} \tag{4.4}$$

This was done to simulate a wound, where the lack of cells in the middle of the domain represents a gouge in tissue. The arctangent function is used since a step

function would cause an incorrect DASPK behavior of reducing the time stepsize to roundoff (an error), due to an error norm which cannot handle non-diffusive discontinuous traveling wave solutions.

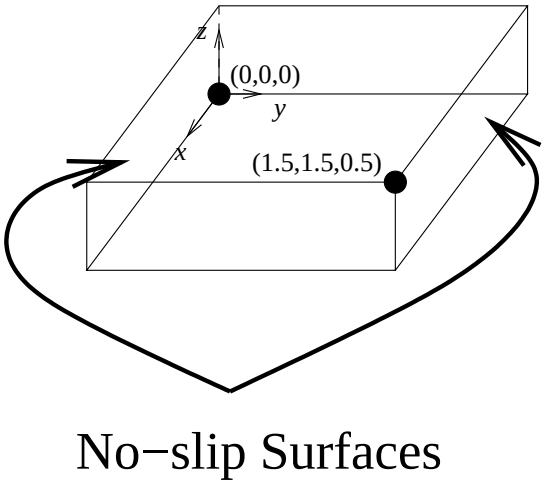

Figure 4.22: Wound problem domain.

The concentration is plotted at four representative times in Figures 4.23 to 4.26

