Copyright
by
Roy Hulen Stogner
2008

The Dissertation Committee for Roy Hulen Stogner
certifies that this is the approved version of the following dissertation:

Parallel Adaptive C^1 Macro-Elements for Nonlinear Thin Film and Non-Newtonian Flow Problems

Committee:

Graham F. Carey, Supervisor

Irene M. Gamba

Omar Ghattas

David B. Goldstein

Thomas J.R. Hughes

Jim Stewart

Parallel Adaptive C^1 Macro-Elements for Nonlinear Thin Film and Non-Newtonian Flow Problems

by

Roy Hulen Stogner, B.S.

DISSERTATION

Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN
August 2008

Parallel Adaptive C^1 Macro-Elements for Nonlinear Thin Film and Non-Newtonian Flow Problems

Publication No. ____________________

Roy Hulen Stogner, Ph.D.
The University of Texas at Austin, 2008

Supervisor: Graham F. Carey

This research deals with several novel aspects of finite element formulations and methodology in parallel adaptive simulation of flow problems. Composite macroelement schemes are developed for problems of thin fluid layers with deforming free surfaces or decomposing material phases; experiments are also run on divergence-free formulations that can be derived from the same element classes. The constrained composite nature and C^1 continuity requirements of these elements raises new issues, especially with respect to adaptive refinement patterns and the treatment of hanging node constraints, which are more complex than encountered with standard element types. This work combines such complex elements with these applications and with parallel adaptive mesh refinement and coarsening (AMR/C) techniques for the first time.

The use of adaptive macroelement spaces also requires appropriate programming interfaces and data structures to enable easy and efficient implementation in parallel software. The algorithms developed for this work are implemented using object-oriented designs described herein.

One application class of interest concerns heated viscous thin fluid layers that have a deformable free surface. These problems occur in both normal scale laboratory and industrial applications and in micro-fluidics. Modeling this flow via depth averaging gives a nonlinear boundary value problem describing the transient evolution of the film thickness. The model is dominated by surface tension effects which are described by a combination of nonlinear second and fourth-order operators.

This research work also includes studies using the divergence-free forms constructed from these elements for certain classes of non-Newtonian fluids such as the Powell-Eyring and Williamson shear-thinning viscosity models. In addition to the target problems we conduct verification studies in support of the simulation development.

In the final application investigated, C^1 elements are used in conforming finite element approximations of the Cahn-Hilliard phase field model for moving interface and phase separation problems. The nonlinear Cahn-Hilliard equation combines anti-diffusive configurational free energy based terms with a fourth-order interfacial free energy based term. Numerical studies include both manufactured and physically significant problems, including parametric studies of directed pattern self-assembly in phase decomposition of thin films.

The main new contributions include construction of C^1 and div-free macroelement classes suitable for AMR/C with nonconforming hanging node meshes; *a posteriori* error estimation for fourth-order problems using these and other element classes; use of projection operators to automate the correct treatment of constraints at hanging nodes and through AMR/C steps; design of supporting data structures and algorithms for implementation in a parallel object oriented framework; variational formulations, methodology and numerical experiments with nonlinear fourth-order flow and transport models; and parametric and Monte Carlo studies of directed phase decomposition.

Table of Contents

Abstract **iv**

List of Figures **x**

Chapter 1. Introduction **1**

1.1 Research Motivation 1
1.2 Chapter Overview 5
1.3 Contributions 6

Chapter 2. Automatic Macroelement Generation and Use **10**

2.1 Introduction 10
2.2 Macroelements 12
2.3 Master Basis Function Derivation 14
2.4 Global Degree of Freedom Calculation 17
2.5 A Priori Error Estimates 18
2.6 Adaptive Refinement on Macroelements 22

Chapter 3. Adaptive Mesh Refinement/Coarsening for Fourth-order Problems **26**

3.1 Adaptive Meshes and Continuity 27
3.2 Projection Operators 30
3.3 A Posteriori Error Estimation 32
3.4 Adaptive Refinement for Steady Problems 41
3.5 Adaptive Refinement for Transient Problems 42
  3.5.1 Adaptive Time Discretization 45
3.6 Biharmonic Benchmark Problem 47

Chapter 4. Software Design **53**

4.1 Finite Element Library 53
4.2 Object Oriented Design Abstractions 54
4.3 Parallel Implementation 62
4.4 Newton-Krylov Methods 71

Chapter 5. Divergence-free Flow Problems **79**
5.1 Introduction 79
5.2 Divergence-free finite element spaces 79
5.3 Solution Existence and Uniqueness 81
5.4 Incompressible Navier-Stokes Flow 84
5.5 Shear-dependent Viscosity Models 85
5.6 Penalty Boundary Conditions 87
5.7 Successive Approximation 87
5.8 Continuation 88
5.9 Newton-Krylov Methods 90
5.10 Time-dependent flow 92
5.11 Unsteady Flow and Transport 94
5.12 Divergence-free Flow Example Results 95
5.13 Transport Results 99

Chapter 6. Surface Tension Driven Flow of Thin Films **100**
6.1 Introduction 100
6.2 Flow and transport equations 102
6.3 Galerkin formulation 108
6.4 Thin Film Flow Results 110

Chapter 7. Cahn-Hilliard Phase Separation **111**
7.1 Introduction 111
7.2 Cahn-Hilliard Equation 111
7.3 Lyapunov Energy Functional 114
7.4 Galerkin Finite Element Approximation 116
7.5 Lyapunov Energy Functional and Spatial Discretization 118
7.6 Lyapunov Energy Functional and Crank-Nicolson Time Discretization 120
7.7 Cahn Hilliard Evolution 124
7.8 Cross Interface 124
7.9 Spinodal Decomposition 126
7.10 Mesh Refinement 131
7.11 Surface Patterning 133
7.12 3D Thin Film Patterning 138

7.13 Solution Postprocessing 141
7.14 Parametric Studies 144
7.15 Simulation Performance 176

Chapter 8. Concluding Remarks **187**

Bibliography **191**

Vita **210**

List of Figures

1.1 Left: A pattern with defects from the experimental literature, showing phase separated deuterated polystyrene and polybutadine on a monolayer substrate [79]. Right: A Cahn-Hilliard simulation from the present study with a weak spatially-dependent surface affinity term, showing similar qualitative behavior. . . 7

2.1 A Hsieh-Clough-Tocher 3-split, a Powell-Sabin-Heindl 12-split, and two adjacent Powell-Sabin 6-split triangles, with value degrees of freedom labeled by circles and derivative degrees of freedom labeled by arrows. Note the mesh-dependent splitting required for the 6-split triangle type. 13

2.2 Derivatives in cartesian directions on the master element may correspond to derivatives in skewed directions on a physical element. 17

2.3 Galerkin approximation error in L_2 and H^1 norms, scaled by the solution norm and evaluated on uniform meshes for the manufactured benchmark biharmonic problem. 21

2.4 A typical hanging node between Hsieh-Clough-Tocher or Powell-Sabin-Heindl 12-split triangles. The coarse triangle nodal degrees of freedom (purple) match those of the refined neighbors, but the refined hanging node degrees of freedom (red) do not match the coarse normal derivative degree of freedom (blue) at the node, and the refined normal derivative degrees of freedom have no corresponding coarse degrees of freedom at all. 23

2.5 An isotropic adaptive refinement of a Powell-Sabin-Heindl 12-split triangle. The subelement node at the edge midpoint of the coarse triangle must also be a node of the neighboring refined triangles. 24

3.1 Streamfunction (left) and vorticity (right) for inviscid flow around a sharp edge. 48

3.2 Galerkin approximation error in L_2, H^1, and H^2 norms, scaled by the solution norm and evaluated on both uniform and adaptive meshes for the cusp flow problem. 51

3.3 An adaptively refined mesh obtained for the cusp flow problem. 52

4.1 A simplified UML diagram of part of the `libMesh` geometric element hierarchy. 56

4.2 A simplified UML diagram of part of the `libMesh` quadrature rule class hierarchy. 57

4.3 A simplified UML diagram of part of the `libMesh` finite element class hierarchy. 58

4.4 A simplified UML diagram including the `libMesh` FEMSystem boundary value problem abstract base class. Application code authors write physics-specific subclasses of FEMSystem to implement particular mathematical models. 59

4.5 A simplified UML diagram of part of the `libMesh` ODE solver class hierarchy. 59

4.6 A simplified UML diagram of part of the `libMesh` nonlinear algebraic solver class hierarchy. 60

4.7 SerialMesh vs. ParallelMesh per-node resident memory usage, for a benchmark application using uniformly refined meshes on varying numbers of processors. 66

4.8 The core loops in a Newton-Krylov solver. 73

4.9 The basic steps in a Brent-based line search for optimal substep length. 77

5.1 Domain and boundary conditions for lid-driven cavity flow . . 96

5.2 Steady-state streamfunction and vorticity plots for Newtonian lid-driven cavity flow at Reynolds number 400 96

5.3 Steady-state streamfunction and vorticity plots for Newtonian lid-driven cavity flow at Reynolds number 1000 97

5.4 Steady-state vorticity plot for Extended Williamson lid-driven cavity flow at Reynolds number 500 and viscosity ratio 0.1. . . 98

5.5 Error convergence, plotted as L_2 and H^1 error norms against number of unconstrained degrees of freedom, for both uniformly and adaptively refined meshes. 98

5.6 Concentration plots for Newtonian lid-driven cavity flow at Reynolds number 500, at nondimensionalized times $t = 5$ and $t = 25$ . . 99

6.1 Initial surfactant concentration, and a thin film flow solution at $t = 0.2$. 110

6.2 Film fluid depth solutions at $t = 0.1$ and $t = 0.2$. 110

7.1 Cross benchmark initial conditions and Cahn-Hilliard solutions at $t = 0.025, 0.050, 0.100, 0.200, 0.400$. Hermite cubic elements are each plotted as four bilinear squares. 125

7.2 Initial conditions with random nodal values and zero edge fluxes, and a Cahn-Hilliard solution at $t = 0.001$. 127

7.3 Cahn-Hilliard solutions at $t = 0.005, 0.01, 0.02, 0.05$. The interface widths are now effectively constant, and interface lengths progressively shorten. 128

7.4 Concentration isosurfaces for a 3D spinodal decomposition problem at $t = 0.033$. Separation between the three parallel isosurface manifolds is approximately equal to interface width. . . . 129

7.5 Integrated free energy over the square domain for the Galerkin approximation to a 2D spinodal decomposition problem. . . . 130

7.6 L_2 error compared to a reference solution, as a function of time, for a 2D spinodal decomposition approximated on a sequence of uniformly refined discretizations 132

7.7 Solutions at $t = 0.015$ on h_1 and h_4 grids. The h_2 solution is slightly perturbed from the reference h_4 solution, and the h_3 solution is visually indistinguishable from h_4. 132

7.8 Left: Refined Clough-Tocher mesh and solution in an adaptive transient Cahn-Hilliard problem, tracking a random spinodal decomposition at $t = 0.4$. Right: Refined Hermite mesh and Cahn-Hilliard solution, progressing from a cross-shaped initial condition to $t = 0.15$. 134

7.9 An experimental pattern of deuterated polystyrene and polybutadine on a monolayer substrate [79]. 135

7.10 Patterned spinodal decomposition simulations run with a 4% and 8% configurational free energy bias amplitudes, at $t = 0.5$. 136

7.11 Patterned spinodal decomposition simulations run with a 12% and 16% configurational free energy bias amplitudes, at $t = 0.5$. 137

7.12 Patterned spinodal decomposition simulations run with a 20% and 24% configurational free energy bias amplitudes, at $t = 0.5$. 137

7.13 Concentration isosurfaces in a patterned 3D spinodal decomposition simulation run with a 4% configurational free energy bias amplitude, at $t = 0.125$ and $t = 0.5$. 139

7.14 Concentration isosurfaces in a patterned 3D spinodal decomposition simulation run with a 16% configurational free energy bias amplitude, at $t = 0.125$ and $t = 0.5$. 140

7.15 Concentration isosurfaces in a patterned thin 3D spinodal decomposition simulation run with a 4% configurational free energy bias amplitude, at $t = 0.125$ and $t = 0.5$. 140

7.16 The rate of change of the concentration (in H^2 norm) plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple). 146

7.17 The free energy of the Cahn-Hilliard system plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple). 147

7.18 The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple). 148

7.19 The defect count of the Cahn-Hilliard system plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph. 149

7.20 Plots of the concentration in a patterned phase decomposition, with an initial perturbation magnitude of 0.001, at time $t \approx 0.0008$ on the left and $t \approx 0.1$ on the right. The early-stage solution appears to fit the imposed pattern, but full decomposition has not yet occurred. 149

7.21 The rate of change of the concentration (in a scaled H^2 norm) plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple). 151

7.22 The free energy per unit volume of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple). 152

7.23 The horizontal and vertical (in-film directions) correlation lengths of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple). 153

7.24 The thickness (through-film) direction correlation lengths of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple). 153

7.25 The defect count of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph. 155

7.26 The rate of change of the concentration (in a scaled H^2 norm) plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue). 157

7.27 The free energy per unit area of the Cahn-Hilliard system plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue). 157

7.28 The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue). 158

7.29 The defect count per unit length of the Cahn-Hilliard system plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue). 159

7.30 The rate of change of the concentration (in H^2 norm) plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple). 161

7.31 The free energy of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple). 161

7.32 The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple). 162

7.33 The defect count of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph. 162

7.34 The development of an instability in a patterned spinodal decomposition problem with $\epsilon_c = 0.04$. 164

7.35 The rate of change of the concentration (in H^2 norm) plotted against time, for average concentrations ranging from 0.25 (red) to 0.5 (purple). 164

7.36 The free energy of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.25 (red) to 0.5 (purple). 165

7.37 The rate of change of the concentration (in H^2 norm) plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple). 167

7.38 The free energy of the Cahn-Hilliard system plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple). 167

7.39 The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple). 169

7.40 The defect count of the Cahn-Hilliard system plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph. 170

7.41 The chemical free energy density of the Cahn-Hilliard system plotted against local concentration, for various temperatures. . 171

7.42 The free energy of the Cahn-Hilliard system plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple). 172

7.43 The rate of change of the concentration (in H^2 norm) plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple). . 173

7.44 The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple). 174

7.45 The defect count of the Cahn-Hilliard system plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph. 174
7.46 The time-step lengths at each point in simulation time for a single run of a patterned spinodal decomposition problem. . . 178
7.47 The minimum, mean, and maximum time-step length at each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study. 179
7.48 The number of inexact Newton steps taken at each point in simulation time for a single run of a patterned spinodal decomposition problem. 180
7.49 The minimum, mean, and maximum number of inexact Newton solver steps taken at time steps covering each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study. 181
7.50 The number of Krylov steps taken at each point in simulation time for a single run of a patterned spinodal decomposition problem. 182
7.51 The minimum, mean, and maximum number of total Krylov solver steps taken during time steps covering each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study. 183
7.52 The solver efficiency (expressed as a ratio of simulation time elapsing to clock time elapsing) at each point in simulation time for a single run of a patterned spinodal decomposition problem. 185
7.53 The minimum, mean, and maximum solver efficiency (expressed as a ratio of simulation time elapsing to clock time elapsing) during time steps covering each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study. 186

Chapter 1

Introduction

1.1 Research Motivation

Mathematical modeling and simulation of scientific and engineering applications has been been dramatically advancing in recent decades. Much of this advance has been driven by the evolution of miniaturization technology. The successes of microelectronics development has provided new opportunities to utilize an exponential growth in computational power for solving more complex problems. The limitations of computer technology have lead to new challenges in software design to enable the use of the power of modern parallel computers in complex codes for large-scale applications. The needs of miniturization technology themselves have led to new challenges in microscale and multiscale application problems. New application areas such as microfluidics and nanoscience are emerging, and in every application area new multiphysics and multiscale problems are continuously coming into the scope of numerical analysis capabilities. The research herein is a multidisciplinary investigation into the mathematical modeling of nonlinear fourth-order problems arising in flow and transport applications, ranging from non-Newtonian fluid flow and surfactant-driven thin film transport processes to phase separation processes and pattern self-assembly. To enable these studies this research includes new methodology, algorithms, and software contributions which also have applicability in the wider numerical analysis field.

Finite Element Methods are a time-tested way to efficiently simulate a wide range of continuum physical processes [31, 42, 72, 107]. They build on a solid theoretical foundation for providing accurate approximations to Boundary Value Problem and Initial-Boundary Value Problem solutions. Finite element meshes can be constructed to approximate domains of very general complex geometry, and can be locally refined to resolve particular solution features while adding minimal additional computational expense. The low-order C^0 Lagrange finite elements in common use are well-suited to many physical problems where a second-order partial differential equation is to be solved, and most of the finite element literature focuses on the application of the technology to second-order problems.

For the fourth-order problems of interest here, however, the numerical analysis literature is more restricted, often to less efficient formulations and invariably to less flexible methods. For instance, spectral methods and structured finite difference methods remain popular even for problems whose solutions exhibit thin features that would naturally be more efficiently resolved by an adaptive, unstructured discretization [7, 9, 29, 62]. Problems to be solved by such methods are often limited to simple geometries and cartesian grids.

There is a long history of successful application of finite element methods to fourth-order equations arising in plate bending models. These ideas have been extended to the streamfunction formulation of divergence-free viscous flow and other selected nonlinear problems. Of particular interest are the families of composite macroelements which can be used to cover irregular domain geometry with C^1 bases with suitable convergence properties for evaluating variational formulations on H^2 and similar function spaces. The current popularity of such methods may be limited, not by the versatility and performance with which they can be used to solve general problems, but by the complexity of the infrastructure which must be developed to enable a parallel adaptive finite element code to use these more complicated elements. Accordingly, in this research work we develop new methods for enabling the use of general classes of C^1 elements in a modern code. Extensions for treating hanging node constraints, projections, error indicators, and other requirements of adaptivity are discussed, as are algorithms for the efficient use of such general classes of elements on parallel unstructured meshes. The resulting code, along with contributions from other researchers, is made available under an open source license to the wider science and engineering community. By creating such infrastructure in a modular, object-oriented design, we meet the goal of creating tools that are extensible and flexible enough for use in a wide variety of applications as demonstrated in this work.

More specifically, this fourth-order finite element infrastructure is utilized here, in a series of physical problems of increasing complexity, from the simple biharmonic problem, up through incompressible flow problems in non-Newtonian shear-thinning fluids, and finally to surfactant-driven flow and to phase decomposition in thin films. The latter two problem classes involve, respectively, evolution of curved surfaces driven by surface tension modulated by temperature and surfactant gradients, and evolution of concentration phases in separating mixtures as described by phase field equations [5,57] with non-local energy terms in the underlying thermodynamics.

This final physical model, the Cahn-Hilliard diffuse-interface treatment of phase separation with an interfacial free energy term, is of current interest in material science processes from the microscale to the nanoscale [121,144], in solids and fluids [91,126], and for applications ranging from manufacturing [19, 79, 113, 119] to biology [84, 140], which are often naturally best described by such phase field equations. Even for materials naturally separated by sharp interfaces, diffuse interface methods using an artificially diffuse interface have been found to be a valuable way to simulate moving front problems [18, 21, 40, 97, 132, 138]. Diffuse interface models are of natural interest in any application where the moving interfaces undergo topological change, since with phase field methods such changes are naturally modeled by the equations and require no front-tracking special cases to simulate.

Problems with diffuse interfaces involve interesting physical phenomena at disparate space and time scales, making them a promising target for the adaptivity techniques in this work. In the studies here we first simulate generic Cahn-Hilliard processes of general interest, then phase decomposition processes with pattern self-assembly. The process of phase decomposition and the associated spinodal decomposition problem adds an additional layer of complexity to the problem, which now depends on both the deterministic Cahn-Hilliard evolution problem and on the aleatory uncertainty in the initial perturbation from a single-phase mixture.

1.2 Chapter Overview

Using a new macroelement infrastructure described in Chapter 2, we show how to automatically construct C^1 finite element spaces from low degree polynomials on arbitrary meshes. We experiment with Powell-Sabin-Heindl and Clough-Tocher elements, examining convergence rates and compatibility with adaptively refined meshes. The hanging node constraint matrix construction in Chapter 3 enables the easy extension of most C^1 finite element spaces to locally refined, non-conforming meshes, and the *a* posteriori error indicator derived in that chapter allows us to construct meshes which are adaptively refined into solution features of fourth-order boundary value problems. These results demonstrate a significant efficiency improvement from adaptive solutions to such problems, including improved convergence rates on problems with strong singularities. Reliable heuristics have also been developed for employing adaptive element refinement on transient problems, alone or in combination with adaptive time discretization.

As part of this dissertation, these techniques have now been incorporated into an object-oriented software library for performing parallel adaptive solutions to finite element formulations. Some of the design concepts, parallel data structures and algorithms, and solver operation in this work are described in Chapter 4. The general utility of the design decisions here explains their effectiveness at solving the diverse class of physics applications which follows.

Chapter 5 demonstrates the value of C^1 elements in incompressible fluid flow problems using Newtonian and shear-thinning viscosity models. By using the streamfunction formulation and solving the associated fourth-order problem here, strongly rather than weakly divergence-free solutions are obtained, and solvers deal with positive definite and positive semidefinite systems rather than the saddle point problems associated with more traditional mixed velocity-pressure methods.

Thin film flow and transport problems are becoming increasingly important in microelectronic, micromechanical, and microfluidic design. In Chapter 6 and Chapter 7, we examine two classes of physics which are of importance in such industrial thin fluid layer problems. In Chapter 7 we develop an appropriate formulation and simulate the instabilities arising in phase decomposition of fluid mixtures, using the Cahn-Hilliard diffuse interface model. The fourth-order interfacial energy term in the Cahn-Hilliard equation is a natural candidate for conforming approximation on C^1 elements, and the narrow interfaces between material phases prove to be ideal targets for local mesh refinement. We finally apply these techniques and algorithms to the problem of patterned phase separation, exploring the physical parameter space while using Monte Carlo sampling to take into account the random initial perturbations inherent in the phase decomposition problem.

1.3 Contributions

The primary original contributions of this work are:

1. The extension of C^1 conforming macroelement types to non-conforming finite element grids, via techniques that apply to other general element classes as well.

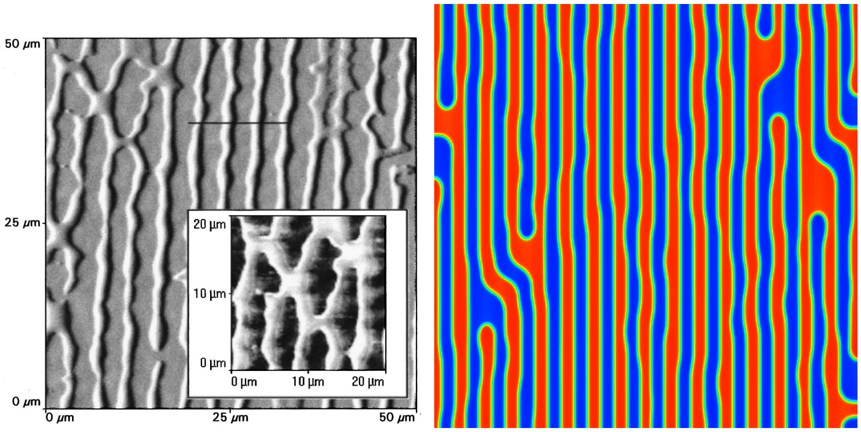

Figure 1.1: Left: A pattern with defects from the experimental literature, showing phase separated deuterated polystyrene and polybutadine on a monolayer substrate [79]. Right: A Cahn-Hilliard simulation from the present study with a weak spatially-dependent surface affinity term, showing similar qualitative behavior.

2. Derivation of an upper bound error indicator for the biharmonic problem with special consideration for composite macroelements, and results using simplified derived versions of this indicator on that and other fourth-order boundary value problems.

3. Algorithms for automatic adaptive refinement and coarsening on transient problems.

4. Implementation of these and other data structures and algorithms in a modern, open source library code which is in public use.

5. Adaptive streamfunction solutions to divergence-free flow problems with non-Newtonian fluids in benchmark problems exhibiting strong singularities.

6. Demonstration of a new fully coupled, non-mixed weak formulation for depth-averaged thermally driven thin film flow and surfactant transport.
7. Studies of Cahn-Hilliard phase decomposition in 2D and 3D, including parametric and Monte Carlo studies of pattern self-assembly with an added configurational free energy bias.

Parts of this work have been published in *Engineering With Computers* [86] and the *International Journal for Numerical Methods in Engineering* [116, 118]. Further papers are in preparation [115, 117].

Other new contributions in this work include:

1. A new benchmark problem based on incompressible viscous flow, with an analytic solution and no manufactured forcing function, for testing adaptive refinement strategies for fourth-order problems.
2. Algorithms and distributed-memory data structures for enabling the efficient use of parallel unstructured adaptive finite element grids on massively parallel clusters.
3. Strategies for the reliable application of transient and nonlinear solver techniques to a variety of discretized Initial-Boundary Value Problems.
4. Development and implementation of object-oriented software design principles for abstraction and modularization of transient multiphysics boundary value problem codes.
5. Proposed formulations for finite element Cahn-Hilliard discretizations with provably non-increasing total free energy

6. Identification of physical parameters which strongly impact the reliability of pattern formation in thin film phase decomposition.

Chapter 2

Automatic Macroelement Generation and Use

2.1 Introduction

We consider macroelement classes designed to support C^1 continuous finite element bases with few degrees of freedom, and thus to efficiently solve fourth-order problems on fine meshes. Such C^1 bases are of interest in plate bending, in the streamfunction formulation of 2D incompressible viscous flow, in thin film problems, and in the construction of divergence-free bases for more general incompressible flow problems [30, 61].

We begin with the fourth-order case, and with the familiar biharmonic equation as the model problem. The macroelement bases of interest here are composed from standard polynomial bases on simplicial subelements. The basic macroelement type is described first, and an algorithm is given for constructing C^1 bases (easily extensible to C^r) on a piecewise polynomial master element in $\mathbb{R}^d$. Representative macroelement basis functions are graphed, and interpolation and approximation error convergence rates are discussed and tested on uniformly refined grids. Next, the use of C^1 macroelements is extended to hierarchical adaptive mesh refinement (AMR). The problem of compatibility between macroelement subdivision and adaptive refinement subdivision is investigated and a simple algorithm for generating degree-of-freedom constraints at “hanging nodes” of refined elements is developed. In later chapters, a general *a posteriori* error estimator for the biharmonic problem is derived, and a simplified error indicator is tested numerically on a simple model problem and on wide classes of application problems.

This error estimator is applicable to any C^1 element with adequate interpolation accuracy for functions in Hilbert space $H^2(\Omega)$ on a bounded domain Ω. The element basis construction procedures are applicable to any elements whose macroelement splittings are preserved under affine transformation, and the procedures are most conveniently and efficiently applied to elements whose local function spaces are preserved under affine transformation.

A macroelement splitting may be termed "affine-transformable" if for any two such elements K and K' related by an affine transformation $K' = T(K)$, every subelement $S \subset K$ corresponds to a subelement $T(S) \subset K'$. We call a finite element function space affine-transformable if any basis function $\phi_i^{K'}$ on one element can be expressed as a weighted sum of transformed basis functions from the other element, $\phi_i^{K'} = \sum c_{ij} T(\phi_j^K)$.

The fully affine-transformable C^1 elements include the quadratic Powell-Sabin-Heindl 12-split triangle [103], the cubic Hsieh-Clough-Tocher 3-split triangle [43], the quartic Clough-Tocher-Percell 3-split triangle [100], the quintic Argyris triangle, and the quintic Alfeld-Awanou-Lai 4-split tetrahedron [6]. C^1 elements with affine-transformable splittings but without affine-transformable function spaces include the quadratic reduced Clough-Tocher triangle [42] and the cubic reduced Alfeld tetrahedron [3], and C^1 elements with mesh-dependent (and thus in general not affine-transformable) splittings include the quadratic Powell-Sabin 6-split triangle, the Lai quadrilateral [90], the

Worsey-Farin 12-split tetrahedron [141], and the Worsey-Piper 24-split tetrahedron [142].

Our subsequent numerical experiments specifically use the cubic Clough-Tocher and the Powell-Sabin triangles, which are described in detail below.

2.2 Macroelements

Enforcing C^1 continuity between finite elements requires additional degrees of freedom corresponding to normal derivatives on element boundaries. For high-degree polynomial elements, these can replace internal degrees of freedom, but low-degree elements on arbitrary meshes do not have enough internal degrees to permit this. For example, the Bogner-Fox-Schmidt rectangle can in general only maintain C^1 continuity on C^1 mappings of rectilinear grids [101], and elements utilizing polynomials on regular simplicial grids can only maintain full interpolation accuracy for $k \geq 5$ (e.g. the Argyris triangle) in 2D or for $k \geq 9$ in 3D [77, 130].

A C^1 continuous function space on an arbitrary unstructured grid with low-degree polynomials may be constructed by using macroelements composed of subelements. By making the basis functions piecewise polynomial on these subelements rather than polynomial on the macroelement, one can add more degrees of freedom internal to the macroelement. Some of these additional degrees of freedom are then constrained to satisfy intra-element continuity restrictions, and the remainder allow us to create enough vertex and side degrees of freedom to exactly represent both function values and gradients across element boundaries. Our current results use the Hsieh-Clough-Tocher

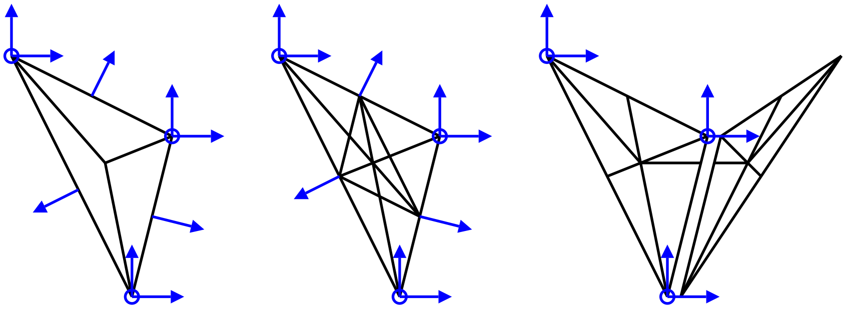

Figure 2.1: A Hsieh-Clough-Tocher 3-split, a Powell-Sabin-Heindl 12-split, and two adjacent Powell-Sabin 6-split triangles, with value degrees of freedom labeled by circles and derivative degrees of freedom labeled by arrows. Note the mesh-dependent splitting required for the 6-split triangle type.

(HCT) 3-split, the Powell-Sabin-Heindl (PSH) 12-split, and the PSH 6-split triangles, whose subelement splittings and degrees of freedom are shown in Figure 2.1.

Each subtriangle in the Powell-Sabin-Heindl (PSH) 6-split triangle has one vertex at a macroelement vertex, one vertex along a macroelement edge (directly between the interior vertices of the two neighboring macroelements), and one vertex in the macroelement interior (usually at the macroelement centroid). Basis functions on this PSH element are quadratic on each subtriangle, and nine degrees of freedom (corresponding to function values and x and y derivatives at each vertex) uniquely define basis functions which include all quadratics but which now have first derivatives continuous with neighboring elements.

Because satisfying continuity with the PSH 6-split triangle requires a subelement splitting which varies depending on the geometry of neighboring subelements, finite element codes using the PSH 12-split triangle are more convenient. The 12-split triangle no longer restricts the location of the edge vertex (which is usually placed at an edge midpoint), but it instead adds internal edges between each pair of midpoints, as well as three new degrees of freedom corresponding to function normal derivatives at these midpoints. The 12-split triangle edge is then able to represent any piecewise-quadratic function and piecewise-linear normal derivative.

The same 12 degrees of freedom can alternatively be used to define a unique piecewise cubic function on the HCT 3-split triangle. The fixed values and tangential derivatives at each node define a unique cubic function along each edge, and the fixed normal derivatives at each node and midpoint define a unique quadratic normal derivative. Hence, the 3-split triangle gives a C^1-continuous, piecewise cubic function matching any cubic edge data with quadratic edge derivatives.

2.3 Master Basis Function Derivation

The construction of early macroelements such as the HCT triangle was done manually, requiring clever proofs to show unisolvence and tedious algebraic manipulations to find basis function coefficients. Modern linear algebra software can circumvent this difficulty, making it possible to derive and prove the unisolvency of more complicated macroelement basis functions algorithmically [85].

The HCT and PSH macroelements considered here are constructed from subtriangles with between 30 and 72 total unconstrained degrees of freedom. In general, enforcement of continuity restrictions between neighboring subelements involves the application of many constraint equations with subtle linear interdependencies. Constructing quadrilateral, three dimensional, or higher degree macroelements further increases the problem complexity. For elements such as the HCT and PSH 12-split triangles whose splittings are always preserved by affine transformations, the basis functions on a fixed "master" element can be obtained automatically with any linear algebra package.

Given a subelement splitting and local degree-of-freedom functionals $\hat{\sigma}_j$ for a C^1 macroelement in $\mathbb{R}^d$, the steps for constructing the master basis are:

1. Choose and index a convenient (e.g. monomial) set of explicitly defined basis functions $\phi_j(\vec{\xi})$ which are each supported and continuous on exactly one subelement of the master element $\hat{K}$. The goal is to find basis functions $\theta_i(\vec{\xi}) \equiv \sum_j r_{ij}\phi_j(\vec{\xi})$ which are C^1 continuous across $\hat{K}$ and which uniquely solve the degree-of-freedom equations $\hat{\sigma}_j(\theta_i) = \delta_{ij}$.
2. On each internal side of the subelements of $\hat{K}$, identify the regularly distributed Lagrange interpolation points $\vec{\xi}_n$. For functions of polynomal degree p on a simplicial side, there will be $\frac{(p+d-1)!}{p!(d-1)!}$ such points.
3. At each point $\vec{\xi}_n$, express one function equality constraint and d gradient component equality constraints, each in the form $\sum_j C_{kj} r_{ij} = 0$, assigning each constraint a unique number k. For equality constraints, these coefficients are of the form $C_{kj} = s\phi_j(\vec{\xi}_{n_k})$, where s is arbitrarily chosen to equal 1 for degrees of freedom j supported on one subelement sharing the side and -1 for degrees of freedom supported on the other subelement. For gradient constraints, the coefficients take the form $C_{ik} = s\frac{\partial \phi_j}{\partial \xi_l}(\vec{\xi}_{n_k})$.

4. Put this constraint matrix C in row-reduced form. Truncate any all-zero rows (and any nearly-zero rows if using floating point arithmetic).

5. Append boundary degree-of-freedom functionals $\hat{\sigma}_k$ as additional rows, which will take the form $C_{ik} = \hat{\sigma}_k(\phi_i)$. For functionals which are evaluated on points shared by the boundary of multiple subelements, all basis functions except those on a single (arbitrary) subelement should be evaluated (and have their derivatives evaluated) as zero at such points. Verify the linear independence of each new row, e.g. by testing the rank of C.

6. For the macroelements discussed in this article, C will now be a square matrix. For more general macroelements, if C is not square at this step then make it so by adding interior degree-of-freedom functions, again checking each for linear independence.

7. Invert C, and multiply by unit vector $\hat{e}_i$ to get the basis coefficients r_{ij} corresponding to the degree of freedom on row i.

This calculation guarantees the construction of basis functions $\theta_i(\vec{\xi}) \equiv \sum_j r_{ij}\phi_j(\vec{\xi})$ which are C^1 continuous within the master macroelement and which are a unique solution to the degree-of-freedom equations $\hat{\sigma}_j(\theta_i) = \delta_{ij}$. If the degrees of freedom on the element boundary are sufficient to uniquely define any admissible function values and normal derivatives on an element boundary, then the resulting element will form a C^1 continuous function on an arbitrary mesh.

2.4 Global Degree of Freedom Calculation

Because global degrees of freedom σ_j on a C^1 continuous finite element space depend on first derivative components as well as function values, they can not be created by composing a master element "shape" function θ_i with the inverse of the map $T : \hat{K} \to K$ between the master and physical element. Composing master element shape functions with the inverted mapping function does give functions $\theta_i \circ T^{-1}$ spanning the global function space restricted to the element, but none necessarily correspond directly to any global degree of freedom.

As a simple example, consider how nodal gradient degrees of freedom on a two dimensional element are affected by a non-identity map T at the node. In the global function space, a natural choice of degrees of freedom are the x and y derivatives, the gradient in the direction of vectors $\vec{e}_x$ and $\vec{e}_y$. The local degrees of freedom, however, correspond to multiples of the gradient in the direction of $T\vec{e}_\xi$ and $T\vec{e}_\eta$, which in general will not correspond to the global vectors as illustrated in Figure 2.2.

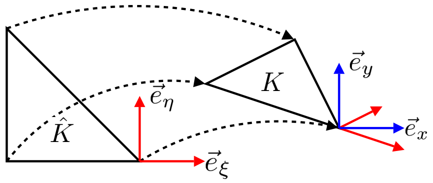

Figure 2.2: Derivatives in cartesian directions on the master element may correspond to derivatives in skewed directions on a physical element.

The shape function Θ_i which does correspond to each global degree of freedom will be a linear combination of multiple transformed master shape functions, $\Theta_i \equiv \sum_j A_{ij}\theta_j \circ T^{-1}$. Again, the coefficients of this transformation can be obtained algebraically. Evaluating all transformed local functions at all degrees of freedom, $\sigma_i(\Theta_k) = \sum_i A_{kj}\sigma_i(T^{-1}(\theta_j)) = \delta_{jk}$, gives a small matrix $S_{ji} \equiv \sigma_i(T^{-1}(\theta_j))$ whose inverse can be used to find the coefficients: $AS = I \Rightarrow A = S^{-1}$.

Both S and S^{-1} are sparse. For both the HCT and PSH 12-split cases, only 33 of 144 matrix entries are non-zero. Because this calculation must be repeated on each physical element, efficient calculations should take this sparsity structure into account. Although the present results only use meshes of affine-transformed elements, the only requirement for the above construction algorithm is that T is invertible.

2.5 A Priori Error Estimates

Interpolation Error PSH and HCT elements possess stable local bases which can interpolate quadratic and cubic functions (polynomial degree $k \equiv 2, 3$) respectively. Thus, on a polygonal domain Ω with Lipschitz boundary, a regular family of meshes with maximum element diameter h can interpolate any sufficiently smooth function w with an accuracy restricted by k and by the smoothness of w in spaces of order up to $n \leq k+1$. That is, the interpolant

$$I_h(f) \equiv \sum_{i=1}^{N} \sigma_i(f)\phi_i \tag{2.5.1}$$

satisfies

$$||w - I_h w||_{H^m(\Omega)} \leq Ch^{\mu}\,|w|_{H^n(\Omega)} \tag{2.5.2}$$

for $\mu \equiv \min\,(k+1-m, n-m)$.