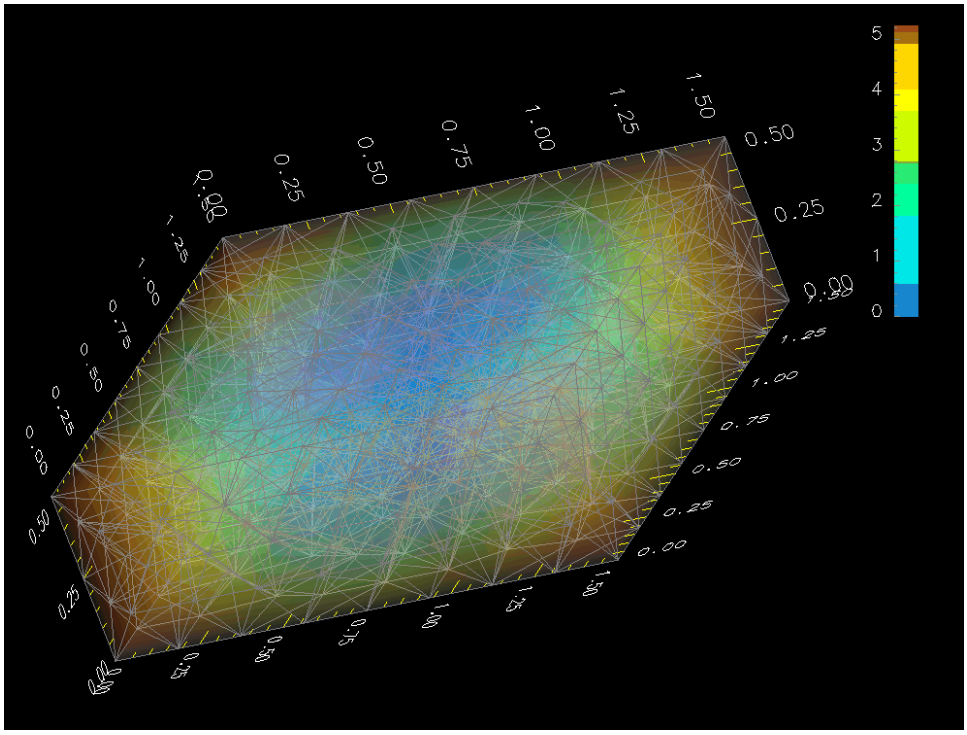

Figure 4.23: $t = 0$

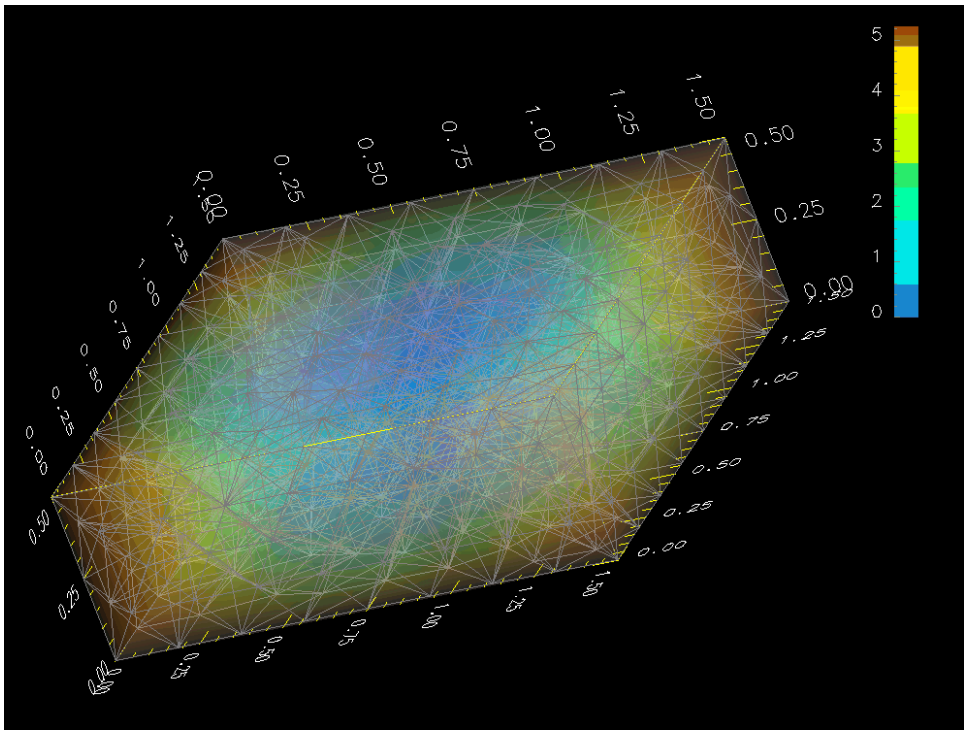

Figure 4.24: $t = 1.86$

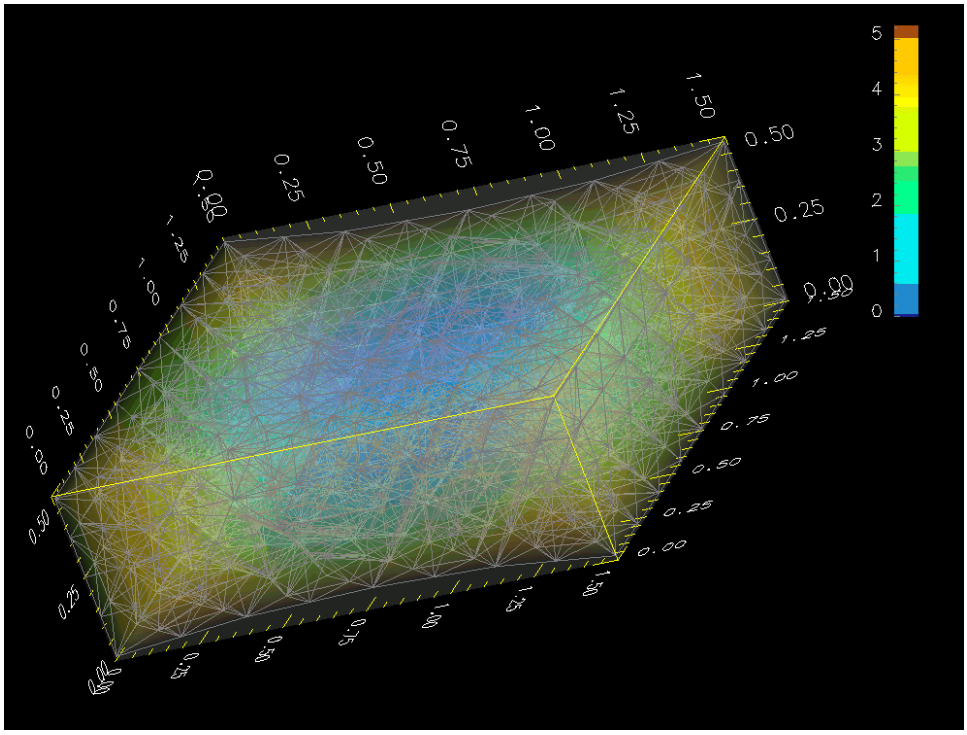

Figure 4.25: $t = 38.34$

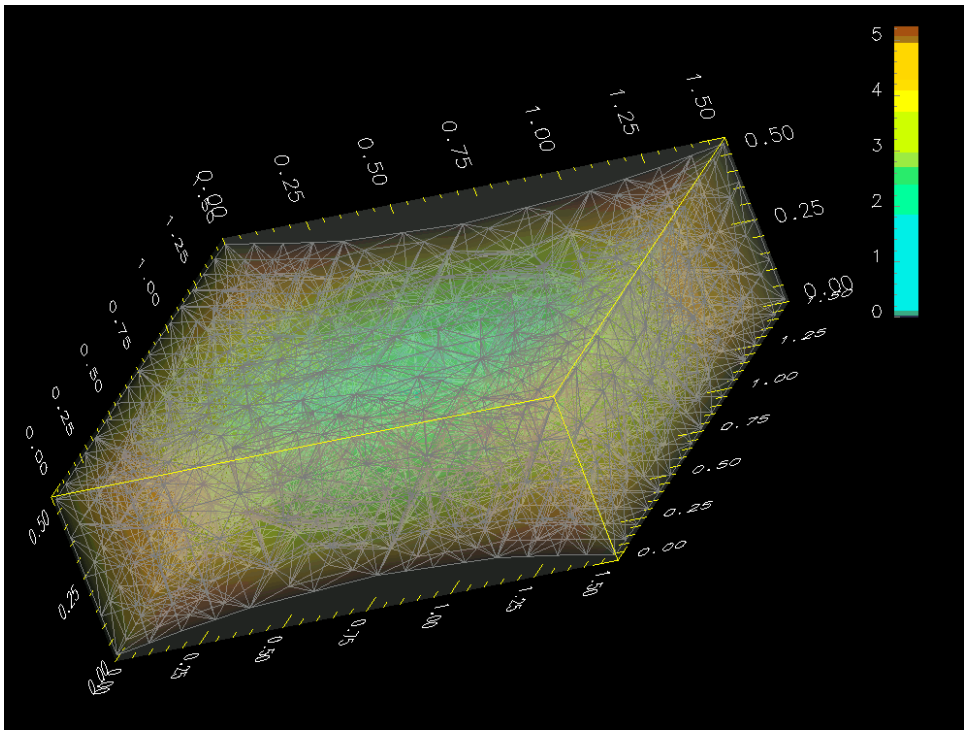

Figure 4.26: $t = 60$

As before, the mesh is refined in time, but unlike the problem in §4.2.1, the refinement is more uniform throughout the domain. This is because of the initial configuration, which results in some amount of gradient throughout much of the domain.

4.3 Discussion

Extending the object-oriented simulation framework Trellis to efficiently handle systems of DAEs by incorporating DASPK and SPOOLES, the ABT equations of Barocas and Tranquillo were solved for the first time in three-dimensions. While no exact comparisons can be made to the results obtained in §3.2, except for the minor note that the value of the concentration at $t = 60$ for the results in §4.2 is of the same order, the characteristics of the solution (having a sharp gradient where the no-slip boundary conditions meet the free boundary condition) and the pattern of refinement leads one to believe that the equations are being solved correctly. Both the three-dimensional ICTA problem in the hexahedral domain and the wound problem exhibit behavior consistent with are expected from the ABT equations.

CHAPTER 5
SUMMARY AND FUTURE WORK

5.1 Summary

In this thesis we have taken several significant steps toward creating a PDESE for biomechanical problems. First, we have developed a new efficient quadrature method, the table look-up method that is applicable for all orders, unlike the Gaussian quadrature method. Improvements in the data structure greatly reduced table size. While the optimal interpolation points are available only up to eighth order in three-dimensions, non-optimal interpolation points (*e.g.,* uniformly distributed points) may be generated up to all orders; thus, making this method applicable for higher order methods. The table look-up method is optimal for planar faced regions, though. Second, we have solved three biological and biomechanical problems, the ABT equations of Barocas and Tranquillo, the incompressible Navier-Stokes equations, and the nonlinear reaction-diffusion Lyme disease propagation equations with h-adaptivity and mesh motion. It was shown in the first two cases that using adaptivity greatly increases the efficiency of the computations. Third, we have collaboratively developed and extended a new object-oriented simulation framework, Trellis. Trellis now can solve systems of PDEs of mixed type, due to our incorporation of the DAE solver DASPK. Warm restarts were also implemented and shown to yield a significant improvement in efficiency. The ICTA problem in a hexahedral domain and the wound problem, which could not be solved before (since their geometry gives rise to no dimension-reducing symmetries), are solved using Trellis, in part due to the author's inplementation of some new nonlinear problems capabilty. Due to our work of providing some development and extending Trellis, it is closer to a complete PDESE. While the focus of this thesis was on biological and biomechanical problems, Trellis, as a PDESE, is capable of solving a large class of problems in a variety of fields.

Future work to be done is listed by topic in §5.2 through §5.3.

5.2 Table Look-Up

Future work should include attempting the table look-up method on more complex (nonlinear) problems, such as the Navier-Stokes equations for high p and many elements or the ABT equations. In this thesis, only $\Upsilon(\boldsymbol{\xi})$ was interpolated. It would also be worthwhile to consider interpolating $\theta(\boldsymbol{\xi})\Upsilon(\boldsymbol{\xi})$, where $\theta(\boldsymbol{\xi})$ is a factor of $\Theta(\boldsymbol{\xi})$. The choice of $\theta(\boldsymbol{\xi})$ would be problem dependent. Also, the order in which the table is accessed may be improved by reordering the integration so that it looks up only one part of the table at a time. This will allow caching to be more effective.

The flexibility of the table look-up method provides for a number of research opportunities in exploiting problem knowledge to decrease the expense of numerical integration in the finite element method. For example, one very important potential generalization of Trellis is incorporation of the generalized finite element methods (GFEMs) developed by Melenk and Babuška [96], where non-polynomial basis functions are used. While the cited paper gave good results for an artificial problem, the results should be extended to practical problems. One of the inefficiencies with current methods, such as GFEMs and meshless methods [32], is with integration procedures. The table look-up method may provide the key to exploiting these new methods.