Galerkin Approximation Error Consider the model fourth-order problem $\Delta^2 u = f$, where Δ^2 denotes the biharmonic operator. On the domain Ω, the weak form of the biharmonic operator is $B(u, v) \equiv (\Delta u, \Delta v)_\Omega$, where Δ denotes the Laplacian operator and $(\cdot, \cdot)_\Omega$ is the standard inner product on Lebesgue space $L_2(\Omega)$. If the weak problem is restricted by essential boundary conditions to a subset $H_0 \subset H^2(\Omega)$ on which B is coercive, then B will satisfy an inf-sup condition on both H_0 and the H_0 conforming function subspaces H_0^h. Galerkin orthogonality (the Céa lemma) then gives the expected (linear for PSH, quadratic for HCT) rate of convergence in the H^2 norm, for problems with sufficiently smooth solutions (contained in $H^n(\Omega)$, with $n > m$). See e.g. [32]. More specifically, let

$$\alpha \equiv \inf_{v \in H_0} \sup_{w \in H_0} \frac{B(v, w)}{||v||_{H^m(\Omega)} \, ||w||_{H^m(\Omega)}} \tag{2.5.3}$$

and let M be the bound on B for all admissible v and w satisfying

$$|B(v, w)| \leq M \, ||v||_{H^m(\Omega)} \, ||w||_{H^m(\Omega)} \tag{2.5.4}$$

Then

$$\begin{aligned} ||u - u_h||_{H^m(\Omega)} &\leq \frac{M}{\alpha} \, ||u - I_h u||_{H^m(\Omega)} && (2.5.5) \\ &\leq \frac{M}{\alpha} C h^\mu \, |u|_{H^n(\Omega)} && (2.5.6) \end{aligned}$$

As anticipated, the HCT elements yield an additional power of h in the H^2 convergence rate with no more degrees of freedom than the 12-split PSH. Less obvious is that the difference increases in the L_2 (i.e. H^0) norm. Application of the Aubin-Nitsche method gives an H^r error estimate for $r < m$.

Define $\nu \equiv \min(k+1-m, m-r)$, the interpolation error convergence rate in H^m for the solution to an adjoint problem with data in H^{-r}. Also define μ as before, the approximation error convergence rate for a solution $u \in H^n$ to the forward problem. Then, as proven in Sec. 3.3 of [32], if u_h is the Galerkin approximation to a bilinear elliptic problem on $H^m(\Omega)$, its error is of order $\eta \equiv \nu + \mu = \min(2(k+1-m), k+1-r, n-r)$:

$$||u - u_h||_{H^r(\Omega)} \leq Ch^\eta ||u||_{H^n(\Omega)} \tag{2.5.7}$$

For $k = 3$, or for $k = 2$ and $r \geq 1$, this is the usual result: measuring error in a lower order norm adds a power of h to the convergence rate. Note, however, that in the case of $m = 2$ (fourth-order problems), $k = 2$ (quadratic elements) and $r = 0$ (the L_2 norm), the $\eta \leq 2(k+1-m)$ term dominates, and the error bound is only quadratic, not cubic. Numerical tests confirm the suboptimal behavior in this norm.

This result appears to make PSH elements an inferior choice for plate-bending problems, depth-averaged thin layer problems, or any other applications where the scalar solution to a fourth-order equation is of direct interest. However, the lower L_2 convergence rate merely implies poor streamfunction error in streamfunction and divergence-free flow formulations. Velocities in these problems are usually of more direct concern, and velocity L_2 error is only limited by streamfunction H^1 error; in the H^1 norm PSH elements achieve the full $\mathcal{O}(h^2)$ convergence, so optimal results are achieved here for velocity approximations.

As a demonstration that these error bounds are strict, we compute the Galerkin approximation to $u \equiv (x-x^2)^2(y-y^2)^2$ on the unit square by solving the biharmonic problem $\Delta^2 u = f$, with a manufactured forcing function f corresponding to this u. As indicated by the slopes of the log-log error plots in Figure 2.3, both 6-split and 12-split elements show quadratic error convergence (with nearly the same constant, in fact) in L_2 and H^1 norms, whereas the 3-split element error convergence is asymptotically cubic in H^1 and quartic in L_2.

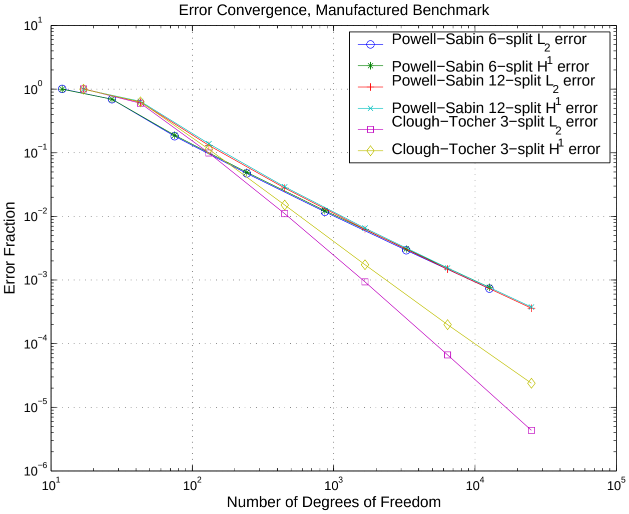

Figure 2.3: Galerkin approximation error in L_2 and H^1 norms, scaled by the solution norm and evaluated on uniform meshes for the manufactured benchmark biharmonic problem.

The direct Galerkin solution of fourth-order problems can compare favorably in degree-of-freedom cost to popular alternatives, due to the consolidation of degrees of freedom that continuity enforcement entails. An HCT mesh provides cubic representations with up to six degrees of freedom per node, for example, whereas the classical mixed method [95] with paired cubic finite elements would require up to eighteen, and a continuous/discontinuous Galerkin method [58] on C^0 cubic finite elements would require up to nine.

2.6 Adaptive Refinement on Macroelements

In each of the preceding error bounds, the convergence rates are only optimal for sufficiently smooth solutions, and are reduced for problems with singularities. Even in smooth problems with sharp layers it may be inefficient or impractical to reach the asymptotic regime with a uniformly refined mesh. To produce an accurate approximation with fewer degrees of freedom, we seek to adapt the mesh to the particular problem being solved, grading the mesh locally into solution features where higher resolution is needed.

From a coarse initial mesh, we can obtain an isotropically adaptively refined mesh by repeatedly subdividing elements with the greatest contribution to approximation error. In all but one-dimensional problems, this produces a non-conforming mesh, with "hanging nodes" on the sides of coarse elements with more refined neighbors. To produce a C^1 conforming solution on such a "non-conforming" mesh, the degrees of freedom of these refined elements must be constrained in terms of the degrees of freedom of their coarser neighbors.

AMR schemes with hanging node constraints are commonly used with Lagrangian finite elements, but more exotic element types add the complication that nodes can share physical location without sharing degrees of freedom. For example, at a hanging node in a HCT or PSH 12-split mesh, the midedge normal derivative degree of freedom on the coarse element will be equal to a linear combination of the nodal gradient component degrees of freedom on the refined neighbor elements, but in general the normal derivative will not be equal to either component alone.

If hanging node refinement is to reduce approximation error, these con-

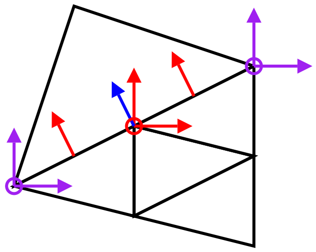

Figure 2.4: A typical hanging node between Hsieh-Clough-Tocher or Powell-Sabin-Heindl 12-split triangles. The coarse triangle nodal degrees of freedom (purple) match those of the refined neighbors, but the refined hanging node degrees of freedom (red) do not match the coarse normal derivative degree of freedom (blue) at the node, and the refined normal derivative degrees of freedom have no corresponding coarse degrees of freedom at all.

tinuity constraints must constrain the degrees of freedom on the fine elements at a hanging node, never the coarse element. When using macroelements, this can limit an AMR scheme. Figure 2.5 illustrates this point with the 12-split PSH element. With these elements, approximations are piecewise quadratic on each edge and have a potentially discontinuous tangential second derivative at the subelement node on the macroelement edge. If this subelement node is not made to correspond with a subelement node or hanging node of the neighboring refined elements, then the edge value would be forced to be the same quadratic across the subelement node, the coarse element degrees of freedom would no longer be independent, and the locality of those basis functions would be destroyed. For the 12-split elements, both uniform subelement division and isotropic refinement use edge midpoints, and so a hanging node mesh is natural. For the 6-split elements, isotropic refinement requires first moving the subelement node on the coarse macroelement to the macroelement edge midpoint, then moving the subelement nodes on the refined elements to make

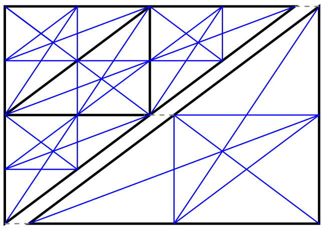

Figure 2.5: An isotropic adaptive refinement of a Powell-Sabin-Heindl 12-split triangle. The subelement node at the edge midpoint of the coarse triangle must also be a node of the neighboring refined triangles.

those subelements sides quadratic with linear gradients rather than piecewise quadratic with piecewise linear gradients.

The choice of macroelement splitting can make adaptive refinement schemes much easier or harder. With HCT elements, each edge borders a single subelement, and any AMR scheme will be compatible with this splitting. The Worsey-Farin and Worsey-Piper tetrahedra, on the other hand, each require a face splitting which is not compatible with isotropic quadrisection of a triangle, and which is therefore only well-suited to conforming meshes.

The question of p refinement on macroelement meshes raises similar concerns about function space compatibility. Macroelements which require splitting of element sides, such as the PSH triangles and the Worsey-Farin and Worsey-Piper tetrahedra, may not be compatible with elements with unsplit element sides, such as the HCT triangles and the quintic Alfeld-Awanou-Lai 4-split tetrahedron. On the other hand, macroelements with unsplit sides can be made compatible with higher degree elements all the way up to unsplit C^1 finite elements. It may be possible to make C^1 triangles suitable for adaptive p refinement from quadratic to quintic and above, for example, by finding compatible bases for the function spaces defined by the restricted HCT triangle, HCT triangle, Clough-Tocher-Percell triangle, and Argyris triangle.

Chapter 3

Adaptive Mesh Refinement/Coarsening for Fourth-order Problems

C^1 finite elements, and macroelements in particular, raise new difficulties when used in conjunction with adaptive mesh refinement. Of particular interest is the treatment of hanging nodes on hierarchically refined meshes. In this chapter we discuss methods for building and working with conforming function spaces on such non-conforming meshes.

The important developments which make this possible are:

1. The ability to construct conforming function spaces on non-conforming adaptively refined meshes, as explained in Section 3.1.
2. A projection operator to accurately and efficiently transfer the solution from each time step or adaptive step onto the new mesh for the next step, such as in Section 3.2.
3. An error indicator which can reliably guide mesh refinement for fourth-order problems, as in Section 3.3
4. An algorithm to use the information provided by those error estimates to generate the new mesh at each time step or adaptive step, such as in Section 3.4 for steady problems and Section 3.5 for time-dependent problems.

In Section 3.6, we test some of these techniques, using a fourth-order boundary value problem designed to produce an analytically known solution with a point singularity.

3.1 Adaptive Meshes and Continuity

For convenience of notation, let us call the more refined and less refined macroelements sharing a hanging node "fine" and "coarse" respectively. Once it has been ensured that the coarse element function space restricted to a side is contained in the function space on the fine elements sharing the side, it is necessary to constrain the fine element degrees of freedom to enforce C^1 continuity across the side. Degrees of freedom u_i^F on a side of the fine element K_F with solution u^F must be expressed in terms of degrees of freedom u_i^C on the overlapping side of the coarse element K_C. The goal is to enforce that on this side $\gamma \equiv \bar{K_F} \cap \bar{K_C}$ all derivatives $D^\alpha u$ up to the required continuity are equal. That is, $\forall\ \vec{x} \in \gamma$, $\forall\ |\alpha| \leq r$,

$$D^\alpha u^F(\vec{x}) = D^\alpha u^C(\vec{x}) \tag{3.1.1}$$

Expanding in their respective bases,

$$D^\alpha \sum_i u_i^F \phi_i^F(\vec{x}) = D^\alpha \sum_j u_j^C \phi_j^C(\vec{x}) \tag{3.1.2}$$

Because these spaces have finite dimension, the constraint equations can be solved numerically. For codes which have direct access to the finite element degree-of-freedom equations, the most efficient solution is to directly use the fine element degrees of freedom on γ:

$$\sigma_k^F(u^F) = \sigma_k^F(u^C) \tag{3.1.3}$$

Expanding in their respective bases and using the linearity of degree-of-freedom functionals:

$$\sum_i u_i^F \sigma_k^F(\phi_i^F) = \sum_j u_j^C \sigma_k^F(\phi_j^C) \tag{3.1.4}$$

If the degree-of-freedom equations are used, then the desired degree of continuity will be obtained automatically; otherwise the number of continuous derivatives r must be taken into consideration to produce a non-singular system with a C^r continuous solution. This may be done using a suitable set of collocation points $\vec{x}_k \in \gamma$:

$$\sum_{|\alpha|\leq r} D^\alpha u^F(\vec{x}_k) = \sum_{|\alpha|\leq r} D^\alpha u^C(\vec{x}_k) \tag{3.1.5}$$

Expanding and using the linearity of derivatives:

$$\sum_i u_i^F \sum_{|\alpha|\leq r} D^\alpha \phi_i^F(\vec{x}_k) = \sum_j u_j^C \sum_{|\alpha|\leq r} D^\alpha \phi_j^C(\vec{x}_k) \tag{3.1.6}$$

As a final alternative, constraint matrices can be constructed from L_2 projections on γ:

$$\sum_{|\alpha|\leq r} (D^\alpha u^F, D^\alpha \phi_k^F)_\gamma = \sum_{|\alpha|\leq r} (D^\alpha u^C, D^\alpha \phi_k^F)_\gamma \tag{3.1.7}$$

Expanding and using the linearity of derivatives:

$$\sum_i u_i^F \sum_{|\alpha|\leq r} (D^\alpha \phi_i^F, D^\alpha \phi_k^F)_\gamma = \sum_j u_j^C \sum_{|\alpha|\leq r} (D^\alpha \phi_j^C, D^\alpha \phi_k^F)_\gamma \tag{3.1.8}$$

Each of these alternatives gives a linear system with different coefficients but the same general form:

$$A_{ki} u_i^F = B_{kj} u_j^C \tag{3.1.9}$$

If the degree-of-freedom equations are used, A will be an identity matrix and B will be the desired constraint matrix. In any case, the unique constraint matrix will be obtained by solving: (3.1.9):

$$u_i^F = A_{ki}^{-1} B_{kj} u_j^C \tag{3.1.10}$$

to calculate a constraint matrix, which can be used to pre- and post-multiply element stiffness matrices as in [27].

These methods can be extended to adaptive p refinement, where two or more elements sharing a face may need to be constrained on that face to the lowest polynomial degree among them. Although in this case the definition of “refined” and “coarse” elements must now include higher and lower polynomial degree, the techniques required should be little different.

Currently, the constraint calculations in our finite element software library `libMesh` are implemented with the degree-of-freedom functionals for Lagrange elements, and with L_2 projections for all other elements including the C^1 elements used in this research. Constraints for h refinement are calculated between a “child” element and its own “parent” to minimize inter-node communication in parallel problems. This process can be done recursively on meshes where elements are allowed to have neighbors which are more than one level more refined. Constraint equations for p refinement are only supported by `libMesh` for hierarchic finite element bases; the resulting constraint equations are all of the form $u_i^F = 0$ and their construction is relatively easy to implement in software.

3.2 Projection Operators

For small steady-state problems, no projection operator is necessary. Forcing functions, material property fields, and the like can be incorporated into finite element solutions via approximate numerical integration, without first projecting those functions into a finite element function space.

For large steady-state problems and time-dependent problems, however, it can become important to be able to efficiently and accurately transfer solution data from one mesh to another. When solving steady-state problems on successively more refined meshes, the solution on a fine mesh can be obtained more rapidly by using the solution on a previous coarser mesh as an initial iterate, or by using the bilinear operator on a previous coarse mesh as a multigrid preconditioner. When solving time-dependent problems via adaptive refinement and coarsening, the time integration scheme may require that solutions at previous time steps be projected onto a differently-refined mesh at the current timestep.

Implementing the projections required by adaptive mesh refinement and coarsening, requires a projection operator well suited to hierarchical mesh refinement methods. Such an operator should be accurate, obviously, but other operator characteristics are equally important. The projection operator should be computationally efficient - to be useful in multigrid schemes, in fact, the operator should have a computational cost which scales linearly with the number of elements in the mesh. The projection operator should be uniquely defined and parallelizable - when using the parallel computation features in `libMesh` to operate on neighboring finite elements simultaneously, for instance, the projections computed should agree at the neighbors' interface. Finally, the projection operator should be as independent of finite element type as possible. There is a wide variety of finite element families useful for constructing C^1 function spaces, and it would be ideal to use the same algorithm for all of them.

As with constraint matrix construction, it is possible to use Hilbert space projections to create element-independent code. Using an L_2 or H^1 projection over each element interior is efficient, runs in parallel without interprocessor communication, and gives an exact solution in the case of refinement using nested finite element spaces. For coarsening, however, an element-wise H^s projection would not be uniquely defined, as the projections from neighboring cells could produce different function values along their shared side. A more complicated but similarly efficient algorithm restores uniqueness by acting on these shared degrees of freedom first, as follows:

Start by interpolating degrees of freedom on coarse element vertices. Holding these vertex values fixed, do projections along each coarse element edge. Because these projections involve only data from the original refined elements on that edge and not data from element interiors, they are uniquely defined. In 3D, next project element faces while holding vertex and edge data fixed. Finally, project element interior degrees of freedom while holding element boundary data fixed. Although this series of projections is more complicated than a single per-element projection, the number of degrees of freedom to be solved for at each stage is much smaller, and so the dense local matrix inversions required are faster.

3.3 A Posteriori Error Estimation

A variety of error indicators of different degrees of complexity and computational cost can be devised for adaptive simulations. These include interpolation-based indicators, residual indicators, flux jump indicators, patch recovery indicators and adjoint-based dual indicators [2,28,36,59,69,146,147]. The simple flux jump indicator based on the Laplace operator has been used successfully for a variety of second order problems, so the natural extension here is to use an analogous indicator based on the biharmonic operator.

For elliptic second-order problems, it is common to guide AMR by using a flux error estimator [81], integrating weighted normal derivative jumps around the boundary of each element. In a C^1 finite element space, there are no first derivative jumps, but from the classic biharmonic PDE one can derive analogous *a posteriori* error estimators. The finite element literature on fourth-order problems includes estimators derived for mixed formulations [39], and for special bases such as tensor products on rectilinear meshes [1]; however the estimator below will be valid for arbitrary meshes of affine-transformable C^1 elements.

Consider the problem of finding u such that $\Delta^2 u = f$ on the polygonal domain Ω, with Dirichlet (essential) boundary conditions $u = h_1$ on $\Gamma_{D1} \subset \partial\Omega$ and $\partial_{\vec{n}} u = h_2$ on $\Gamma_{D2} \subset \partial\Omega$, and with Neumann (natural) boundary conditions $\partial_{\vec{n}} \Delta u = g_1$ on $\Gamma_{N1} \equiv \partial\Omega - \Gamma_{D1}$ amd $\Delta u = g_2$ on $\Gamma_{N2} \equiv \partial\Omega - \Gamma_{D2}$.

Choose a fixed $u_B \in H^2(\Omega)$ such that $u_B|_{\Gamma_{D1}} = h_1$ and $\partial_{\vec{n}} u_B|_{\Gamma_{D2}} = h_2$. Define a function space H_0 with zero Dirichlet boundary conditions:

$$H_0 \equiv \left\{ v \in H^2(\Omega) : v|_{\Gamma_{D1}} = 0, \partial_{\vec{n}} v|_{\Gamma_{D2}} = 0 \right\} \tag{3.3.1}$$

Then, in the weak formulation, seek a solution u such that $u - u_B \in H_0$. Define the bilinear functional $B : H^2(\Omega) \times H^2(\Omega) \to \mathbb{R}$:

$$B(u, v) \equiv (\Delta u, \Delta v)_\Omega \tag{3.3.2}$$

and the linear functional $L : H^2(\Omega) \to \mathbb{R}$:

$$L(v) \equiv (f, v)_\Omega - (g_1, v)_{\Gamma_{N1}} + (g_2, \partial_{\vec{n}} v)_{\Gamma_{N2}} \tag{3.3.3}$$

Then, multiplying the residual by a test function v and integrating by parts twice lets us express the problem in the weak formulation:

$$B(u, v) = L(v) \quad \forall v \in H_0 \tag{3.3.4}$$

For convenience, assume that the finite element function space H^h can interpolate the Dirichlet boundary conditions exactly, and that $u_B \in H^h$. Define a boundary constrained finite element space H_0^h:

$$H_0^h \equiv \left\{ v_h \in H^h : v_h|_{\Gamma_{N1}} = 0, \partial_{\vec{n}} v_h|_{\Gamma_{N2}} = 0 \right\} \tag{3.3.5}$$

Then, the Galerkin formulation is to find an approximate solution $u_h \in H^h$ such that $u_h - u_B \in H_0$ and

$$B(u_h, v_h) = L(v_h) \quad \forall v_h \in H_0^h \tag{3.3.6}$$

Defining the error $e \equiv u - u_h$, subtract $B(u_h, v)$ from both sides of the original weak equation (3.3.4) to obtain

$$B(e, v) = -B(u_h, v) + L(v) \quad \forall v \in H_0 \tag{3.3.7}$$

Divide Ω into subdomains, choosing the coarsest partition of each element such that each subdomain $S \in K$ satisfies $H^h|_S \in H^4(S)$. For a macroelement-based finite element space, solutions on whole finite elements may not have H^4 regularity, in which case the subdomain partition will be the set of subelements. Express $B(e, v)$ as a sum of separate integrals over each of these subelements

$$B(e,v) = \sum_S \left[-(\Delta u_h, \Delta v)_S + (f, v)_S - (g_1, v)_{\partial S \cap \Gamma_{N1}} + (g_2, \partial_{\vec{n}} v)_{\partial S \cap \Gamma_{N2}}\right] \quad \forall v \in H_0 \tag{3.3.8}$$

Integrate by parts twice. The regularity of u_h on subelements makes this operation well defined.

$$\begin{aligned} B(e,v) \;&=\; \sum_S [(f - \Delta^2 u_h, v)_S - (g_1 - \partial_{\vec{n}} \Delta u_h, v)_{\partial S \cap \Gamma_{N1}} + (\partial_{\vec{n}} \Delta u_h, v)_{\partial S - \Gamma_{N1}} + \qquad (3.3.9) \\ &\quad (g_2 - \Delta u_h, \partial_{\vec{n}} v)_{\partial S \cap \Gamma_{N2}} - (\Delta u_h, \partial_{\vec{n}} v)_{\partial S - \Gamma_{N2}}] \qquad (3.3.10) \end{aligned}$$

On any point on the shared side between two neighboring subelements S and S', the normal vector $\vec{n}_S = -\vec{n}_{S'}$, so the normal derivative is negative as seen from a subelement's neighbor. Define the Laplacian jump on the Neumann boundary of the domain as

$$[\![\Delta u_h]\!]_{\Gamma_{N2}} \equiv 2(g_2 - \Delta u_h) \tag{3.3.11}$$

Define the Laplacian jump to be zero on Γ_{D2}, and define it on internal sides as:

$$[\![\Delta u_h(\vec{s})]\!]_{\partial S} \equiv \lim_{\vec{x} \to \vec{s}, \vec{x} \in S'} \Delta u_h(\vec{x}) - \lim_{\vec{x} \to \vec{s}, \vec{x} \in S} \Delta u_h(\vec{x}) \tag{3.3.12}$$

Define the Laplacian flux jump on its Neumann boundary as:

$$[\![\partial_{\vec{n}} \Delta u_h]\!]_{\Gamma_{N1}} \equiv 2(g_1 - \partial_{\vec{n}} \Delta u_h) \tag{3.3.13}$$

Define the Laplacian flux jump to be zero on Γ_{D1}, and define it on internal sides as:

$$[\![\partial_{\vec{n}} \Delta u_h(\vec{s})]\!]_{\partial S} \equiv \lim_{\vec{x}\to\vec{s}, \vec{x}\in S'} \partial_{\vec{n}_S} \Delta u_h(\vec{x}) - \lim_{\vec{x}\to\vec{s}, \vec{x}\in S} \partial_{\vec{n}_S} \Delta u_h(\vec{x}) \tag{3.3.14}$$

Rearrange the summation by taking half of the internal side terms on each subelement and assigning them to neighboring subelements, giving

$$\begin{aligned} &\forall v \in H_0, \\ B(e,v) &= \sum_S \Bigg[(f - \Delta^2 u_h, v)_S + \frac{1}{2}([\![\partial_{\vec{n}} \Delta u_h]\!], v)_{\partial S} - \\ &\qquad \frac{1}{2}([\![\Delta u_h]\!], \partial_{\vec{n}} v)_{\partial S} \Bigg] \end{aligned} \tag{3.3.15}$$

The error e can be decomposed into a finite dimensional interpolant $I_h e$ contained in H^h and a remainder $R_h e \equiv e - I_h e$. The bilinearity of $B(\cdot,\cdot)$ allows a similar decomposition:

$$B(e,e) = B(e, R_h e) + B(e, I_h e) \tag{3.3.16}$$

Combining (3.3.15) and (3.3.16) and using Galerkin orthogonality $B(e, v_h) = 0$ for $v_h \in H^h$ gives:

$$\begin{aligned} B(e,e) &= B(e, R_h e) \\ &= \sum_S \Bigg[(f - \Delta^2 u_h, R_h e)_S + \frac{1}{2}([\![\partial_{\vec{n}} \Delta u_h]\!], R_h e)_{\partial S} - \\ &\qquad \frac{1}{2}([\![\Delta u_h]\!], \partial_{\vec{n}} R_h e)_{\partial S} \Bigg] \end{aligned} \tag{3.3.17}$$

Applying the Cauchy-Schwarz inequality:

$$\begin{aligned} B(e,e) \;\leq\; & \sum_S \Bigg[\left|\left|f - \Delta^2 u_h\right|\right|_S \left|\left|R_h e\right|\right|_S + \frac{1}{2} \left|\left|[\![\partial_{\vec{n}} \Delta u_h]\!]\right|\right|_{\partial S} \left|\left|R_h e\right|\right|_{\partial S} + \\ & \frac{1}{2} \left|\left|[\![\Delta u_h]\!]\right|\right|_{\partial S} \left|\left|\partial_{\vec{n}} (R_h e)\right|\right|_{\partial S} \Bigg] \end{aligned} \tag{3.3.18}$$

On a quasi-regular family of meshes, using an H^2 conforming finite element with a stable local basis such as the HCT elements, interpolation theory [89] will give an upper bound for the approximation error on an element in Hilbert space norms.

To simplify subsequent notation, a generic constant "C" will refer to several constants which may differ from equation to equation. From interpolation theory, there exists a C dependent only on Ω for which:

$$\begin{aligned} & \forall v \in H^2(\Omega), \\ \left|\left|R_h v\right|\right|_K &\leq C h_K^2 \left|\left|v\right|\right|_{H^2(\Omega)} \\ \left|\left|\nabla R_h v\right|\right|_K &\leq C h_K^1 \left|\left|v\right|\right|_{H^2(\Omega)} \\ \left|\left|R_h v\right|\right|_{H^2(K)} &\leq C \left|\left|v\right|\right|_{H^2(\Omega)} \end{aligned} \tag{3.3.19}$$

For elements like the HCT triangle with mesh-independent splittings, the size h_S of a subelement S contained in element K is proportional to the element size h_K. Function norms on S are bounded by function norms on K, so the upper bound for the approximation error on macroelements is also an upper bound for the approximation error on their subelements:

$$\begin{aligned} \left|\left|R_h v\right|\right|_S &\leq C h_S^2 \left|\left|v\right|\right|_{H^2(\Omega)} \\ \left|\left|\nabla (R_h v)\right|\right|_S &\leq C h_S^1 \left|\left|v\right|\right|_{H^2(\Omega)} \\ \left|\left|R_h v\right|\right|_{H^2(S)} &\leq C \left|\left|v\right|\right|_{H^2(\Omega)} \end{aligned} \tag{3.3.20}$$

Inverting the master to physical element transformation $T : \hat{K} \to K$ lets us scale functions from S onto a subelement $\hat{S}$ of the master element.

$$\hat{v} \equiv v \circ T \tag{3.3.21}$$

$$||R_h v||^2_{\partial S} \le C h_S^{d-1} \left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|^2_{\partial \hat{S}} \tag{3.3.22}$$

Applying the trace theorem on the master subelement gives a bound for boundary norms in terms of interior norms:

$$||R_h v||^2_{\partial S} \le C h_S^{d-1} \left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|_{\hat{S}} \left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|_{H^1(\hat{S})} \tag{3.3.23}$$

Scaling back to S gives upper bounds for the boundary norms of approximation errors based on their interior norms:

$$\begin{aligned} ||R_h v||^2_{\partial S} &\le C h_S^{d-1} \sqrt{\left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|^2_{\hat{S}} \left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|^2_{H^1(\hat{S})}} \\ &\le C h_S^{d-1} \sqrt{\left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|^4_{\hat{S}} + \left|\left|\hat{v} - \hat{I}_h \hat{v}\right|\right|^2_{\hat{S}} \left|\left|\nabla(\hat{v} - \hat{I}_h \hat{v})\right|\right|^2_{\hat{S}}} \qquad (3.3.24) \\ &\le C h_S^{d-1} \sqrt{h_S^{-2d} ||R_h v||^4_S + h_S^{-d} ||R_h v||^2_S \, h_S^{2-d} ||\nabla(R_h v)||^2_S} \end{aligned}$$

Applying the interpolation results from (3.3.19) gives subelement boundary norms of approximation errors based on H^2 global norms:

$$\begin{aligned} ||R_h v||^2_{\partial S} &\le C h_S^{d-1} \sqrt{h_S^{8-2d} ||v||^4_{H^2(\Omega)} + h_S^{4-d} ||v||^2_{H^2(\Omega)} \, h_S^{4-d} ||v||^2_{H^2(\Omega)}} \\ &\le C h_S^3 ||v||^2_{H^2(\Omega)} \qquad (3.3.25) \\ ||R_h v||_{\partial S} &\le C h_S^{3/2} ||v||_{H^2(\Omega)} \end{aligned}$$

Performing a similar process will provide a bound on approximation errors of subelement boundary fluxes. First bound the flux norm in terms of the vector norm of subelement boundary gradients, and then scale back to the master subelement

$$\begin{aligned} ||\partial_{\vec{n}} R_h v||^2_{\partial S} &\leq ||\nabla R_h v||^2_{\partial S} \\ &\leq C h_S^{d-3} \left|\left|\hat{\nabla}(\hat{v} - \hat{I}_h \hat{v})\right|\right|^2_{\partial \hat{S}} \end{aligned} \tag{3.3.26}$$

apply the trace theorem on the master subelement

$$\leq C h_S^{d-3} \left|\hat{v} - \hat{I}_h \hat{v}\right|_{H^1(\hat{S})} \left|\hat{v} - \hat{I}_h \hat{v}\right|_{H^2(\hat{S})} \tag{3.3.27}$$

and again scale the result back to the physical subelement

$$\begin{aligned} ||\partial_{\vec{n}} R_h v||^2_{\partial S} &\leq C h_S^{d-3} \sqrt{\left|\hat{v} - \hat{I}_h \hat{v}\right|^2_{H^1(\hat{S})} \left|\hat{v} - \hat{I}_h \hat{v}\right|^2_{H^2(\hat{S})}} \\ &\leq C h_S^{d-3} \sqrt{\left|\hat{v} - \hat{I}_h \hat{v}\right|^4_{H^1(\hat{S})} + \left|\hat{v} - \hat{I}_h \hat{v}\right|^2_{H^1(\hat{S})} \left|\hat{v} - \hat{I}_h \hat{v}\right|^2_{H^2(\hat{S})}} \\ &\leq C h_S^{d-3} \sqrt{h_S^{4-2d} \left|R_h v\right|^4_{H^1(S)} + h_S^{2-d} \left|R_h v\right|^2_{H^1(S)} h_S^{4-d} \left|R_h v\right|^2_{H^2(S)}} \end{aligned} \tag{3.3.28}$$

Finally, again apply (3.3.19):

$$\begin{aligned} ||\partial_{\vec{n}} R_h v||^2_{\partial S} &\leq C h_S^{d-3} \sqrt{h_S^{8-2d} ||v||^4_{H^2(\Omega)} + h_S^{4-d} ||v||^2_{H^2(\Omega)} h_S^{4-d} ||v||^2_{H^2(\Omega)}} \\ &\leq C h_S ||v||^2_{H^2(\Omega)} \\ ||\partial_{\vec{n}} R_h v||_{\partial S} &\leq C h_S^{1/2} ||v||_{H^2(\Omega)} \end{aligned} \tag{3.3.29}$$

We currently have a bound on $B(e, e) = ||\Delta e||^2_\Omega$, but would like a bound on the full H^2 norm of the error. Coercivity of B on H_0 can be derived from regularity of the Laplacian operator Δ given certain boundary conditions. If $\Gamma_{D1} \cup \Gamma_{D2} = \partial\Omega$ and this boundary is sufficiently smooth, then the regularity result follows:

$$\begin{aligned} &\forall v \in H_0, \\ ||v||_{H^2(\Omega)} &\leq C(||\Delta v||_\Omega + ||v||_{H^{3/2}(\Gamma_{D1})} + ||\partial_{\vec{n}} v||_{H^{1/2}(\Gamma_{D2})}) \end{aligned} \tag{3.3.30}$$

When this regularity is applied to approximation error of a function which exactly satisfies the Dirichlet boundary conditions, $e|_{\Gamma_{D1}} = 0$ and $\partial_{\vec{n}} e|_{\Gamma_{D2}} = 0$, the result is an H^2 error bound:

$$||e||^2_{H^2(\Omega)} \leq C\, ||\Delta e||^2_{\Omega} \tag{3.3.31}$$

If the boundary is not smooth, but the problem has completely Dirichlet boundary conditions where $\Gamma_{D1} = \Gamma_{D2} = \partial\Omega$, it is still possible to prove the coercivity of B by extending Ω to a superset $\tilde{\Omega}$ with a smooth boundary. Given any function $v \in H_0$, extend it by $v(x) = 0 \quad \forall x \in \tilde{\Omega} - \Omega$, giving a function $\tilde{v} \in H^2_0(\tilde{\Omega})$. Elliptic regularity in this case gives us:

$$||\tilde{v}||_{H^2_0(\tilde{\Omega})} \leq C\, ||\Delta \tilde{v}||_{\tilde{\Omega}} \tag{3.3.32}$$

Because $\tilde{v} = 0$ on $\tilde{\Omega} - \Omega$, it follows that $||\tilde{v}||_{H^2_0(\tilde{\Omega})} = ||v||_{H^2_0(\Omega)}$. Because $\Delta\tilde{v} = 0$ on $\tilde{\Omega} - \Omega$, it follows that $||\Delta\tilde{v}||_{\tilde{\Omega}} = ||\Delta v||_{\Omega}$. Making these substitutions again gives the required error bound (3.3.31).

Applying the interpolation results from (3.3.25) and (3.3.29) to the right hand side of the error inequality (3.3.18), then applying the coercivity property (3.3.31) to the left hand side, the result is an upper bound on the square of the H^2 error:

$$\begin{aligned}
||\Delta e||^2_{\Omega} &\leq C \sum_{S} \Big[\left|\left| f - \Delta^2 u_h \right|\right|_S ||R_h e||_S + \frac{1}{2} ||[\![\partial_{\vec{n}} \Delta u_h]\!]||_{\partial S}\, ||R_h e||_{\partial S} + \\
&\qquad \frac{1}{2} ||[\![\Delta u_h]\!]||_{\partial S}\, ||\partial_{\vec{n}} R_h e||_{\partial S} \Big] \\
||e||^2_{H^2(\Omega)} &\leq C \sum_{S} \Big[\left|\left| f - \Delta^2 u_h \right|\right|_S h_S^2\, ||e||_{H^2(\Omega)} + \frac{1}{2} ||[\![\partial_{\vec{n}} \Delta u_h]\!]||_{\partial S}\, h_S^{3/2}\, ||e||_{H^2(\Omega)} + \\
&\qquad \frac{1}{2} ||[\![\Delta u_h]\!]||_{\partial S}\, h_S^{1/2}\, ||e||_{H^2(\Omega)} \Big]
\end{aligned} \tag{3.3.33}$$

Finally, dividing by $||e||_{H^2(\Omega)}$ gives the error bound

$$\begin{aligned}||e||_{H^2(\Omega)} \;\; &\leq \;\; C\sum_S \left[\left|\left| f-\Delta^2 u_h \right|\right|_S h_S^2 + \frac{1}{2}\, ||[\![\partial_{\vec{n}}\Delta u_h]\!]||_{\partial S}\, h_S^{3/2} + \right. \\ &\left. \frac{1}{2}\, ||[\![\Delta u_h]\!]||_{\partial S}\, h_S^{1/2} \right] \qquad (3.3.34)\end{aligned}$$

Accumulating the subelement error contributions on each element K gives an error estimator on macroelements:

$$\eta_K \equiv \sum_{S\subset K} \left[\left|\left| f-\Delta^2 u_h \right|\right|_S h_S^2 + \frac{1}{2}\, ||[\![\partial_{\vec{n}}\Delta u_h]\!]||_{\partial S}\, h_S^{3/2} + \frac{1}{2}\, ||[\![\Delta u_h]\!]||_{\partial S}\, h_S^{1/2} \right] \qquad (3.3.35)$$

For PSH elements, the third derivatives of shape function are zero, and the Laplacian flux jump terms vanish. For PSH elements in unforced problems, the interior residual terms likewise vanish, and the only nonzero terms in the error bound are the Laplacian jumps across element and subelement boundaries. For the HCT elements or for problems with forcing functions, all terms must be taken into consideration to give a proven upper bound on error. For problems with inexactly interpolated Dirichlet boundary conditions, an additional error term corresponding to the norm of this interpolation error is necessary.