It would be worthwhile to investigate using the table look-up method with non-polynomial basis functions where traditional methods are often too computationally expensive. The quadrature rule of Trellis (in AOMD) may be replaced by the table look-up method and new non-polynomial basis functions (supporting GFEMs) may be added. One possibility is to try non-polynomial basis functions which have a rational singularity in the derivative, *e.g.,* of the form $1/r^{\alpha}$, at one corner. This type of basis may be used to capture sharp gradients, such as those found in the ABT equations, without requring many mesh enrichments near the singularity. Furthermore, the singularity may be captured using far fewer DOF. However, if one used Gaussian quadrature instead of table look-up, then the method reduces to a polynomial interpolation problem, since Gaussian quadrature methods are interpolatory [83]. Thus, the two avenues: non-polynomial basis functions and table look-up, should be coded into Trellis simultaneously so that one may see some benefits.

5.3 Two- and Three-Dimensional Anisotropic Biphasic Equations

There are many issues remaining in solving the ABT equations both in two and three dimensions, as well as extensions to Trellis.

- $\mathbf{\Omega}_c$ was set to the identity matrix in three dimensions, since the routines to compute the nonlinear deformation tensor were unavailable at the time of this writing. A three-dimensional analog of $\mathbf{\Omega}_c$ used in §3.2 should be implemented in Trellis.

- Domains often have very different length scales. It may be worthwhile to use an anisotropic mesh adaptation similar to the work by Remacle *et al.* [113].

- *A posteriori* error estimation procedures should replace the error indicators of this study. They will enhance reliability. Currently, there is work underway to incorporate an equilibrated residual error estimator [104] into Trellis. However, since most users are interested in a specific variable or quantity of interest, it may be worthwhile incorporating error estimators that are computed on this quantity of interest. This will not be trivial since a dual problem needs to be set up and solved [11].

- Adaptive order enrichment, *p*- and *hp*-refinement, offer the possibility of exponential convergence rates and, as such, they should be explored. Optimal enrichment strategies are needed to decide when and where to alter the mesh and vary the order. *P*-adaptivity should be incorporated into Trellis.

- In this study, linear algebra was performed by a sparse direct procedure. Preconditioned Krylov iteration has been effective on the ABT system [22] and this should be attempted within the adaptive system. Trellis currently has interfaces to two iterative solver packages, PETSc and Sparskit, so the mechanism by which to incorporate preconditioners is present. At this time, there is no coupling between the algebraic solvers and the equation numbering system. This needs to be altered since many preconditioners take advantage of a

specific structure of the global matrix that cannot be determined through the `AlgebraicSystem` interface.

- The reinforced ABT study (§3.2.4.3) was only the initial step in an optimal design study. A combination of the adaptive solution and optimization software, COOPT [118], would be necessary for a more thorough investigation. The incorporation of COOPT into Trellis should be relatively straightforward, and should bring Trellis closer to the PDESE ideal.

- The ABT equations are actually twelve equations in three dimensions plus three equations for mesh motion. Due to the Babuška-Brezzi condition and the fact that in many parts of the domain the solution is smooth, higher order methods are used. This implies that the number of DOF increases rapidly with the number of elements. While the meshes in §4.2 are sufficient to capture the solution behavior in those instances, it may be necessary to use a large number of elements for other problems. Thus, it is very likely that this problem under Trellis needs to be run on parallel computers. Work is currently underway to parallelize Trellis. AOMD [112], PETSc [17], and SPOOLES [14] are packages which already incorporate support for parallel architectures. Even using parallel architectures, runs may take a long time (greater than one week), so a mechanism for checkpointing and restarting should be implemented.

- Work should commence on meshes where the element types include hexahedral elements as well as tetrahedral. Hexahedral elements have many desirable properties, including the fact that efficient quadrature rules exist for all orders by using a tensor product of one-dimensional Gauss quadrature rules.

- Work to make Trellis into a real PDESE is in order. Currently, the methods by which one specifies parameters, boundary conditions, and initial conditions are not user-friendly. Indeed, much of it is hard-coded. This will be remedied when the `Attributes` subpackage of `Core` is completed, which will allow one to read in an attribute file at the beginning of a run, specifying parameters and conditions. Furthermore, the handling of both input and output data

files is lacking in easy to use tools or support for standardized formats such as Hierarchical Data Format (HDF) [74]. Finally, the addition of more nonlinear operators should be incorporated. Correcting these shortcomings in Trellis as well as implementing the above work would result in a general purpose PDESE.

5.4 MEMS Pump

Many more experiments are still to be done. More two-dimensional experiments should be done, including changing the membrane forcing function. Also, a more realistic simulation can be done using Trellis in a three-dimensional setting. A two-dimensional membrane (unlike the one-dimensional membrane of §3.3) would allow for a greater variety of membrane functions. Even in two dimensions, more runs are needed to verify and extend the results of §3.3 to different frequencies. Since coding the incompressible Navier-Stokes equation is relatively straightforward in Trellis, it may be advantageous to redo the runs in Trellis. This should take far less time compared to using the prototype PDESE, due to the efficiencies of AOMD and the presence of warm restarts. Ultimately, it would be worthwhile using a PDESE (with efficient methods of solution) to consider the optimization problem of what type of forcing function would bring about a maximal efficiency over a range of pressure rises.

5.5 Lyme Disease Propagation

Many opportunities for further simulations exist. For example, the initial density of infected mice could be more realistic rather than an approximation of a sharp Gaussian. More realistic terrain (geometry) may be employed to simulate a particular area in which Lyme disease is to be simulated. Diffusion affected by terrain should also be investigated.

BIBLIOGRAPHY

[1] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, Tensor Analysis, and Applications*, 2nd edition, Springer-Verlag, 1988.

[2] M. Abramowitz and I. A. Stegun, editors, *Handbook of Mathematical Functions*, Dover Publications Inc., 1972.

[3] S. Adjerid and J. E. Flaherty, "A Moving Mesh Finite Element Method with Local Refinement for Parabolic Partial Differential Equations", Comp. Meths. Appl. Mech. Engrg., 55, pp. 3–26, 1986.

[4] S. Adjerid, M. Aiffa, J. E. Flaherty, J. B. Hudson, and M. S. Shephard, "Modeling and Adaptive Numerical Techniques for Oxidation of Ceramic Composites", Ceramic Engng. and Sci. Procs., 18, pp. 315–322, 1997.

[5] S. Adjerid, M. Aiffa, and J. E. Flaherty, "Hierarchical Bases for Triangular and Tetrahedral Elements", Comput. Meth. Appl. Mech. Engrg., 190, pp. 2925–2941, 2001.

[6] S. Adjerid, I. Babuška, and J. E. Flaherty, "A Posteriori Error Estimation for the Finite Element Method-of-Lines Solution of Parabolic Problems", Math. Models Meth. Appl. Sci., 9, pp. 261–286, 1999.