Even an approximate error indicator, however, may be sufficient for directing AMR and obtaining a well-graded grid. A numerical example will demonstrate this, using an error indicator limited to a single term: the Laplacian jumps across element (but not all subelement) boundaries:

$$\eta'_K = \sqrt{h_K}\, ||[\![\Delta u_h]\!]||_{\partial K} \qquad (3.3.36)$$

Numerical experiments demonstrate that this reduced indicator performs very well, despite the elimination of terms from the more rigorous estimator and despite our use of it for problems where the simplified PDE and boundary assumptions do not all apply.

3.4 Adaptive Refinement for Steady Problems

With the error indicator (3.3.36), it is possible to automatically obtain a finite element mesh which is graded to efficiently resolve the solution to a new problem. By first finding approximate solutions on coarse meshes, an automatic algorithm can obtain enough information about the true solution to allow the creation of fine meshes which are adapted to the solution's most difficult to resolve features.

Practical adaptively-refined grids can be obtained iteratively. Begin with the coarsest mesh M_0 which defines the domain geometry. On each mesh M_i, perform the following steps:

1. Find the Galerkin approximate solution u_h^i, if possible using a projection of u_h^{i-1} as an initial solver iterate.
2. Apply the error indicator on each element in M_i.
3. Sort the elements by error, then uniformly refine some fraction F of the elements with the highest indicated error.
4. Refine any "unrefined islands": elements which are coarser than all their neighbors.

5. Refine away “level two mismatches”: elements with any neighbors that are more than one level more refined.
6. Call the resulting mesh M_{i+1}
7. Repeat until a specified number of degrees of freedom or error tolerance is reached.

Several of the steps in this algorithm are pragmatic choices for AMR performance, not strict mathematical requirements. Unrefined islands constrain the degrees of freedom of all neighboring elements, and if these neighbors have been chosen for refinement by the error indicator then the unrefined island appears to be a productive target for refinement itself. Requiring elements to be no more than one level more refined than their neighbors improves the likelihood that the family of meshes will be quasi-regular. Finally, choosing what fraction F of elements to refine is a tradeoff between computational cost and mesh quality. Refining more elements at each adaptive step causes a fine mesh to be reached in fewer adaptive steps, and thus with fewer expensive numerical solutions. Refining fewer elements at each adaptive step means that on average each element refinement decision is made based on error estimates from a finer grid, which gives more optimally graded meshes and a less expensive final solve.

3.5 Adaptive Refinement for Transient Problems

At each time step, the time discretization gives an elliptic boundary value problem, with known data from the solution at the previous time step.

As the character of the time-evolving solution changes, so will the character of the optimal meshes on which to discretize that solution. Optimally, the approximate solution at each time step should be found on a mesh which has been adapted to represent both the new and old solutions at that time step efficiently and accurately.

At each adaptive step, the algorithm will refine some elements and coarsen other elements to adapt the mesh to the evolving solution. Every active element in the mesh is a potential candidate for immediate refinement, which replaces that element with 2^d "child" elements. Not all child elements are candidates for immediate coarsening, however. Because coarsening replaces a child element and all its "sibling" elements with their common "parent", and because we do not want to reverse more than one level of refinement at each adaptive step, only elements without refined siblings are eligible for immediate coarsening.

Calculating the Laplacian jump error indicator gives a scalar error indicator value η_K on each active finite element K in the mesh. If a set of sibling elements are all active, a useful error indicator on their common inactive parent element K_p is:

$$\eta_{K_p} = c_\eta 2^n \sqrt{\sum_{K \subset K_p} \eta_K} \tag{3.5.1}$$

where n is the expected asymptotic order of convergence of the finite element family used in the norm approximated by the error indicator, and $c_\eta > 1$ is a constant used to make coarsening behavior more conservative. The asymptotic convergence rate n is used to make η_{K_p} a reasonable predictor of the error indicator on K_p during subsequent refinement steps, and c_η makes it less likely that coarsening will be based on an underpredicted K_p and thus followed by immediate re-refinement on subsequent adaptive steps.

To optimize the accuracy of the transient simulation, we use these AMR/C tools to roughly equidistribute the error indicator values between mesh cells, refining to improve accuracy where the error indicator values rise and coarsening to reduce computational cost where the error indicator values fall. There are many possible strategies for AMR/C. In many of the adaptive results herein, we choose to maintain a bounded degree of freedom count (and thus an upper bound on simulation memory usage and cost) by refining to a desired element count $n_d(t)$ as follows:

At time t_n, if the number of active elements n_a in the mesh exceeds $n_d(t_n)$, then enough coarsenable elements are flagged for coarsening to eliminate the excess, with those elements having the lowest error indicators η_{K_p} chosen first. If instead $n_d(t_n) > n_a$, then enough active elements are flagged for refinement to eliminate the deficit, with those elements having the highest error indicators η_K chosen first.

The remaining sets of unflagged refinable active elements and unflagged coarsenable parent elements are then sorted by error indicator values. Beginning with the elements having lowest η_{K_p} and highest η_K, elements are flagged in pairs, one active element for refinement and one parent element for coarsening, for as long as pairs can be found satisfying $\eta_{K_p} < \eta_K$.

If K is refined while the children of K_p are coarsened, then the number of elements in the mesh will be unchanged, the number of “raw” degrees of freedom in the mesh will be unchanged, and even after hanging node constraints are applied, the number of degrees of freedom in the mesh will be unchanged. If $\eta_{K_p} < \eta_K$, then this trade should provide a reduction in the total error on the mesh at approximately the same computational cost.

3.5.1 Adaptive Time Discretization

In many physical problems of interest, adaptive discretization is also necessary to efficiently resolve widely varying time scales. In the spinodal phase decomposition problem in Section 7.9, for example, the rapid changes in material concentration induced by phase separation and by the evolution of many labyrinthine interfaces require small time steps to simulate accurately. However, carrying out a simulation with such a small δt and uniform time steps would then become unnecessarily inefficient in later time steps where well-separated phases and long interfaces lead to slow, smooth time evolution which can be accurately approximated with long time steps.

To choose efficient and accurate time step sizes, we again attempt to equidistribute locally computable discretization error estimates. For each time step, the simulation is advanced from time t_n to $t_{n+1} \equiv t_n + \delta t$ in two different ways, by taking two time steps of size $\delta t/2$ to calculate the next solution u_{n+1} from u_n, and by taking a single time step of length δt to calculate $\hat{u}_{n+1}$ from u_n. The local relative time discretization error estimate for this time step is

$$\hat{e}_n \equiv \frac{||u_{n+1} - \hat{u}_{n+1}||}{\max u_{n+1}, \hat{u}_{n+1})} \tag{3.5.2}$$

and the global relative time discretization error estimate is

$$e_n \equiv \frac{\hat{e}_n}{\delta t} \tag{3.5.3}$$

The time step adaptivity is based on a target error tolerance e_{TOL} and a maximum acceptable error tolerance e_{MAX}. If at a time step n the solution achieves $e_n < e_{MAX}$, then the time solver proceeds with a new time step size

$$\delta t_{n+1} \equiv \delta t_n \cdot \left(\frac{e_{TOL}}{e_n} \right)^{1/p} \tag{3.5.4}$$

where here p is the global convergence rate of the timestepping algorithm being used; e.g. $p = 2$ for the trapezoidal rule.

If at a time step n the time solver finds $e_n < e_{MAX}$, or if the nonlinear solver fails, then that time step is discarded and is recalculated with a smaller δt_n.

The finite element techniques used in this work are compatible with a wide variety of time discretization methods. For other techniques in time truncation error estimation and adaptive time step control see [106, 127].

3.6 Biharmonic Benchmark Problem

To verify and demonstrate the performance of this AMR scheme with macroelements on fourth-order problems, we construct a benchmark biharmonic problem with a singularity and a closed form analytic solution. We anticipate that this problem will be of general value to the engineering community as a benchmark for verification studies of numerical techniques [110]. Selection of benchmarks for Stokes and Navier-Stokes flow studies is a well known difficulty. For example, the driven cavity problem is the best known benchmark for incompressible flow with a sharp singularity, but from a modeling standpoint the discontinuous boundary conditions and lack of an analytic solution are unfortunate. The following proposed new benchmark problem is constructed with a known analytic solution for Stokes flow with a singularity, and could be extended to Navier-Stokes flow by adding the appropriate manufactured forcing function on the domain interior.

We begin with a related exterior flow problem that has an analytic series solution in polar coordinates. From this, a specific test problem is constructed by choosing boundary conditions which correspond to only a single term in that series.

Benchmark Problem: Consider an attached viscous fluid flow around the edge of a flat plate on the positive x axis. Any 2D incompressible velocity field can be expressed by a scalar streamfunction u, unique up to a constant, where the velocity $\vec{v}$ is given by the curl of u: $\vec{v} \equiv \partial_y u \vec{e}_x - \partial_x u \vec{e}_y$. In the low Reynolds number limit, flow is described by the incompressible Stokes equations; these are equivalent to the streamfunction equation $\Delta^2 u = 0$. No-slip and no-penetration boundary conditions on the velocity at the plate imply that the gradient of the streamfunction is zero along the plate. Because the streamfunction is only unique up to a constant, arbitrarily choose to set it to $u = 0$ at the plate tip. Then, the zero gradient of u leads to the essential boundary conditions $u = 0$ and $\partial_{\vec{n}} u = 0$ along the entire plate. These boundary conditions make it simple to find a family of admissible solution modes by expressing the problem in polar coordinates r and θ and using separation of variables. For the benchmark problem, we choose one of the two dominant streamfunction modes near the plate edge, the mode corresponding to attached flow around the edge:

$$u \equiv r^{3/2} \left(\sin(3\theta/2) - 3\sin(\theta/2)\right) \tag{3.6.1}$$

The streamfunction u and vorticity Δu of this flow are plotted in Figure 3.1. Although the streamfunction appears smooth, its second derivatives are singular at the plate tip.

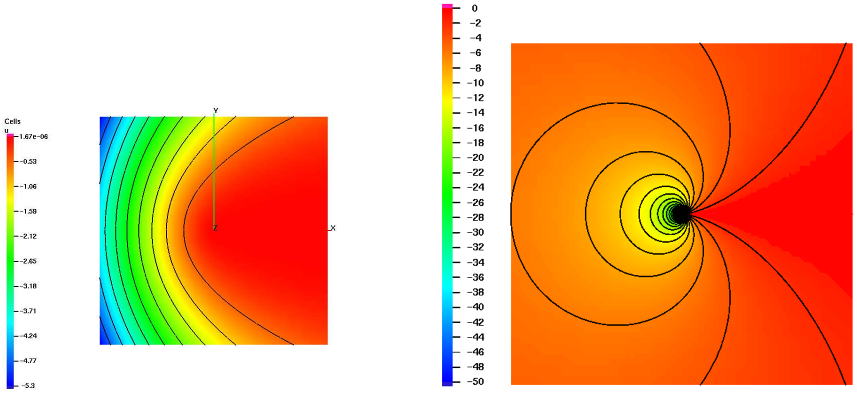

Figure 3.1: Streamfunction (left) and vorticity (right) for inviscid flow around a sharp edge.

We set the plate location P to be the positive x axis, and solve the streamfunction problem $\Delta^2 u = 0$ on the domain $(-1,1) \times (-1,1) - P$. The boundary conditions along P are as described above, and we use the analytic u to give essential boundary conditions on the outer boundary. Using the notation introduced in Section 3.3, in this problem $\Gamma_{D1} = \Gamma_{D2} = \partial\Omega$.

Note that except for along the plate, the boundary conditions for this problem are not exactly representable by the piecewise cubic boundary values and piecewise quadratic boundary fluxes of an HCT macroelement approximation space. In the finite element approximation, rather than solve the weak approximation (3.3.6) with exact boundary conditions, we enforce the essential boundary conditions using a penalty method as described in Section 5.6. Recall the Galerkin formulation of the approximate problem: to find $u_h \in H^h + u_B$ such that

$$B(u_h, v_h) = L(v_h) \quad \forall v_h \in H_0^h \tag{3.6.2}$$

Instead, choose a small penalty value $0 < \epsilon \ll 1$, and find $u_h \in H^h$ such that $\forall v_h \in H^h$,

$$\begin{aligned} B(u_h, v_h) + \frac{1}{\epsilon}\left[(u_h, v_h)_{\Gamma_{D1}} + (\partial_{\vec{n}} u_h, \partial_{\vec{n}} v_h)_{\Gamma_{D2}}\right] &= && (3.6.3) \\ L(v_h) + \frac{1}{\epsilon}\left[(u_B, v_h)_{\Gamma_{D1}} + (\partial_{\vec{n}} u_B, \partial_{\vec{n}} v_h)_{\Gamma_{D2}}\right] & && (3.6.4) \end{aligned}$$

This is equivalent to solving the weak formulation of the biharmonic problem with the Robin boundary conditions

$$\begin{aligned} \epsilon\, \partial_{\vec{n}} \Delta u\big|_{\Gamma_{D1}} &= u - u_B && (3.6.5) \\ \epsilon\, \Delta u\big|_{\Gamma_{D2}} &= \partial_{\vec{n}} u - \partial_{\vec{n}} u_B && (3.6.6) \end{aligned}$$

As $\epsilon \to 0$, these boundary conditions approach the desired Dirichlet conditions.

The approximate solutions in Figure 3.1 are obtained using `libMesh` [86], a C++ finite element library designed to support parallel computations on adaptive hybrid meshes. To support calculations of problems in $W^{2,p}$ spaces, we have added library support for several C^1 elements including the cubic Clough-Tocher triangles used for this numeric example. Support for hanging node constraints on C^1 elements in adaptive meshes was implemented via the projection method described in Section 3.1, and support for calculations of basis function second derivatives was added to the library.

Linear system coefficients are calculated by numerical integration with double precision arithmetic, using a Gaussian quadrature rule exact for sixth degree polynomials on each subtriangle. The adaptive error indicator derived in Section 3.3 is used to guide AMR, and "exact" error values are numerically integrated using the same quadrature rule. The linear solver uses a stabilized biconjugate gradient (BICG) solver, with an incomplete LU factorization for preconditioning. A penalty value $\epsilon = 10^{-10}$ was used to define the boundary conditions.

To fairly compare uniform and adaptive results, in Figure 3.2 approximation errors are plotted against the number of unconstrained degrees of freedom in the finite element space.

The second derivatives of the exact solution u become singular at $r = 0$, and this singularity limits the accuracy that can be expected from uniformly refined meshes. The exact solution is not contained in $H^3(\Omega)$, and referring back to the error estimates in Section 2.5 note that in the H^2 norm approximate solutions on uniform meshes are only expected to converge with sublinear accuracy with respect to h. On a uniformly refined 2D mesh, h is inversely proportional to the square root of the number of degrees of freedom N, and the error plot in Figure 3.2 confirms a convergence rate below $\mathcal{O}\left(N^{-1/2}\right)$ in the uniform case.

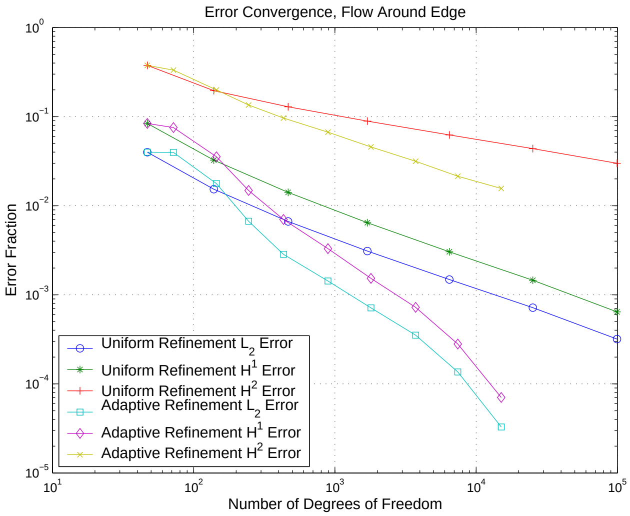

Figure 3.2: Galerkin approximation error in L_2, H^1, and H^2 norms, scaled by the solution norm and evaluated on both uniform and adaptive meshes for the cusp flow problem.

For the adaptive refinement strategy, we repeatedly refine the 25% of elements with the highest error levels as reported by the laplacian jump error indicator. The choice of 25% refinement in each step is arbitrary.

The adaptive refinement algorithm appropriately grades the mesh into the singularity, as shown by Figure 3.3, which displays an adaptive mesh at a typical stage of the refinement process. As expected, here the AMR algorithm adds the most degrees of freedom near the plate edge, to resolve the solution there. The error in this sequence of adaptively meshed solutions quickly drops below the errors in uniform solutions with equal numbers of degrees of freedom.

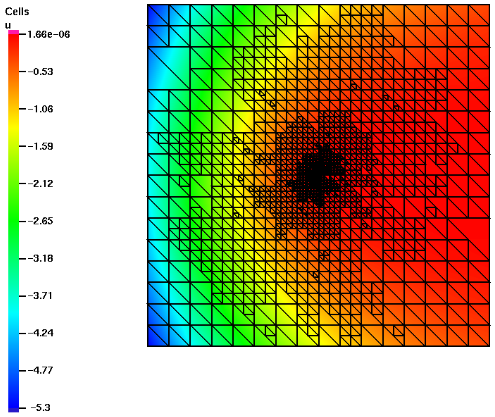

Figure 3.3: An adaptively refined mesh obtained for the cusp flow problem.

Chapter 4

Software Design

4.1 Finite Element Library

The parallel adaptive work described here applies and extends the capability of the `libMesh` framework [86]. Core features of the library include:

1. Support for mixed finite elements on hybrid unstructured grids in one through three dimensions.
2. Adaptive mesh h-refinement of Lagrange elements with hanging nodes
3. Parallel system assembly and solution
4. Integration with third party software packages such as:
    (a) PETSc and LASPack sparse linear algebra
    (b) METIS and ParMETIS mesh partitioning
    (c) Triangle and Tetgen mesh generation
5. Mesh and solution export and import using common data and visualization formats
6. Inline API documentation with Doxygen [73]

Some of the new contributions made as part of this dissertation work include:

1. Macroelement construction and quadrature rules
2. C^1 macroelement, Hermite finite element classes
3. Basis Hessian calculations
4. Parallel adaptivity for general element types
5. Projection, interpolation for general element types
6. Parallel distributed mesh data structures and algorithms
7. New nonlinear FEM solver framework
8. New time integration framework
9. Adaptive time integration
10. Additional spatial adaptivity strategies
11. Additional error estimators
12. Utilities for parametric and Monte Carlo studies

4.2 Object Oriented Design Abstractions

The `libMesh` software has been developed via a group effort to create general purpose parallel adaptive finite element tools. Because `libMesh` is available to the scientific community as a publically downloadable open source project, it has attracted users and collaborators both nationally and internationally. What makes this collaboration feasible, however, is the use of Object Oriented Programming principles to modularize the code, making it easier for new developers to add additional capabilities and features without compromising the existing Application Programming Interfaces.

`libMesh` is written in C++ [93] and uses abstract base classes to represent fundamental building blocks of finite element software. Both library code and user code call methods from the abstract interfaces which provide generic support for a wide variety of possible implementations. The C++ language virtually dispatches those function calls to concrete library classes which implement those interfaces for a specific type of object. Standard design patterns [65] such as factory methods, template methods, and strategy classes make it easy for user code to use these capabilities.

A few core examples illustrate this concept. In Figure 4.1, a simplified part of an inheritance tree is shown, which ends with geometric element objects as specific as a 27-noded hexahedron, implemented in the `Hex27` class. Code common to all hexahedra is shared with all classes which derive from the `Hex` parent class. Code common to all three-dimensional elements is shared with all derivatives of the `Cell` grandparent class. A `Cell` is a type of `Elem`, which is the only class which most `libMesh` application codes need to interact with to perform operations on and calculations for arbitrary geometric elements. Finally, `libMesh` can associate degrees of freedom with geometric elements by using the parent `DofObject` class internal to the library.

Similarly, quadrature rules and finite elements are represented by class hierarchies in Figures 4.2 and 4.3. Application physics routines initialize an `FEBase` reference with each geometric `Elem` object in the mesh in turn, and then use the `FEBase` interfaces to request shape function values, derivatives,

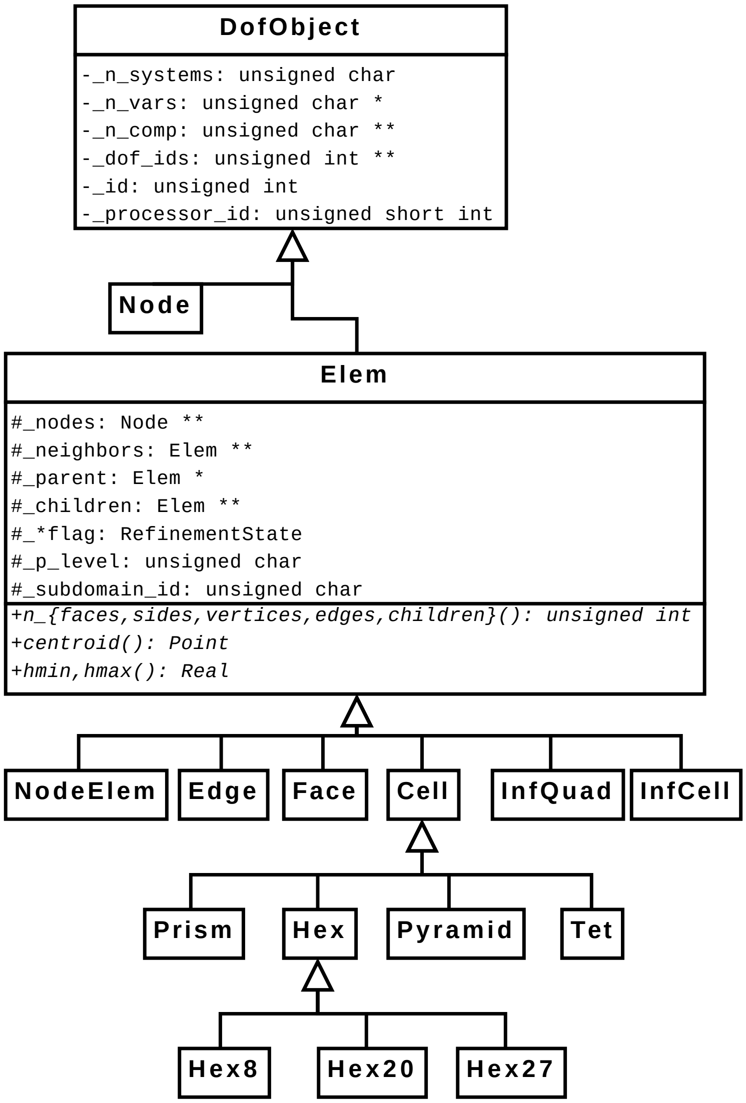

Figure 4.1: A simplified UML diagram of part of the `libMesh` geometric element hierarchy.

etc. as necessary to evaluate the desired weighted residual equations and their derivatives. A quadrature rule object derived from the `QBase` class uses the global polynomial degree information supplied by the `FEBase` object, along with any local p-refinement information from the `Elem` object, to choose an appropriate quadrature to use. The combination of a specific finite element family type, quadrature rule type, and geometric element determines the specific calculations performed by the library; for example, when using a `QGauss` object and `FE<LAGRANGE>` object initialized with a `Hex27` element, the application physics would be given shape function information appropriate to the integration of Lagrange basis functions on that hexahedron using a Gaussian quadrature rule. Using abstract interfaces enables a great degree of modularity, separating the users' physics code from the specific choices of spatial discretization. It also enables a great degree of code reuse, as algorithms which apply to all geometric elements, all finite elements, etc. need only be written once using the abstract interfaces.

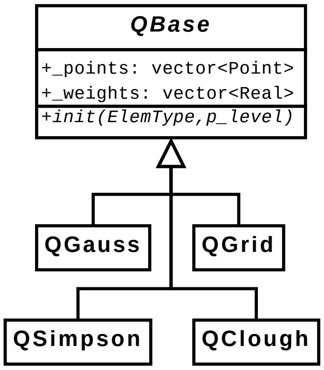

Figure 4.2: A simplified UML diagram of part of the `libMesh` quadrature rule class hierarchy.

The `libMesh` library has been expanded to facilitate the present work

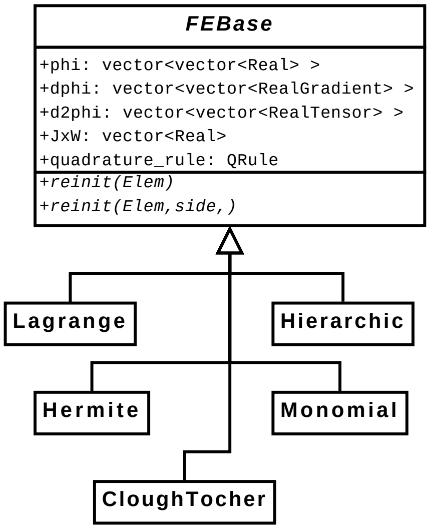

Figure 4.3: A simplified UML diagram of part of the `libMesh` finite element class hierarchy.

by adding an additional abstraction, this time designed to represent entire Initial/Boundary Value Problem systems of equations. Instead of requiring each application code to implement a nonlinear solver scheme and time stepping scheme, and to do so for the entire system of weak equations at once, the new `FEMSystem` object simply requires user code to supply the weak residuals (and optionally Jacobians for efficiency) of constraint equations $G_i(u, v_i) = 0$ and time derivative terms $(\frac{\partial u}{\partial t}, v_i) = F_i(u)$ integrated over a single mesh element. These terms are defined by abstract interfaces in the `FEMSystem` base class shown in Figure 4.4, and application physics is then written by defining a subclass with concrete implementations of those interfaces. Simulation programs instantiate subclasses of a nonlinear solver object as in Figure 4.6 to find algebraic system solutions, as well as subclasses of a `TimeSolver` object as in Figure 4.5 to do time integration (or to find a steady-state solution, or eigenvalues and eigenfunctions, etc.).

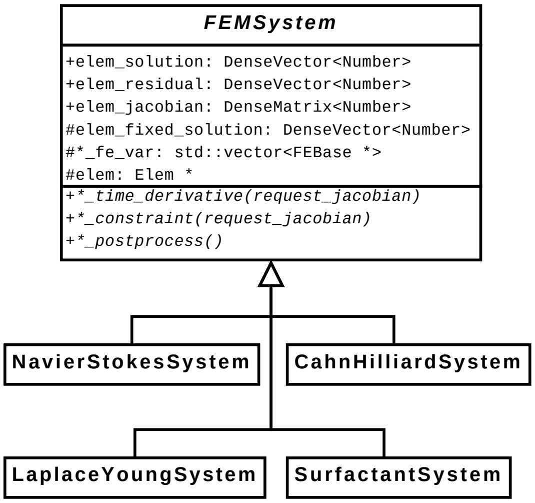
FEMSystem
+elem_solution: DenseVector<Number>
+elem_residual: DenseVector<Number>
+elem_jacobian: DenseMatrix<Number>
#elem_fixed_solution: DenseVector<Number>
#*_fe_var: std::vector<FEBase *>
#elem: Elem *
+*_time_derivative(request_jacobian)
+*_constraint(request_jacobian)
+*_postprocess()
NavierStokesSystem
CahnHilliardSystem
LaplaceYoungSystem
SurfactantSystem

Figure 4.4: A simplified UML diagram including the `libMesh` FEMSystem boundary value problem abstract base class. Application code authors write physics-specific subclasses of FEMSystem to implement particular mathematical models.

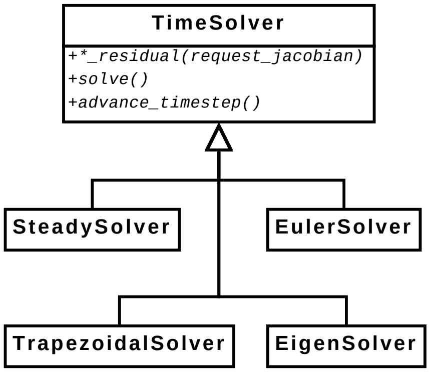

Figure 4.5: A simplified UML diagram of part of the `libMesh` ODE solver class hierarchy.

There are several benefits of this type of design. It has enabled easier implementation of new physics models in the diverse application studies to

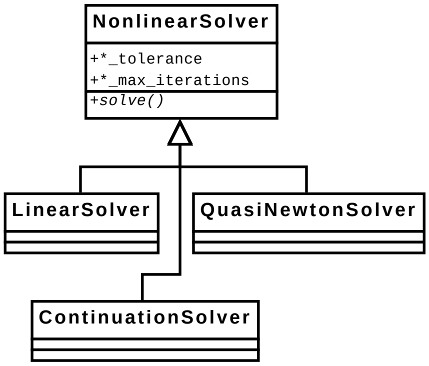

Figure 4.6: A simplified UML diagram of part of the `libMesh` nonlinear algebraic solver class hierarchy.

follow, encapsulated implementation details from application code, improved testing coverage and reliability, and given library-level code better access to specific application physics.

The first benefit, easier implementation of new and more complex systems of equations, was the primary motivation for this code refactoring. This dissertation includes experiments based on physical systems which have much fundamental physics in common but which model that physics with significantly differing sets of equations. By factoring out as much common code as possible, the total work necessary to implement several physics applications has been significantly reduced. The divergence-free flow, thin film flow, and phase decomposition studies to come are developed using many shared components, with differences only in physics-specific code.

Encapsulation of implementation details from user code is important not just because it reduces the complexity of user code, but because it makes that code more extensible. With the `FEMSystem` framework, once a core library feature like an adaptive time-stepping strategy or a new inexact Newton solver has been written, using it in a simulation is simply a matter of instantiating an object of the new class, without having to modify existing physics code. The `AdaptiveTimeSolver` algorithms used in the Cahn-Hilliard parameter studies to follow, for example, were first tested on more straightforward linear equations which allowed for easier development and debugging.

The improvement in testing coverage comes about because of the reuse of code. Code which has been factored out of physics-dependent software files is executed in several different applications, each of which is a new opportunity to shake out bugs. Code which has been generalized enough to add to the core `libMesh` library receives public testing as well. Adding new features becomes more worthwhile and more practical when they only need to be written and debugged once, rather than once for each time they are used.

Finally, by giving the `libMesh` library itself the ability to evaluate the weighted residual equations for a specific application on an element-by-element level, we can enable library algorithms that were impractical to implement when user code was required to perform a full system assembly itself. In the present work, the use of finite differenced element residuals to numerically approximate Jacobians on each element enabled rapid prototyping of new equations. Using numeric instead of analytical Jacobians in nonlinear solvers provides solutions that may be more expensive in CPU time but are cheaper in programmer time and less error-prone. Even when analytical Jacobians were implemented for each application code, the numerical Jacobian capability still provided a useful error detection system, catching bugs in analytical

Jacobian code by comparing numerical and analytic results entry-by-entry. Future uses of the new element-by-element physics API may include improved physics-dependent error estimators or adjoint techniques for inverse problems.

4.3 Parallel Implementation

Parallel execution of `libMesh` applications is supported on shared memory, distributed memory, and multilevel parallel computers. The Intel Threading Building Blocks [109] library is used to enable parallel computation on groups of processors (or, equivalently, processor cores) that share a single memory address space. In a distributed memory "cluster" computing environment, the Message Passing Interface (MPI) standard [92] is used to enable parallel communication between processors or groups of processors that access separate address spaces.

In `libMesh`, degrees of freedom on a finite element mesh are stored topologically on degree of freedom objects corresponding to element vertices, edges, faces, and interiors. For each degree of freedom, the corresponding basis function has support on the set of elements which "contain" its degree of freedom object, where an element is considered to contain its own vertices, edges, etc. This definition is independent of any internal edges and vertices which are part of a macroelement splitting. By treating any associated degrees of freedom as ordinary element interior degrees, the software storing them does not need to be aware of subelements, and can operate independently of the element types in use on a mesh.

The parallel distribution of global vectors and matrices follows directly from the parallel subdivision of the mesh itself; the challeneges of load balancing [74] are then handled by the partitioning and repartitioning of the mesh. A partitioner such as METIS assigns each element of the mesh to a particular subdomain, and all degrees of freedom which are assigned to with elements from only a single subdomain are "owned" by the processor associated with that subdomain. Degrees of freedom on the boundary of multiple subdomains could be assigned to any of the processors associated with those subdomains. In `libMesh` such degrees of freedom are simply assigned to the eligible processor with the lowest MPI rank number. The degree of freedom numbering is performed subdomain-by-subdomain, so that each processor owns a contiguous set of degree of freedom indices.

While each degree of freedom may only be owned by one processor, some may still be synchronized with "ghost" degrees of freedom on neighboring processors. Calculations along interfaces between neighboring elements, for example, can require data from "ghost" elements adjoining the local processor's subset of the mesh. When operating in parallel, this data is kept synchronized by `libMesh` before every assembly operation and synchronized by the linear solver package before each matrix-vector multiply.

To enable parallel solutions of the nonlinear algebraic systems generated by our finite element code, the `libMesh` algebra interface has a concrete implementation which uses the PETSc solver library [8] to perform sparse linear solves. The Krylov algorithms and linear preconditioner options available in PETSc are thus made available in `libMesh`. Most of the results to follow below are obtained using the Generalized Minimal Residual Method with Jacobi preconditioning which is easily parallelized, or with parallel block Jacobi preconditioning using an ILU subpreconditioner on each block. The PETSc nonlinear solvers are also now available via `libMesh` interfaces; however, in our numerical experiments better performance was obtained by using the PETSc linear solvers within the hand-coded inexact Newton nonlinear solver described in Section 4.4.

Mesh Parallelization In most of the results below, the resulting partitioned mesh is stored as an object of the `SerialMesh` class. `SerialMesh` objects store a copy of the entire mesh on every processor, in vectors of `Node` and `Elem` objects which represent each node and element. In the course of an adaptive simulation, a `SerialMesh` is kept in a conceptually identical state by synchronizing all mesh refinement and coarsening flags between processors. When applying the same AMR/C operations to the same mesh, each processor ends up creating and deleting new elements and nodes in the same order, and so the location and id of each element and node remains consistent from processor to processor. This design works well for coarse and moderate-scale parallelism, but it limits scalability on very large clusters and it can cause problems on very fine meshes to be limited by per-processor available memory.

To enable memory savings and better parallel speedup on large-scale problems, a `ParallelMesh` class has been developed and is being tested. Objects of this class are divided in memory to improve scaling on distributed-memory clusters. Each `ParallelMesh` object uses hash maps rather than vectors to store nodes and elements. Hash maps retain the $\mathcal{O}(1)$ index lookup efficiency of vector-based storage, but for N_e elements distributed among N_p

processors, a hash map only has $\mathcal{O}(N_e/N_p)$ memory requirements, rather than the $\mathcal{O}(N_e)$ memory usage of full vectors. On each processor, instead of allocating copies of every element and node in the mesh, a `ParallelMesh` allocates copies of only local elements and of "semilocal" elements which share nodes with local elements. Semilocal elements include ancestor elements (i.e. parent elements as described in Section 3.5, parents of parent elements, etc.) which are needed for various operations on hierarchically-refined meshes, as well as neighbor elements, which are needed for discontinuous Galerkin calculations, certain flux-based and patch-recovery-based error estimators, and constraint calculations on hanging nodes.

The memory savings enabled by mesh parallelization is illustrated in Figure 4.7, a graph of per-node memory usage for a benchmark application tested with a uniformly refined sequence of meshes. Although the hash table overhead makes a `ParallelMesh` take up slightly more memory when constructed on a single node, on two or more nodes the improvements seen with a parallel mesh can be dramatic. Even on the relatively small cluster used here (four quad-processor nodes with 4GB memory each), much larger problems can be solved before running out of physical memory.

The memory efficiency and scalability improvements enabled by a distributed memory mesh come at a cost of increased network communication and increased code complication when adaptive refinement and coarsening are brought into play. The following examples illustrate some of the increasing complexity:

Mesh refinement creates new nodes and elements which need to be given

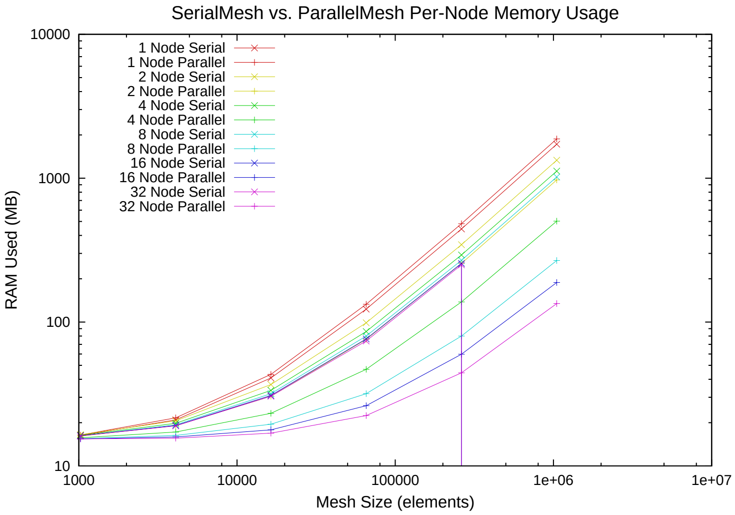

Figure 4.7: SerialMesh vs. ParallelMesh per-node resident memory usage, for a benchmark application using uniformly refined meshes on varying numbers of processors.

globally unique ids. A refined `libMesh` mesh is a hierarchical data structure (binary tree, quadtree, or octree, depending on dimension), similar to other parallel octree work [124], but rooted in a fully unstructured coarse mesh which may also contain pyramid, prism, or tetrahedral elements. Parallel adaptive refinement of pure tetrahedral meshes [45] is also found in the literature, using edge collapse techniques rather than hierarchic refinement and hanging nodes. In `libMesh`, where each processor has a unique integer processor id (i.e. the MPI rank), a processor with id r is allowed to temporarily reserve any element or node id i satisfying $i \bmod (N_p + 1) = r$. This ensures global id uniqueness when the object being identified is first created. New elements and nodes are then synchronized between processors based on a unique topological or geometric identification, after which a separate parallel renumbering operation then reassigns element and node ids into contiguous blocks for later convenience. This parallel renumbering must be done in four stages for efficient parallel operations: one pass to count the number of local objects, followed by a parallel broadcast of the number of objects on each processor, a second pass to actually assign each local object an id, and a second parallel communication to request ids from semilocal objects. This parallel renumbering algorithm is an $\mathcal{O}\left(N_e/N_p + N_p\right)$ operation. A more straightforward numbering scheme, where processor r completely finishes numbering before telling processor $r+1$ which id to use to start numbering, would have required $\mathcal{O}\left(N_e\right)$ runtime, because although each processor would still only be performing $\mathcal{O}\left(N_e/N_p\right)$ work, each processor would also spend $\mathcal{O}\left(N_e\right)$ time waiting for the other processors to finish.