[7] S. Adjerid, B. Belguendouz, and J. E. Flaherty, "A Posteriori Finite Element Error Estimation for Diffusion Problems", SIAM J. Sci. Comput., 21, pp. 728–746, 1999.

[8] S. Adjerid, J. E. Flaherty, J. B. Hudson, and M. S. Shephard, "Modeling and the Adaptive Solution of CVD Fiber-Coating Processes", Proc. Amer. Ceram. Soc., 18, pp. 315–322, 1997.

[9] M. Aiffa, *Adaptive Hp-Refinement Methods for Singularly-Perturbed Elliptic and Parabolic Systems*, Rensselaer Polytechnic Institute, 1997.

[10] M. Ainsworth and J. T. Oden, "A Unified Approach to A Posteriori Error Estimation Using Element Residual Methods", Numer. Math., 65, pp. 23–50, 1993.

[11] M. Ainsworth and J. T. Oden, *A Posteriori Error Estimation in Finite Element Analysis*, John Wiley & Sons, Inc., 2000.

[12] G. C. Archer, *Object-Oriented Finite Element Analysis*, Ph.D. thesis, University of California at Berkeley, 1996.

[13] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM, 1998.

[14] C. Ashcraft and R. Grimes, "SPOOLES: An Object-Oriented Sparse Matrix Library", in "Proceedings of the 1999 SIAM Conference on Parallel Processing for Scientific Computing", 1999.

[15] H. L. Atkins and C.-W. Shu, "Quadrature-free implementation of discontinuous galerkin method for hyperbolic equations", *AIAA J.*, 36:775–782, 1998.

[16] I. Babuška, B. Andersson, B. Guo, J. M. Melenk, and H.-S. Oh, "Finite Element Method for Solving Problems with Singular Solutions", *J. Comput. Appl. Math.*, 74, pp. 51–70, 1996.

[17] S. Balay, W. D. Gropp, L. C. McInnes, B. F. Smith, "PETSc 2.0 Users Manual", technical report ANL-95/11 - Revision 2.0.22, Argonne National Laboratory, 1998.

[18] A. G. Barbour and D. Fish, "The Biological and Social Phenomenon of Lyme Disease", Science (Washington, D.C.), 260, pp. 1610–1616, 1993.

[19] V. H. Barocas, A. G. Moon, and R. T. Tranquillo, "The Fibroblast-Populated Microsphere Assay of Cell Traction Force — Part 2. Measurement of the Cell Traction Coefficient", J. Biomech. Engng., 117, pp. 161–170, 1995.

[20] V. H. Barocas, *Anisotropic Biphasic Modeling of Cell-Collagen Mechanical Interactions in Tissue Equivalents*, Ph.D. thesis, University of Minnesota, 1996.

[21] V. H. Barocas and R. T. Tranquillo, "An Anisotropic Biphasic Theory of Tissue-Equivalent Mechanics: the Interplay Among Cell Traction, Fibrillar Network Deformation, Fibril Alignment, and Cell Contact Guidance", J. Biomech. Engng., 119, pp. 137–145, 1997.

[22] V. H. Barocas and R. T. Tranquillo, "A Finite Element Solution for the Anisotropic Biphasic Theory of Tissue-Equivalent Mechanics: The Effect of Contact Guidance on Isometric Cell Traction Measurement", J. Biomech. Engng., 119, pp. 261–269, 1997.

[23] V. H. Barocas, T. S. Girton, and R. T. Tranquillo, "Engineered Alignment in Media-Equivalents: Consequences of Cell Induced Compaction on Magnetic Prelaignment", J. Biomech. Engng., 120, pp. 660-666, 1998.

[24] V. H. Barocas, private communications.

[25] J. J. Barton and L. R. Nackman, *Scientific and Engineering C++: An Introduction with Advanced Techniques and Examples*, Addison-Wesley, 1994.

[26] C. Batut, D. Bernardi, H. Cohen, and M. Olivier, *PARI*, Laboratory A2X, University of Bordeaux I, Cedex, FRANCE, 1997.

[27] M. W. Beall and M. S. Shephard, "A General Topology-Based Mesh Data Structure", Int. J. Num. Meth. Engng., 40, pp. 1573-1596, 1997.

[28] M. W. Beall and M. S. Shephard, "An Object-Oriented Framework for Reliable Numerical Simulations", Engrg. Comp., 15, pp. 61–72, 1999.

[29] M. W. Beall, *An Object-Oriented Framework for the Reliable Automated Solution of Problems in Mathematical Physics*, Ph.D. thesis, Rensselaer Polytechnic Institute, 1999.

[30] R. Beck, B. Erdmann, and R. Roitzsch, "An Object-Oriented Adaptive Finite Element Code: Design Issues and Applications in Hyperthermia Treatment Planning", in *Modern Software Tools for Scientific Computing*, edited by E. Arge, A. M. Bruaset, and H. P. Langtangen, Birkhäuser, 1997.

[31] E. Bell, B. Ivarsson, and C. Merrill, "Production of a Tissue-like Structure by Contraction of Collagen Lattices by Human Fibroblasts of Different Proliferative Potential in Vivo", Proc. of the National Academy of Sciences of the USA, 76, pp. 1274–1278, 1979.

[32] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl, "Meshless Methods: An Overview and Recent Developments", Comp. Meth. Appl. Mech. Engng., 139 pp. 3–47, 1996.

[33] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, 1994.

[34] R. B. Bird, W. E. Stewart, and E. N. Lightfoot, *Transport Phenomena*, John Wiley & Sons, Inc., 1960.

[35] G. Birkhoff, "Some Mathematica Problems of Numerical Ocean Acoustics", Comp. & Maths. with Appls., Vol. 11, No. 7/8, pp. 643–654, 1985.

[36] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley Longman, Inc., 1999.

[37] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, SIAM, 1996.

[38] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, 1991.

[39] W. J. Brown, R. C. Malveau, H. W. McCormick III, and T. J. Mowbray, *AntiPatterns*, Wiley, 1998.

[40] T. Caraco, S. Glavanakov, G. Chen, J. E. Flaherty, T. K. Ohsumi, and B. K. Szymanski, "Stage-Structured Infection Transmission and a Spatial Epidemic: A Model for Lyme Disease", Am. Nat., 160, pp. 348–359, 2002.

[41] Q. Chen and I. Babuška, "Approximate Optimal Points for Polynomial Interpolation of Real Functions in an Interval and in a Triangle", *Comp. Meth. Appl. Mech. Engng.*, 128, pp. 405–417, 1995.

[42] Q. Chen and I. Babuška, "The Optimal Symmetrical Points for Polynomial Interpolation of Real Fucntions in the Tetrahedron", *Comp. Meth. Appl. Mech. Engng.*, 133, pp. 319–346, 1996.

[43] P. Carnevali, R. B. Morris, Y. Tsuji, and G. Taylor, "New Basis Functions and Computational Procedures for p-version Finite Element Analysis", *I. J. Numer. Meths. Eng.*, 36, pp. 3759–3779, 1993.

[44] M. Carmona, S. Marco, J. Samitier, and J. R. Morante, "Dynamic Simulation of Micropumps", J. Micromech. Microeng., 6, pp. 128–130, 1996.

[45] Division of Vector-Borne Infectious Diseases, National Center for Infectious Diseases, Centers for Disease Control and Prevention, P. O. Box 2087, Fort Collins, Colorado 80522.

[46] A. J. Chorin and J. E. Marsden, *A Mathematical Introduction to Fluid Mechanics*, 3rd edition, Springer-Verlag, 1993.

[47] P. G. Ciarlet and P.-A. Raviart, "The Combined Effect of Curved Boundaries and Numerical Integration in Isoparametric Finite Element Methods", pp. 409–474, in *The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations*, edited by A. K. Aziz, Academic Press, 1972.

[48] R. Cools and P. Rabinowitz. "Monomial Cubature Since 'Stroud': a Compilation", *J. Comput. Appl. Math.*, 48, pp. 309–326, 1993.

[49] P. J. Davis and P. Rabinowitz, *Methods of Numerical Integration*, 2nd edition, Academic Press, Inc., 1984.

[50] D. DeCourtye, M. Sen, and M. Gad-el-Hak, "Analysis of Viscous Micropumps and Microturbines", Int. J. Comput. Fluid Dyn., 10, pp. 13–25, 1998.

[51] M. Dembo and F. Harlow, "Cell Motion, Contractile Networks, and the Physics of Interpenetrating Reactive Flow", Biophys. J., 50, pp. 109–121, 1986.

[52] L. Demkowicz, J. T. Oden, W. Rachowicz, and O. Hardy, "Toward a Universal h-p Adaptive Finite Element Strategy, Part 1: Constrained Approximation and Data Structure", *Comp. Meth. Appl. Mech. Engng.*, 77, pp. 79–112, 1989.

[53] K. D. Devine and J. E. Flaherty, "A Parallel Adaptive *hp*-refinement Techniques for Conservation Laws", *Appl. Num. Math.*, 6, pp. 1–20, 1996.

[54] S. Dey, *Geometry-based Three Dimensional hp Finite Element Modeling and Computations*, Ph.D. thesis, Rensselaer Polytechnic Institute, 1997.

[55] S. Dey, M. S. Shephard, and J. E. Flaherty, "Geometric Representation Issues Associated with p-version Finite Element Computations", Comp. Meth. Appl. Mech. Engng., 150, pp. 39–55, 1997.

[56] S. Dey, J. E. Flaherty, T. K. Ohsumi, and M. S. Shephard, "An Efficient Integration Method for p-Version Finite Elements in Curved Domains", In preparation, 2001.

[57] R. B. Dickenson, S. Guido, and R. T. Tranquillo, "Biased Cell Migration of Fibroblasts Exhibiting Contact Guidance in Oriented Collagen Gels", Ann. of Biomed. Engng., 22, pp. 342–356, 1994.

[58] D. F. D'Souza and A. C. Wills, *Objects, Components, and Frameworks with UML: the Catalysis Approach*, Addison Weley, 1999.

[59] M. Dubiner, "Spectral Methods on Triangles and Other Domains", em J. Sci. Comp., 6, pp. 345–390, 1991.

[60] D. A. Dunavant, "High Degree Efficient Symmetrical Gaussian Quadrature Rules for the Triangle", Int. J. Num. Meth. Engng., 21, pp. 1129–1148, 1985.

[61] F. Durst, A. Al-Salaymeh, and J. Jovanovic, "Theoretical and Experimental Investigations of a Wide-range Thermal Velocity Sensor", Meas. Sci. Tech., 12, pp. 223-237, 2001.

[62] G. Dwyer and J. S. Elkinton, "Host Dispersal and the Spatial Spread of Inset Pathogens", Ecology, 76, pp. 1262–1275, 1995.

[63] C. Dye and B. G. Williams, "Nonlinearities in the Dynamics of Indirectly-Transmitted Infections (or, Does Having a Vector Make a Difference?)", in *Ecology of Infectious Diseases in Natural Populations*, edited by B. T. Grenfell and A. P. Dobson, Cambridge University Press, 1995.

[64] H. P. Ehrlich and J. B. M. Rajaratnam, "Cell Locomotion Forces Versus Cell Contraction Forces for Collagen Lattice Contraction: An In Vitro Model of Wound Contraction", Tissue and Cell, 22, pp. 407–417, 1990.

[65] J. Fish and R. Guttal, "Recent advances in the p-version of the finite element method for shells", Computing Systems in Engineering, 6, pp. 195–211, 1995.

[66] G. J. Fix, "Effects of Quadrature Errors in Finite Element Approximation of Steady State, Eigenvalue, and Parabolic Problems", pp. 525–556, in *The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations*, edited by A. K. Aziz, Academic Press, 1972.

[67] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns*, Addison-Wesley, 1995.

[68] F. Grinnell, "Fibroblasts, Myofibroblasts, and Wound Contraction", J. Cell Biol., 124, pp. 401–404, 1994.

[69] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd edition, Johns Hopkins University Press, 1996.

[70] A. Gooray, G. Roller, P. Galambos, K. Zavaldi, R. Givier, F. Peter, J. Crowley, "Design of a MEMS Ejector for Printing Applications", J. Imaging Sci. Tech., 46(5), pp. 415–421, 2002.

[71] A. Grundmann and H. M. Möller, "Invariant Integration Formulas for the n-simplex by Combinatorial Methods", *SIAM J. Numer. Anal.*, 15, pp. 282–290, 1978.

[72] C. Guidry and F. Grinnell, "Contraction of Hydrated Collagen Gels by Fibroblasts: Evidence of Two Mechanism by Which Collagen Fibrils Are Stabilized", Collagen and Related Research, 6, pp. 515–529, 1986.

[73] S. E. Gursky and M. S. Sherman, "Yale Sparse Matrix Package — II. Nonsymmetric Codes", technical report tr114, , Dept. of Comp. Sci., Yale University, 1977.

[74] *HDF5*, The National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, 2000.

[75] O. Hededal, *Object-Oriented Structuring of Finite Elements*, Ph.D. thesis, Aalborg University, 1994.

[76] H. E. Hinnat, "A Fast Method of Numerical Quadrature for p-version Finite Element Matrices", *Int. J. Numer. Mech. Engng.*, 37, pp. 3723–3750, 1994.

[77] J. Hirai, K. Kanda, T. Oka, and T. Matsuda, "Highly oriented, Tubular Hybrid Vascular Tissue for a Low Pressure Circulatory System", ASAIO J., 40, pp. 383–388, 1994.

[78] E. E. Holmes, M. A. Lewis, J. E. Banks, and R. R. Veit, "Partial Differential Equations in Ecology: Spatial Interactions and Population Dynamics", Ecology, 75, pp. 17–29, 1994.

[79] D. Huang, T. R. Chang, A. Aggarwal, R. C. Lee, and H. P. Ehrlich, "Mechanisms and Dynamics of Mechanical Strengthening In Ligament-Equivalent Fibroblast-Populated Collagen Matrices", Ann. Biomed. Eng., 22, pp. 289–305, 1993.

[80] T. J. R. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Dover, 2000.

[81] V. L. Hughes and S. E. Randolph, "Testosterone Depresses Innate and Acquired Resistance to Ticks in Natural Rodent Hosts: a Force for Aggregated Distributions of Parasites", J. Parasitology, 87, pp. 49–54, 2001.