Mesh repartitioning requires nodal geometry and element connectivity data to be transferred from processor to processor. In `libMesh`, the fundamental data defining each element or node is first converted into a format suitable for an MPI message, then collected into groups for each destination processor, and finally transmitted and used to construct new objects by the elements' and nodes' new owners.

Mesh refinement flags require processing to satisfy level-one rules and other such heuristics while simultaneously maintaining consistency between processors. To accomplish this, a processing loop first makes heuristic updates to its local element flags as necessary, then updates flags on its semilocal elements to match values requested from neighboring processors, then repeats until every processor has consistent and valid flags.

Element neighbor connectivity information is not as simple to describe when a ghost element's neighbors may not all exist on the local processor. To signify this condition, a "`RemoteElem`" singleton object is created, which acts as a placeholder target for such topological links. It can be complicated to maintain the correctness of these remote element links over the course of repeated adaptive refinements and coarsenings in a transient problem.

Finally, even with a working parallel data structure, making practical use of it may require many auxilliary algorithms to be parallelized as well. File input/output requires either complicated cooperation between all nodes or less efficient chunk-by-chunk data serialization to a single I/O node. Mesh generation can be done by building a coarse serial mesh, distributing it in parallel, then performing parallel refinement, but generating the original mesh in parallel may be more convenient or efficient. Even simple operations such as calculating a bounding box or finding the element containing a desired point may need to be re-examined with parallel operation in mind.

For certain common operations, the algorithm developments which were necessary to support distributed memory `ParallelMesh` data structures also proved to be beneficial when used on shared memory `SerialMesh` data structures. Calculations using `libMesh` with `SerialMesh` objects on multiple processors have always parallelized the most computationally expensive steps in a finite element code: the solution of the nonlinear algebraic systems of equations, and the assembly of the residuals and Jacobians of those equations. However, other steps that are necessary on adaptive unstructured meshes have turned out to be just as amenable to parallelization. Hanging node constraint equations are now calculated only on local processors, then synchronized over the network. Degree of freedom numbering is now done by each processor for its own local subset of the mesh, then broadcast to other proceseors while numberings for those processors' subsets are received. Localization of work, which is necessary with a parallelized mesh structure that only gives each processor access to local and semilocal data, is still reasonable even when processors would be capable of redundantly duplicating each others' workload. On a serialized mesh, the synchronization between processors is still an $\mathcal{O}\left(N_e\right)$ operation, so the localized algorithms are not asymptotically more efficient than a simple duplication of work. However, on a distributed parallel mesh, the only data in need of synchronization lies on ghost elements. A typical d-dimensional mesh has on the order of $\mathcal{O}\left(N_p^{1/d} N_e^{(d-1)/d}\right)$ ghost elements. So, for a simulation parallelized in such a way that $N_e \gg N_p$, mesh synchronization costs are limited, and parallel mesh modifications are expected to be dominated by the $\mathcal{O}\left(N_e/N_p\right)$ asymptotic cost of the local workload.

The computational benefits of serial vs. parallel mesh data structures depend on the size of the mesh being used, on the number of nodes being used in a distributed memory machine to operate on that mesh, and on the memory capacity of each node. For small and intermediate-sized problems, the wasted CPU effort required to perform serial mesh operations on every node of a cluster may be worthwhile. This duplication of work avoids the many network messages which would be required to perform synchronization steps and maintain consistency between different processors' parts of a parallelized mesh. On finer meshes running on more processors, the improved scalability of mesh operations on a parallel mesh can reverse this tradeoff. The more frequent synchronization steps required by a parallel mesh are also relatively less expensive on larger meshes, because at each step each processor needs only to receive information about its own ghost elements, not information about every element in the mesh as is required by a serial mesh. The clearest benefits for parallel mesh data structures are on very fine meshes. If a fully-refined serial mesh would be too large to fit alongside other finite element and linear algebra data structures in the free physical memory on a single node, then parallel distribution of that mesh among all the nodes in a cluster is an indispensible feature to have.

4.4 Newton-Krylov Methods

For most of the physical systems simulated in this work, discretization in space and time eventually reduces the simulation problem to the solution of nonlinear systems of algebraic equations, $F(\vec{u}) = 0$ at each time step. When the nonlinearity of F is not strong, such as in the slowest flow regimes in Section 5.4, successive approximation can be used to find a linearly converging sequence $\vec{u}_k$ approaching the desired solution. However, to reliably solve general nonlinear problems and to attain rapid solution convergence, a more sophisticated solution procedure is preferable.

When F possesses Lipschitz-continuous derivatives in a neighborhood of its zero, Newton's method converges and attains asymptotically quadratic convergence from starting iterates sufficiently near this root [99]. However, even when this assumption holds for a particular system of equations, Newton's Method is limited in the context of solutions of nonlinear discretized PDEs, by the difficulty of obtaining efficient solutions to the large linear systems which must be solved at each iteration, and by the limited domain of attraction in which a successful starting iterate must lie. For some problems, an appropriate initial Newton iterate may be obtained from the results of another method such as successive approximation or parameter continuation; however our inexact Newton solver is intended to reliably provide convergence even from an initial iterate outside the region of convergence of a strict Newton scheme.

At each iterate $\vec{u}^{(k)}$ in a Newton iteration, a linear approximation $L \approx F$ can be written as

$$L(\vec{u}) = F(\vec{u}^{(k)}) + F'(\vec{u}^{(k)})(\vec{u} - \vec{u}^{(k)}) \tag{4.4.1}$$

Assuming $F'(\vec{u}^{(k)})$ is invertible, the approximation L has a single root which is used as the next Newton iterate: $L(\vec{u}^{(k+1)}) = 0$ where

$$\vec{u}^{(k+1)} = \vec{u}^{(k)} - \left[F'(\vec{u}^{(k)})\right]^{-1} F(\vec{u}^{(k)}) \tag{4.4.2}$$

This is often an impractical equation to solve exactly in the context of a residual F in a Finite Element Method formulation. At each step the Jacobian F' is typically a sparse $N \times N$ matrix, requiring $\mathcal{O}(N)$ memory to store in a sparse format and $\mathcal{O}(N)$ CPU time to use in a matrix-vector product. Its inverse $[F']^{-1}$, however, is typically not a sparse matrix, and its many non-zero entries lead to $\mathcal{O}(N^2)$ storage requirements. Even the action of $[F']^{-1}$ on a particular residual vector typically requires $\mathcal{O}(N^3)$ CPU time to calculate via "exact" direct solvers, and in this context the meaning of "exact" typically depends on interactions between the conditioning of the matrix and the floating point error of the computer used to solve the linear system. For all but the smallest problems, indirect linear solvers are a more efficient way to obtain a solution to a desired tolerance. The nonlinear solver newly implemented in `libMesh` is a modification of a standard Newton-Krylov method, which uses preconditioned Krylov iterations for the linear solve at each Newton step until the specified relative linear system residual reduction for that step is achieved, as shown in Figure 4.8.

By beginning with a linear tolerance well in excess of the final desired nonlinear tolerance, the Newton-Krylov method avoids wasting computational effort on "oversolving" early linear systems. But, by choosing residual reduction tolerances for the Krylov solver that adaptively decrease as the nonlinear residual decreases, the Newton-Krylov method maintains the expected

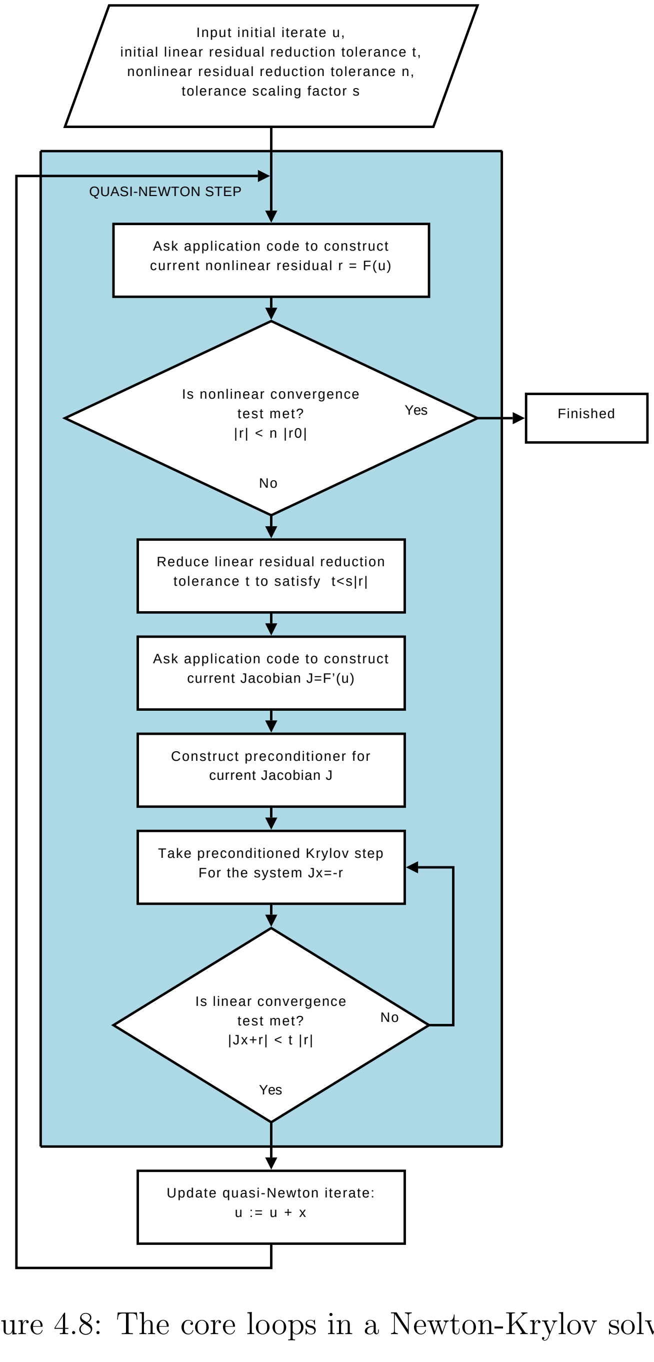
Input initial iterate u,
initial linear residual reduction tolerance t,
nonlinear residual reduction tolerance n,
tolerance scaling factor s
QUASI-NEWTON STEP
Ask application code to construct current nonlinear residual r = F(u)
Is nonlinear convergence test met? |r| < n |r0|
Yes
Finished
No
Reduce linear residual reduction tolerance t to satisfy t<s|r|
Ask application code to construct current Jacobian J=F'(u)
Construct preconditioner for current Jacobian J
Take preconditioned Krylov step For the system Jx=-r
Is linear convergence test met? |Jx+r| < t |r|
No
Yes
Update quasi-Newton iterate: u := u + x

Figure 4.8: The core loops in a Newton-Krylov solver.

quadratic convergence near the root of F. The choice of Krylov solver tolerances is a subject of extensive prior research [24, 49, 51, 125] but is somewhat problem dependent. In the results demonstrated below, the `libMesh` solver is typically configured to begin each nonlinear solve with a linear residual reduction tolerance of 10^{-3}, further decreasing this tolerance as the nonlinear residual is reduced. Using a smaller linear solver tolerance has the effect of leading to nonlinear solves which utilize fewer inexact Newton steps but more Krylov steps. Numerical experience with the FEM applications to follow suggests that this tradeoff often reduces the overall CPU time used. Particularly with higher order elements on unstructured grids, assembling a new Jacobian matrix and a corresponding preconditioner for a new inexact Newton step is much more expensive than evaluating a few sparse matrix-vector multiplies for a Krylov step.

To improve the reliability of the nonlinear solver, the solution to the linear solve in a Newton-Krylov step is not necessarily immediately used to update the nonlinear solution iterate for the next inexact Newton step. In all of the application problems tested in this thesis, simply using unmodified linear solver solutions for every step leads to solver divergence for some problem parameters. A reliable nonlinear solver must be able to recover from a diverging iteration, and the solver implemented for this work does so in several ways:

Although the Newton-Krylov method is guaranteed to give quadratic convergence near the exact zero of F, when $\vec{u}^{(k-1)}$ is too far from this root, an inexact Newton step may not even reduce the residual. To make convergence more likely without requiring more expensive linear solves, if $\left|\left|F^{(k)}\right|\right| \geq \left|\left|F^{(k-1)}\right|\right|$ then a line search in the direction of $\vec{u}^{(k)} - \vec{u}^{(k-1)}$ is used to find a point with reduced residual. With problems for which F is twice differentiable, a line search in the direction of an exact Newton step is guaranteed to find a reduced residual. A line search based on Brent's Method [22] for minimization has been developed and implemented in the present algorithm: First, step sizes are progressively halved, until some step length which produces a residual reduction is found. Finding such a reduction eventually would be inevitable if the linear system (4.4.2) were solved exactly, since Newton's Method provides a descent direction at any differentiable point. In practice, our numerical methods typically produce descent directions even for inexact linear solves. Once a step size z is found to provide a residual reduction, we have a bracket for the function $f(x) \equiv F(\vec{u}^{(k)} + x(\vec{u}^{(k+1)} - \vec{u}^{(k)}))$; i.e. we have a triplet $a < z < b$ of points for which $f(z) < f(a)$ and $f(z) < f(b)$. Given such a bracket, we search for a local minimizer between a and b by finding a sequence of smaller brackets, each of which obeys the same relations and is a strict subset of the previous bracket. This search involves two types of steps. First, we may fit a parabola to the three points of the current bracket, and find the minimum of the parabola, which is guaranteed to exist between a and b for any bracket. The x-coordinate w of the parabola's minimum is then used as a point in a new bracket which throws away either the upper or lower bound of the old bracket. For well-behaved functions, parabolic steps can converge very rapidly, but in the worst case a method relying solely on such steps may not converge at all. For reliability, Brent's Method adds the option to take steps based on golden-ratio sections. Of the two intervals (a, z) and (z, b) in the bracket, whichever is larger is divided in two by a point w in such a way that w is closer to z and the ratio of the two new segments adjoining w is the golden mean, $\frac{\sqrt{5}-1}{2}$. Using w and throwing away an extremum point to produce a new smaller bracket, the golden-ratio section ensures an asymptotic geometric decrease in bracket size, thus ensuring that to converge from our initial bracket of length 1 to a bracket smaller than a tolerance t, the computational cost is $\mathcal{O}\left(-\log(t)\right)$. The two types of steps are combined for the present algorithm in a process illustrated in the simplified flow chart in Figure 4.9. For full details on heuristics for step type selection and consideration of floating point error, see [104]. In the numerical experiments in this work, even tight line search tolerances can usually be met with less than a half dozen residual evaluations.

For more extreme nonlinearities, even this optimized substepping may not be sufficient to obtain convergence within the maximum number of nonlinear iterations allowed by the user. In such cases, rather than spend time on additional expensive Jacobian assemblies and linear solves, the nonlinear solver terminates with a failure code which instructs a higher level in the solve process to provide the solver with an easier problem. For a continuation-based solver, the nonlinear parameter in the continuation scheme is restored to its previous value and the parameter continuation increment is reduced, as described in Section 5.8. For an adaptive time stepping algorithm, the time step is attempted again with a reduced step size δt. In either case, the effect is to provide the nonlinear solver with a problem whose solution (and therefore region of convergence) is closer to the initial iterate used by the nonlinear solver's search. In addition to improving reliability by making it possible to recover from steps which the nonlinear solver cannot solve, this capability makes it

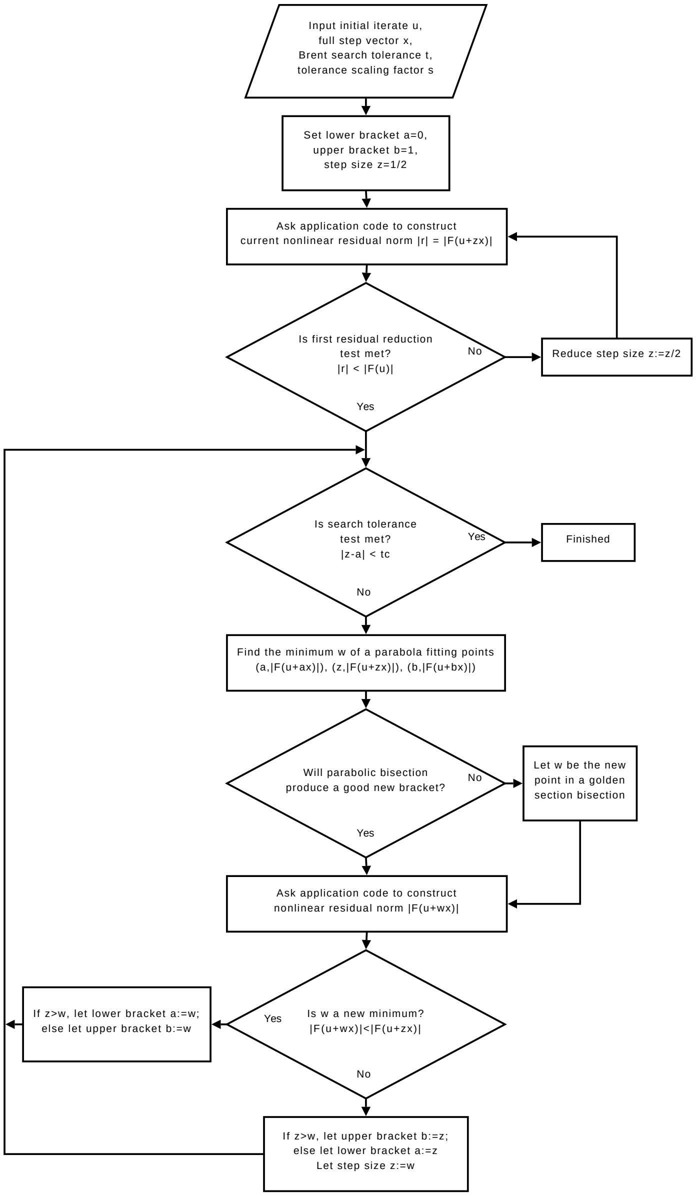
Input initial iterate u,
full step vector x,
Brent search tolerance t,
tolerance scaling factor s
Set lower bracket a=0,
upper bracket b=1,
step size z=1/2
Ask application code to construct
current nonlinear residual norm |r| = |F(u+zx)|
Is first residual reduction
test met?
|r| < |F(u)|
No
Reduce step size z:=z/2
Yes
Is search tolerance
test met?
|z-a| < tc
Yes
Finished
No
Find the minimum w of a parabola fitting points
(a,|F(u+ax)|), (z,|F(u+zx)|), (b,|F(u+bx)|)
Will parabolic bisection
produce a good new bracket?
No
Let w be the new
point in a golden
section bisection
Yes
Ask application code to construct
nonlinear residual norm |F(u+wx)|
If z>w, let lower bracket a:=w;
else let upper bracket b:=w
Yes
Is w a new minimum?
|F(u+wx)|<|F(u+zx)|
No
If z>w, let upper bracket b:=z;
else let lower bracket a:=z
Let step size z:=w

Figure 4.9: The basic steps in a Brent-based line search for optimal substep length.

possible to use a solver whose parameters are set to give up even on otherwise tractable problems. This strategy was found to improve overall computational efficiency with some of the nonlinear equations in the simulations to follow. In many cases, the solver was found to perform better when asked to solve multiple “easy” problems rather than a single equivalent “hard” problem. When using an adaptive continuation or time stepping algorithm, these reductions in step size are often temporary, and the algorithm can safely increase step size again once the confounding solution behavior is passed; some examples of this can be seen in the results in Section 7.14.