[82] A. V. Ilin, B. Bagheri, R. W. Metcalfe, and L. R. Scott, "Error Control and Mesh Optimization for High-Order Finite Element Approximation of Incompressible Viscous Flow", Comp. Meths. Appl. Mech. Engrg., 150, pp. 313–325, 1997.

[83] E. Isaacson and H. B. Keller, *Analysis of Numerical Methods*, Dover, 1994.

[84] Y. Jinyun, "Symmetric Gaussian Quadrature Formulae for Tetrahedral Regions", Comp. Meth. Appl. Mech. Engng., 43, pp. 349–353, 1984.

[85] C. Johnson, "Error Estimates and Adaptive Time-Step Control for a Class of One-Step Methods for Still Ordinary Differential Equations", SIAM J. Numer. Anal., 25, pp. 908–926, 1988.

[86] C. G. Jones, R. S. Ostfeld, E. M. Schauber, M. Richard, and J. O. Wolff, "Chain Reactions Linking Acorns to Gypsy Moth Outbreaks and Lyme-disease Risk", Science (Washington, D.C.), 279, pp. 1023–1026, 1998.

[87] , D. M. Knapp, V. H. Barocas, A. G. Moon, K. Yoo, L. R. Petzold, and R. T. Tranquillo, "Rheology of Reconstituted Type I Collagen Gel in Confined Compression", J. of Rheology, 41, pp. 971–993, 1997.

[88] M. S. Kolodney and E. L. Elson, "Correlation of Myosin Light Chain Phosphorylation with Isometric Contraction of Fibroblasts", J. of Biol. Chem., 268, pp. 23850–23855, 1993.

[89] M. U. Kopp, A. J. deMello, and A. Manz, "Chemical Amplification: Continuous-flow PCR On a Chip", Science, 280, pp. 1046–1048, 1998.

[90] A. R. Krommer and C. W. Ueberhuber, *Computational Integration*, SIAM, 1998.

[91] H. P. Langtangen, *Computational Partial Differential Equations, Numerical Methods and Diffpack Programming*, Springer-Verlag, 1999.

[92] J. Lawson, M. Berzins, and P. M. Dew, "Balancing Space and Time Errors In the Method of Lines for Parabolic Equations", SIAM J. Sci. Stat. Comput., 12, pp. 573–594, 1991.

[93] N. L'Heureux, L. Germain, R. Labbe, and F. A. Auger, "In Vitro Construction of a Human Blood Vessel from Cultured Vascular Cells: A Morphologic Study", J. of Vasc. Surg., 17, pp. 499–509, 1993.

[94] C. A. Lopez Valle, F. A. Auger, P. Rompre, V. Bouvard, L. Germain, "Peripheral Anchorage of Dermal Equivalents", Br. J. Dermatology, 127, pp. 365–371, 1992.

[95] J. A. Madri and B. M. Pratt, "Endothelial Cell-Matrix Interactions: In Vitro Models of Angiogenesis", J. Histochem. Cytochem., 34, pp. 85–91, 1986.

[96] J. M. Melenk and I. Babuška, "The Partition of Unity Finite Element Method: Basic Theory and Applications", Comput. Meths. Appl. Mech. Engrg., 139, pp. 289–314, 1996.

[97] M. B. Monagan *et al.*, *Maple V*, Waterloo Maple Inc., 1998.

[98] A. G. Moon and R. T. Tranquillo, "The Fibroblast-Populated Collagen Microsphere Assay of Cell Traction Force — Part 1. Continuum Model", AIChE J., 39, pp. 163–177, 1995.

[99] V. C. Mow, S. C. Kuei, W. M. Lai, and C. G. Armstrong, "Biphasic Creep and Stress Relaxation of Articular Cartilage in Compression: Theory and Experiments", J. Biomech. Engng., 102, pp. 73–84, 1980.

[100] J. D. Murray, E. A. Stanley, and D. L. Brown, "On the Spatial Spread of Rabies Among Foxes", Proceedings of the Royal Society of London B, Biological Sciences, 229, pp. 11-151, 1986.

[101] A. K. Noor and C. M. Andersen, "Computerized Symbolic Manipulation in Structural Mechanics-progress and Potential", Computer and Structures, 10, pp. 95–118, 1979.

[102] A. K. Noor and C. M. Andersen, "Computerized Symbolic Manipulation in Nonlinear Finite Element Analysis", Computer and Structures, 13, pp. 379–403, 1979.

[103] J. T. Oden, "Parallel Adaptive hp-finite Element Methods for Problems in Fluid and Solid Mechanics", pp. 29–35, in *Recent Developments in Finite Element Analysis*, edited by T. J. R. Hughes, E. Onate, and O. C. Zienkiewicz, 1994.

[104] J. T. Oden and M. Ainsworth, *A Posteriori Error Estimation in Finite Element Analysis*, John Wiley & Sons, Inc., 2000.

[105] T. K. Ohsumi, J. E. Flaherty, V. H. Barocas, S. Adjerid, and M. Aiffa, "Adaptive Finite Element Analysis of the Anisotropic Biphasic Theory of Tissue-Equivalent Mechanics", Comp. Meth. Biomech. Biomed. Engng., 3, 215-229, 2000.

[106] R. S. Ostfeld, O. M. Cepeda, K. R. Hazler, and M. C. Miller, "Ecology of Lme Disease: Habitat Association of Ticks *(Ixodes scapularis)* in a Rural andscape", Ecological Applications, 5, pp. 353–361, 1995.

[107] G. F. Oster, J. D. Murray, and A. K. Harris, "Mechanical Aspects of Mesenchymal Morphogenesis", J. Emb. Exp. Res., 78, pp. 83–125, 1983.

[108] L. Petzold, J. B. Rosen, P. E. Gill, L. O. Jay, and K. Park, "Numerical Optimal Control of Parabolic PDEs using DASOPT", Proc. IMA Workshop on Large-Scale Optimization, 1996.

[109] W. Rachowicz, J. T. Oden, and L. Demkowicz, "Toward a Universal h-p Adaptive Finite Element Strategy, Part 3, Design of h-p Meshes", *Comp. Meth. Appl. Mech. Engng.*, 77, pp. 181–212, 1989.

[110] S. E. Randolph, "Density-dependent Acquired Resistence to Ticks in Natural Hosts, Independent of Concurrent Infection with *Babesia microti*", Parasitology, 108, pp. 413–419, 1994.

[111] J.-F. Remacle and M. S. Shephard, "An Algorithm Oriented Mesh Database", Int. J. for Num. Meth. in Engng., in press, 2002.

[112] J.-F. Remacle, J. E. Flaherty, and M. S. Shephard, "A Parallel Algorithm Oriented Mesh Database", Engineering With Computers, in press, 2002.

[113] J.-F. Remacle, X. Li, N. Chevaugeon, and M. S. Shephard, "Transient Mesh Adaptation Using Conforming and Non Conforming Mesh Modifications", to appear in the Proceedings of the 11th Meshing Roundtable, 2002.

[114] J.-F. Remacle, J. E. Flaherty, and M. S. Shephard, "An Adaptive Discontinous Galerkin Technique with an Orthogonal Basis Applied to Compressible Flow Problems", SIAM Review, 45, pp. 53–72, 2003.

[115] J. Rice and R. Boisvert, "From Scientific Software Libraries to Problem Solving Environments", IEEE Comp. Sci. Engr., 3 , pp. 44–53, Fall 1996.

[116] M. P. Rossow and I. N. Katz, "Hierarchical Finite Elements and Precomputed Arrays, *Int. J. Numer. Meth. Engng.*, 12, pp. 997–999, 1978.

[117] Y. Saad, "SPARSKIT : A Basic Tool Kit for Sparse Matrix Computations", technical report RIACS-90-20, Research Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, CA, 1990.

[118] R. Serban and L. R. Petzold, "COOPT - A Software Package for Optimal Control of Large-Scale Differential-Algebraic Equation Systems", J. Math. Comp. in Simulation, 56, pp. 187–203, 2001.

[119] M. S. Shephard and M. K. Georges, "Automatic Three-Dimensional Mesh Generation by the Finite Octree Technique", Int. J. Num. Meth. Engng., 32, pp. 709–749, 1991.

[120] M. S. Shepard, S. Dey, and J. E. Flaherty, "A Straightforward Structure to Construct Shape Functions for Variable p-order Meshes", Comp. Meth. Appl. Mech. Engng., 147, pp. 209–233, 1997.

[121] S. J. Sherwin and G. E. Karniadakis, "A New Triangular and Tetrahedral Basis for High-order (hp) Finite Element Methods", Int. J. Numer. Meth. Engng., 38, pp. 3775–3802, 1995.

[122] P. Silvester, "Construction of Triangular Finite Element Universal Matrices", Int. J. Numer. Meth. Engng., 12, p.. 237–244, 1978.

[123] A. Spielman, M. L. Wilson, J. F. Levine, and J. Piesman, "Ecology of *Ixodes*-borne Human Babesiosis and Lyme Disease", Annual Review of Entomology, 30, pp. 439–460, 1985.

[124] E. Stemme and G. Stemme, "A Valve-less Diffuser/Nozzle Based Fluid Pump", Sensor and Actuators, 39, pp. 159–167, 1993.

[125] D. Stopak and A. K. Harris, "Connective Tissue Morphogenesis by Fibroblast Traction. I. Tissue Culture Observations", Dev. Biol., 90, pp. 383-398, 1982.

[126] G. Strang, "Variational Crimes in the Finite Element Method", pp. 689–710, in *The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations*, edited by A. K. Aziz, Academic Press, 1972.

[127] G. Strang and G. J. Fix, *An Analysis of the Finite Element Method*, Prentice-Hall, 1973.

[128] T. Strouboulis and K. A. Haque, "Recent Experiences with Error Estimation and Adaptivity, Part I: Review of Error Estimators for Scalar Elliptic Problems", Comp. Meths. Appl. Mech. Engrg., 97, pp. 399–436, 1992.

[129] R. W. Sutherest, K. B. W. Utech, M. J. Dallwitz, and J. D. Kerr, "Intraspecific Competition of *Boophilus microplus* (Canestrini) on Cattle", J. Appl. Ecology, 10, pp. 855–862, 1973.

[130] B. Szabo and I. Babuška, *Finite Element Analysis*, John Wiley & Sons, Inc., 1991.

[131] R. Temam, *Navier-Stokes Equations*, Elsevier Science Publishers B.V., 1984.

[132] T. E. Tezduyar, M. Behr, J. Liou, "A New Strategy for Finite Element Computations Involving Moving Boundaries and Interfaces — the Deforming-Spatial-Domain/Space-Time Procedure: I. The Concept and the Preliminary Numerical Tests", Comp. Meth. Appl. Mech. Engng., 94, pp. 339–351, 1992.

[133] R. T. Tranquillo, T. S. Girton, B. A. Bromberek, T. G. Triebes, and D. L. Mooradian, "Magnetically-Oriented Tissue-Equivalent Tubes: Application to a Circumferentially-Oriented Media-Equivalent", Biomaterials, 17, pp. 349–357, 1996.

[134] D. Vandevoorde and N. M. Josuttis, *C++ Templates: The Complete Guide*, Addison-Wesley, 2002.

[135] R. Verfürth, *A Review of Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, Teubner-Wiley, Stuttgart, 1996.

[136] R. Wait and A. R. Mitchell, *Finite Element Analysis and Applications*, John Wiley & Sons Ltd., 1985.

[137] Y. Wang, *An Adaptive Local HPR-Refinement Finite Element Method for Parabolic Partial Differential Equations*, Ph.D. thesis, Rensselaer Polytechnic Institute, 1991.

[138] L. M. Wilkins, S. R. Watson, S. J. Prosky, S. F. Meunier, and N. L. Parenteau, "Development of a Bilayered Living Skin Construct for Chemical Applications", Biotech. and Bioeng., 43, pp. 747–756, 1994.

[139] C. C. Wong, D. R. Adkins, G. C. Frye-Mason, M. L. Hudson, R. Kottenstette, C. M. Matzke, J. N. Shadid, and A. G. Salinger, "Modeling Transport in Gas Chromatography Columns for the Micro-ChemLab", SPIE Proceedings, 3877, pp. 120–129, 1999.

[140] J. Wu and W. Sansen, "The Glucose Sensor Integratable in the Microchannel", Sens. Actuat., 97-98, pp. 68–74, 2001.

[141] G. Yagawa, G.-W. Ye, and S. Yoshimura, "A Numerical Integration Scheme for Finite Element Method Based on Symbolic Manipulation", Int. J. Numer. Meth. Engng., 29, pp. 1539–1549, 1990.

[142] C. Yang, M. Kummel, and H. Soeberg, "A Transit-time Flow Meter for Measuring Milliliter Per Minute Liquid Flow", Rev. Sci. Instrum., 59, pp. 314–317, 1988.

[143] D.-H. Yu, "Asymptotically Exact A-Posteriori Error Estimators for Elements of Bi-Even Degree", Chinese J. Math. and Appl., 13, pp. 64–78, 1991.

[144] D.-H. Yu, "Asymptotically Exact A-Posteriori Error Estimators for Elements of Bi-Odd Degree", Chinese J. Math. and Appl., 13, pp. 82–90, 1991.

[145] O. C. Zienkiewicz and J. Z. Zhu, "A Simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis", Int. J. Numer. Methods Engng., 24, pp. 337–357, 1987.

[146] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method, Volume 1: The Basis*, 5th edition, Butterworth-Heinemann, 2000.

[147] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method, Volume 3: Fluid Dynamics*, 5th edition, Butterworth-Heinemann, 2000.

APPENDIX A
APPENDIX: Notation

The notation used in this thesis is listed below:

- Vectors are denoted in bold font.
- d denotes the spatial dimension.
- The spatial coordinate $[x_1, \ldots, x_d]^T$ is written as $\mathbf{x}$. In three-dimensions, $x_1 \equiv x$, $x_2 \equiv y$, and $x_3 \equiv z$.
- $dx_1\, dx_2\, dx_3$ (or $dx_1\, dx_2$ in two-dimensions) will be denoted as $d|\mathbf{x}|$.
- t denotes time.
- Ω is the problem domain.
- Ω_e is element e of the meshed domain.
- $\Gamma \equiv \partial\Omega$ is the boundary of domain Ω. Specifically, Γ_D is the part of the boundary where Dirichlet boundary conditions, $\alpha(\mathbf{x})$, are applied and Γ_N is the part of the boundary where Neumann boundary conditions, $\beta(\mathbf{x})$, are applied, where $\alpha(\mathbf{x})$ and $\beta(\mathbf{x})$ are user-specified functions. It is assumed that for scalar systems $\Gamma_D \cap \Gamma_N = \emptyset$.
- The outward normal to Ω for $d > 1$ is $\mathbf{n}$.
- h is the length of the longest edge in the mesh.
- T is the standard reference triangle or tetrahedron, depending on the dimension.
- The canonical coordinate in T, $[\xi_1, \ldots, \xi_d]^T$, is written as ξ.
- $d\xi_1\, d\xi_2\, d\xi_3$ (or $d\xi_1\, d\xi_2$ in two-dimensions) will be denoted as $d|\xi|$.

- The space of all functions whose k-th derivatives are all continuous on a domain Ω is denoted as $C^k(\Omega)$. The space of all smooth functions is denoted as $C^\infty(\Omega)$.
- $u_x \equiv \partial u/\partial x$, $u_{xy} \equiv \partial^2 u/\partial x \partial y$, *etc.*
- The gradient of a scalar function is

$$\nabla u = [u_{x_1}, \ldots, u_{x_d}]^T . \tag{A.1}$$

- The gradient of a vector function [34] $\mathbf{v} = [v_1, \ldots, v_d]^T$ is

$$\nabla \mathbf{v} = \begin{bmatrix} \frac{\partial v_1}{\partial x_1} & \cdots & \frac{\partial v_d}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial v_1}{\partial x_d} & \cdots & \frac{\partial v_d}{\partial x_d} \end{bmatrix}. \tag{A.2}$$

- The divergence of a vector function $\mathbf{v} = [v_1, \ldots, v_d]^T$ is

$$\nabla \cdot \mathbf{v} = \sum_{i=1}^{d} \frac{\partial v_i}{\partial x_i}. \tag{A.3}$$

- The Laplacian of a scalar function is

$$\nabla^2 u = \sum_{i=1}^{d} \frac{\partial^2 u}{\partial x_i^2}. \tag{A.4}$$

- The inner product of two scalar functions, u and v, on Ω is

$$(u, v)_\Omega = \int_\Omega u(\mathbf{x}) v(\mathbf{x})\, d\Omega \tag{A.5}$$

- The inner product of two vector functions, $\mathbf{u} = [u_1, \ldots, u_n]^T$ and $\mathbf{v} = [v_1, \ldots, v_n]^T$, on Ω is

$$(\mathbf{u}, \mathbf{v})_\Omega = \int_\Omega \sum_{i=1}^{n} u_i(\mathbf{x}) v_i(\mathbf{x})\, d\Omega \tag{A.6}$$

- The inner product of two tensor functions,

$$\mathbf{u} = \begin{bmatrix} u_{11} & \cdots & u_{1n} \\ \vdots & \ddots & \vdots \\ u_{n1} & \cdots & u_{nn} \end{bmatrix}, \qquad \text{and} \qquad \mathbf{v} = \begin{bmatrix} v_{11} & \cdots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n1} & \cdots & v_{nn} \end{bmatrix}, \tag{A.7}$$

  on Ω is

$$(\mathbf{u}, \mathbf{v})_\Omega = \int_\Omega \sum_{i=1}^{n} \sum_{j=1}^{n} u_{ij}(\mathbf{x}) v_{ij}(\mathbf{x}) \, d\Omega \tag{A.8}$$

- With a vector function $\mathbf{u}$ and scalar function v, define

$$\langle \mathbf{u}, v \rangle \equiv \int_\Gamma \mathbf{u} \cdot \mathbf{n} \, v \, d\mu, \tag{A.9}$$

  and when $\mathbf{u}$ is a tensor function and $\mathbf{v}$ is a vector function,

$$\langle \mathbf{u}, \mathbf{v} \rangle \equiv \int_\Gamma (\mathbf{u} \cdot \mathbf{n}) \cdot \mathbf{v} \, d\mu, \tag{A.10}$$

  where μ is a measure on Γ.

- When Ω is tessellated into a mesh, similar definitions of the inner product hold for integrals over mesh element Ω_e.

- The $L^2(\Omega)$-norm of a scalar function u is

$$\|u\|_{0,\Omega} = \sqrt{(u,u)_\Omega}. \tag{A.11}$$

  Thus, the set of all functions, u, such that $\|u\|_{0,\Omega} < \infty$ forms the Hilbert space, $L^2(\Omega)$.

- The Sobolev norm of a scalar function u is

$$\|u\|_{m,\Omega} = \sqrt{\sum_{|\alpha| \le m} \|D^\alpha u\|_{0,\Omega}^2}, \tag{A.12}$$

where

$$D^{\alpha}u = \frac{\partial^{|\alpha|}u}{\partial x_1^{\alpha_1}\partial x_2^{\alpha_2}\cdots\partial x_d^{\alpha_d}}, \tag{A.13}$$

for $\alpha = [\alpha_1, \ldots, \alpha_d]^T$, $\alpha_i \geq 0$ for $i = 1, 2, \ldots, d$, and $|\alpha| = \sum_{i=1}^{d} \alpha_i$. The set of all functions, u, such that $\|u\|_{m,\Omega} < \infty$ forms the Sobolev space, $H^m(\Omega)$.

- The space of functions, $v \in H^1(\Omega)$, restricted to Γ, is denoted as $H^{\frac{1}{2}}(\Omega)$.

- The sup-norm of a scalar function, u, is defined as

$$\|u\|_{\infty,\Omega} = \sup_{\mathbf{x}\in\Omega} |u(\mathbf{x})|. \tag{A.14}$$

Thus, the space of functions, u, such that $\|u\|_{\infty,\Omega} < \infty$ is the Banach space, $L^{\infty}(\Omega)$.

- Let

$$L_0^2(\Omega) = \left\{ u \mid u \in L^2(\Omega) \text{ and } u|_{\Gamma_D} = 0 \right\}. \tag{A.15}$$

Similarly, for $H_0^1(\Omega)$.

- Let

$$H_{BC}^1(\Omega) = \left\{ u \mid u \in H^1(\Omega) \text{ and } u|_{\Gamma_D} = 0 \text{ and } \left.\frac{\partial u}{\partial \mathbf{n}}\right|_{\Gamma_N} = \beta \right\}, \tag{A.16}$$

where $\beta \in H^{\frac{1}{2}}(\Omega)$ is the user-prescribed Neumann boundary conditions.

- The Ω will be omitted when the domain is obvious from the context.

- The norms are analogous for vector and tensor functions, $\mathbf{u}$.

- From the generalized form of Stokes Theorem [1] with scalar functions u and v,

$$\left(\frac{\partial u}{\partial x_i}, v\right)_{\Omega} = -\left(u, \frac{\partial v}{\partial x_i}\right)_{\Omega} + \int_{\Gamma} u\, v\, \mathbf{n}_i \, d\Gamma \tag{A.17}$$

where $\mathbf{n}_i$ is the i-th component of the unit vector normal to Ω.