Chapter 5

Divergence-free Flow Problems

5.1 Introduction

Given the ability to construct finite element families supporting C^1 continuous basis functions, one can derive from these a set of locally-supported C^0 divergence-free vector-valued functions. The resulting function space spanned by this set is H^1 conforming, and each function $\vec{u}$ in the finite element space satisfies the divergence-free condition $\nabla \cdot \vec{u} = 0$ in a pointwise sense. Divergence-free finite element spaces are particularly useful in the study of incompressible fluid flow [15, 30] and electromagnetics [16, 131].

5.2 Divergence-free finite element spaces

Consider a scalar-valued finite element space H_f^h on domain Ω, with basis functions ϕ_i which are locally supported, include the space of polynomials $\mathcal{P}^k$ on each element, and are C^1 continuous with bounded first derivatives. The first two properties lead to the usual interpolation error results, and the third property gives $W^{2,p}$ conformity of the space.

First, use H_f^h to construct a streamfunction space H_s^h on Ω, whose bases will be functions $\vec{\phi}_i : \Omega \to \mathbb{R}^3$. If $\Omega \in \mathbb{R}^2$ for a planar flow problem, then $\dim(H_s^h) = \dim(H_f^h)$ and this construction is the trivial assignment $\vec{\phi}_i \equiv \phi_i \hat{e}_z$. If $\Omega \in \mathbb{R}^3$, then construct a space with $\dim(H_s^h) = 3\dim(H_f^h)$, and its basis will be $\phi_i \hat{e}_j$ for each j in $\{x, y, z\}$. Finally, if $\Omega \in \mathbb{R}^2$ is spanned by cylindrical unit vectors $\hat{e}_r$ and $\hat{e}_z$ for an axially symmetric flow problem, then construct a Stokes streamfunction space with $\dim(H_s^h) = \dim(H_f^h)$, and its basis will be $\vec{\phi}_i \equiv \phi_i \hat{e}_\theta$.

To obtain functions spanning a divergence-free function space H^h on our finite element mesh, take the curl of the streamfunction bases:

$$\vec{v}_i \equiv \nabla \times \vec{\phi}_i \tag{5.2.1}$$

The C^1 continuity of $\vec{\phi}_i$ ensures that $\vec{v}_i$ is well-defined and continuous everywhere. Note that $\hat{e}_z \perp \Omega$ for the planar flow case and $\hat{e}_\theta \perp \Omega$ for the cylindrical flow case. Thus, for $\Omega \in \mathbb{R}^2$, elements of $H^h(\Omega)$ will be vector-valued functions $\vec{v}_i : \Omega \rightarrow \mathbb{R}^2$.

The vector calculus identity $\nabla \cdot \nabla \times \vec{a} = 0$ ensures that $\vec{v}$ will be divergence-free at points where H_f^h is C^2 continuous. Because the components of H_f^h are piecewise smooth, this is almost everywhere in Ω. The derivative-based definition of divergence, $\nabla \cdot \vec{v} \equiv \sum \partial_i \vec{v}_i$, may be undefined at points of only C^1 continuity (e.g. along subelement boundaries within a macroelement), but the limit based definition of divergence, $\nabla \cdot \vec{v} \equiv \lim_{\text{diam}(V) \rightarrow 0} \int_V \vec{v} \cdot \vec{n} \, dA$, will still be zero.

Note that although these functions span a divergence-free function space, they are not a basis for that space. The kernel of the curl operator is non-empty, and so in general the set of functions in its image will not be linearly independent. To perform finite element calculations on a set of locally supported functions, these "spanning functions" can be used in the same way that basis functions are traditionally used in weighted residual methods; however the lack of linear independence raises difficulties when searching for a unique numerical solution.

5.3 Solution Existence and Uniqueness

The question of solution existence and uniqueness for wide classes of incompressible flow problems is beyond the scope of this work. In fact, existence and smoothness of solutions to the 3D incompressible Navier-Stokes problem is one of the Clay Mathematics Institute "Millenium Problems" [88]. We can, however, make use of existing results by posing a question more specific to the divergence-free finite element method: if there exists a divergence-free weak solution $\vec{u}$ to a boundary value problem, under what conditions does there also exist a streamfunction $\vec{\phi}$ which generates that solution? What additional constraints should be applied to the problem to make $\vec{\phi}$ unique?

In order to find convergent solutions to boundary problems posed on divergence-free function spaces, it is not sufficient to be able to construct a set of divergence-free functions on the function domain; it is also necessary to ensure that the desired solution is the limit of a sequence of constructable functions. It is true that the image of the curl operator will be a set of divergence free functions, but an existence proof requires specifically that the particular function we seek is in the image of the curl operator. In the terminology of differential forms, every exact differential form is closed, but is every closed 2-form in the space of admissible solutions exact?

For a wide range of domains and boundary conditions, this is the case. If a domain is contractible (homeomorphic to a point), then any smooth divergence-free vector field on that domain is the curl of another vector field (see, e.g. [136]). What does this mean for typical incompressible fluid flow problems, which are often posed on non-contractible domains? If the fluid domain is not contractible because of incompressible solid objects inside the domain, then one can use the object velocities to extend the fluid velocity function to the interior of the objects, use the contractability of the extended domain to prove the existence of a streamfunction for the fluid/solid velocity field, and finally restrict this streamfunction back to the original domain where it is a streamfunction for the fluid velocity field alone.

It is tempting to say that in this fashion one can find a streamfunction for any incompressible flow problem, but counterexamples exist. If a flow is incompressible in a non-contractible domain but is compressible in a volume enclosed by that domain, then it is possible that no streamfunction for the flow exists. A simple example is planar radial flow through an annulus: the velocity field $\vec{u} \equiv \hat{e}_r/r$ is divergence free in the region $1 < r < 2$, but integrating the tangential streamfunction derivative $\frac{\partial\psi}{\partial\vec{t}} = \vec{u}\cdot\vec{n} < 0$ around the $r = 1$ boundary does not give a single-valued function. On such problems, finding a divergence-free flow solution from a streamfunction formulation is more complex; see e.g. [4].

Given a nonlinear functional $J(\vec{v})$ on a space H^h of divergence-free functions for which streamfunctions exist, define the functional $J_s \equiv J \circ (\nabla\times)$ whose range is the range of J but whose domain is the streamfunction space H^h_s. Because $(\nabla\times)$ is a linear operator, whose kernel we will call K, every minimizer $\vec{u}$ satisfying $J(\vec{u}) \leq J(\vec{v})\ \forall\vec{v}$ corresponds to an affine space of streamfunctions $\Psi \equiv \left\{\vec{\psi} : \nabla \times \vec{\psi} = \vec{u}\right\}$, where $\Psi = \vec{\psi} + K$ for any $\vec{\psi} \in \Psi$.

In two dimensions, $\dim(K) = 1$, and one can easily find a unique $\vec{\psi}$ by adding a penalty term which restricts the value of $\vec{\psi}(x)$ to be zero at some point $x \in \Omega$. Using streamfunction boundary conditions instead of velocity boundary conditions accomplishes this restriction automatically, and in practice many combinations of iterative solvers and preconditioners appear to be stable even if the systems they solve are this slightly underdefined.

In three dimensions, however $\dim(K)$ is infinite, and in numerical experiments $\dim(K \cap H^h_s)$ is large enough to make some iterative solvers and preconditioners fail. This kernel is difficult to constrain away analytically without ruining the sparsity structure inherent in finite element system matrices. For special classes of elements, topological arguments may allow the system to be reduced [112] to a subspace on which the linear operator is positive definite, but in general a linear solver which deals robustly with indefinite systems will be most useful. Initial 3D results have been obtained with `libMesh` using Jacobi preconditioning with a GMRES solver.

5.4 Incompressible Navier-Stokes Flow

For a constant density fluid, the incompressible Navier-Stokes equations can be expressed in dimensionless form as:

$$\begin{aligned} \nabla \cdot \vec{u} &= 0 && (5.4.1) \\ \frac{\partial \vec{u}}{\partial t} + \mathrm{Re}(\vec{u} \cdot \nabla)\vec{u} &= -\nabla P + \nabla \cdot \left(\frac{\nu}{2} \left(\nabla \vec{u} + (\nabla \vec{u})^T \right) \right) && (5.4.2) \end{aligned}$$

In a standard velocity-pressure mixed formulation [33,70,108], the corresponding variational saddle point problem is discretized on a compatible pair of velocity and pressure spaces, then velocity test functions are used to enforce the momentum equation weakly while pressure test functions enforce the incompressibility condition weakly, e.g.

$$\begin{aligned} (\nabla \cdot \vec{u}, q)_\Omega &= 0 \quad \forall q \in H_p^h && (5.4.3) \\ \left(\frac{\partial \vec{u}}{\partial t} + \mathrm{Re}(\vec{u} \cdot \nabla)\vec{u}, \vec{v} \right)_\Omega &= (P, \nabla \cdot \vec{v})_\Omega - \left(\frac{\nu}{2} \left(\nabla \vec{u} + (\nabla \vec{u})^T \right), \vec{v} \right)_\Omega && (5.4.4) \end{aligned}$$

Pressure values must be solved for in the above system, and the velocity solutions obtained are not pointwise divergence-free.

Instead, we can use the divergence-free space to satisfy the incompressibility condition exactly, while at the same time eliminating the pressure variable from the weak form of the momentum equation via the following identity:

$$\int_\Omega \nabla P \cdot \vec{v} \, d\Omega = \int_{\partial\Omega} P \vec{v} \cdot \vec{n} \, dS - \int_\Omega P \nabla \cdot \vec{v} \, d\Omega \qquad (5.4.5)$$

When using a divergence-free velocity space, the variational test functions $\vec{v}$ will also be divergence free, and the last term will be zero. Pressure may still appear in a boundary term, if it is specified as boundary data in a pressure gradient driven flow. However, on boundaries where normal velocity $\vec{u} \cdot \vec{n}$ is specified, variation in normal velocity $\vec{v} \cdot \vec{n}$ will be zero. For problems with Dirichlet velocity boundary conditions, pressure will be completely removed from the divergence-free formulation:

$$\begin{aligned}\int_\Omega \frac{\partial \vec{u}}{\partial t} \vec{v}\, d\Omega &= -\mathrm{Re} \int_\Omega \vec{v}(\vec{u} \cdot \nabla)\vec{u}\, d\Omega - \int_\Omega \nu(\nabla \vec{u} + (\nabla \vec{u})^T) : \nabla \vec{v}\, d\Omega \\ &\quad + \int_{\partial\Omega} \nu(\nabla \vec{u} + (\nabla \vec{u})^T)\vec{v} \cdot \vec{n}\, dS \qquad (5.4.6)\end{aligned}$$

Other boundary conditions exist where pressure terms remain in boundary integrals, but in these cases the pressure variable will be problem data, not an unknown to be solved for.

5.5 Shear-dependent Viscosity Models

In the above formulations, we refer to the viscosity ν, which in Newtonian fluid flows is a constant dependent only on fluid composition. Many Non-Newtonian fluids, however, can be described by the same equations by simply modifying ν to be a fluid-specific function of physical conditions like temperature, solute concentration, or shear deformation [12–14, 46, 47].

For the shear-thinning fluids we consider, flow behavior can be described by a shear-dependent viscosity model, in which ν is solely a function of the shear rate $s \equiv 2\sqrt{D(\vec{u}) : D(\vec{u})}$, where to simplify notation we define the rate of deformation tensor of a velocity field to be the symmetric part of the gradient of velocity:

$$D(\vec{u}) \equiv \frac{1}{2}\left(\nabla \vec{u} + (\nabla \vec{u})^T\right) \qquad (5.5.1)$$

It is convenient to reformulate the entire viscosity term in terms of the rate of deformation tensor $D(\vec{u})$ rather than the asymmetric velocity gradient $\nabla\vec{u}$:

$$\begin{aligned}\int_\Omega \frac{\partial\vec{u}}{\partial t}\vec{v}\,d\Omega &= -\mathrm{Re}\int_\Omega \vec{v}(\vec{u}\cdot\nabla)\vec{u}\,d\Omega - \int_\Omega 2\nu(s)D(\vec{u}) : D(\vec{v})\,d\Omega \\ &\quad + \int_{\partial\Omega} 2\nu(s)D(\vec{u})\vec{v}\cdot\vec{n}\,dS \qquad (5.5.2)\end{aligned}$$

The identity $(A+A^T) : A = \frac{1}{2}(A+A^T) : (A+A^T)$ makes this equivalent.

Some examples of shear-dependent fluid viscosity models follow.

For Carreau fluids:

$$\begin{aligned}\gamma > 0 \qquad & r \geq 1 \\ \nu(s) \;=\; & \nu_\infty + \frac{\nu_0 - \nu_\infty}{(1+(\gamma s)^2)^{(2-r)/2}} \qquad (5.5.3)\end{aligned}$$

For Extended Williamson fluids (including classical Williamson fluids, for which $r = 1$):

$$\begin{aligned}\gamma > 0 \qquad & 1 \leq r \leq 2 \\ \nu(s) \;=\; & \nu_\infty + \frac{\nu_0 - \nu_\infty}{1+(\gamma s)^{2-r}} \qquad (5.5.4)\end{aligned}$$

For Oldroyd fluids:

$$\begin{aligned}\gamma_1 > 0 \qquad & \gamma_2 > 0 \\ \nu(s) \;=\; & \nu_0 \frac{1+(\gamma_1 s)^2}{1+(\gamma_2 s)^2} \qquad (5.5.5)\end{aligned}$$

For Powell-Eyring fluids:

$$\gamma > 0$$

$$\nu(s) = \nu_\infty + (\nu_0 - \nu_\infty)\frac{\sinh^{-1}(\gamma s)}{\gamma s} \qquad (5.5.6)$$

5.6 Penalty Boundary Conditions

We apply Dirichlet boundary conditions weakly by adding an additional weighted residual term to the finite element equations, penalizing solutions which differ from the desired Dirichlet boundary value. Using a small penalty coefficient ϵ gives solutions with similarly small error on Dirichlet boundaries [34, 35].

Starting from a discrete nonlinear functional F expressing the boundary value problem as $F_i(\vec{u}) = 0$, enforce $\vec{u} = \vec{u}_D$ on the boundary by adding the penalty term

$$G_i^\epsilon(\vec{u}) \equiv \frac{1}{\epsilon} \int_{\partial\Omega} (\vec{u} - \vec{u}_D) \cdot \vec{v}_i \, dS \tag{5.6.1}$$

For consistency in Newton and inexact Newton methods, the new nonlinear residual $F_i + G_i$ must be differentiated instead of F_i to provide the system Jacobian. The derivative of G_i is symmetric positive semidefinite:

$$\frac{\partial G_i^\epsilon}{\partial u_j}(\vec{u}) = \frac{1}{\epsilon} \int_{\partial\Omega} \vec{v}_j \cdot \vec{v}_i \, dS \tag{5.6.2}$$

And thus symmetry and positive definiteness of the problem being solved are unaffected by the use of penalty boundary conditions.

5.7 Successive Approximation

There are two sources of nonlinearity in flow systems modeled with non-Newtonian viscosity. The Reynolds number, $\mathrm{Re} \equiv VL/\nu_0$, is based on the characteristic velocity, the domain length scale, and the kinematic viscosity. For shear-thinning fluids, define Re based on the limiting viscosity ν_0 at low flow rates. The Reynolds number adds a nonlinear convection term to the momentum equation for Re > 0. The viscosity variation, ν_∞/ν_0, is based on kinematic viscosity ν_∞ at high flow rates. For non-Newtonian fluids, $\nu_\infty \neq \nu_0$ and the viscosity variation adds nonlinearities to the momentum equation's velocity diffusion term.

Rapidly obtaining highly accurate solutions to these equations requires a quadratically converging technique like Newton or (with appropriate linear tolerances) Newton-Krylov. A slower convergence rate but a greater region of convergence can often be obtained by simply lagging the nonlinear terms. For the stationary problem, this gives

$$\begin{aligned}\mathrm{Re} \int_\Omega \vec{v} \cdot (\vec{u}^{(k-1)} \cdot \nabla)\vec{u}^{(k)} \, d\Omega + \int_\Omega 2\nu(s(D(\vec{u}^{(k-1)})))D(\vec{u}^{(k)}) : D(\vec{v}) \, d\Omega \\ - \int_{\partial\Omega} 2\nu(s(D(\vec{u}^{(k-1)})))D(\vec{u}^{(k)})\vec{v} = 0 \quad (5.7.1)\end{aligned}$$

As proven in [30], successive approximation of Stokes flow (Re $= 0$) converges for any shear-thinning fluid with ν_0 finite and ν_∞ bounded above zero from below.

Experimentally, in our current studies for the present flow classes, successive approximation converges when given a sufficiently accurate starting iterate (e.g. an approximate solution at a slightly lower Reynolds number), but diverges from inaccurate starting iterates (e.g. zero flow or approximate solutions at much lower Reynolds numbers).

5.8 Continuation

Lower ν_∞/ν_0 ratios appear to shrink the region of convergence, requiring increasingly accurate initial guesses. Such initial conditions can be obtained reliably through a continuation algorithm, which finds a solution at Reynolds number $\mathrm{Re}_{\max}$ as follows:

1. Choose an initial Reynolds number $\mathrm{Re}_0 = 0$, an initial increment $\delta\mathrm{Re}_0 = 100$, and an initial iterate $\vec{u}_0 = \vec{0}$.
2. Loop, beginning at $i = 0$:
3. Attempt to solve the modified flow problem at Re_i, using $\vec{u}_i$ as a starting iterate.
4. If the nonlinear solver converged and $\mathrm{Re}_i = \mathrm{Re}_{\max}$, we have the desired solution and the algorithm is finished.
5. If the nonlinear solver converged but $\mathrm{Re}_i < \mathrm{Re}_{\max}$, let Re_{i+1} be $\mathrm{Re}_i + \delta\mathrm{Re}_i$, set $\vec{u}_{i+1}$ to the new solution, and let $\delta\mathrm{Re}_{i+1}$ be $2\delta\mathrm{Re}_i$.
6. If the nonlinear solver failed, let $\delta\mathrm{Re}_{i+1}$ be $0.5\delta\mathrm{Re}_i$ and let Re_{i+1} be $\mathrm{Re}_i - 0.5\delta\mathrm{Re}_i$
7. If $\mathrm{Re}_{i+1} > \mathrm{Re}_{\max}$, then let $\delta\mathrm{Re}_{i+1}$ be $\mathrm{Re}_{\max} - \mathrm{Re}_i$ and set $\mathrm{Re}_{i+1} = \mathrm{Re}_{\max}$.
8. Repeat the loop for the next i.

This algorithm may handle bifurcations poorly, and is nearly certain to fail at turning points, but neither were encountered in the moderate nonlinearities tested in the non-Newtonian flow experiments herein. For more highly nonlinear applications, the above algorithm can be extended to more robust arclength or pseudo-arclength forms.

In practice, only the solutions attempted at $\mathrm{Re}_{\max}$ need to be evaluated to a tight error tolerance; even an inexact solution at a nearby Reynolds number is usually sufficient to give a good starting iterate for successive approximation at a higher Reynolds number.

5.9 Newton-Krylov Methods

Because the method of successive approximation converges linearly even near its limit point, it may take many iterations to find a PDE solution accurate to several sigificant digits. Instead, when the constitutive relation leads to an invertible Jacobian $\frac{\partial F_i}{\partial \vec{u}_j}$, Newton-Krylov methods can be used to obtain quadratic convergence in the nonlinear solver.

Expressing the velocity approximation as a linear sum of basis functions

$$\vec{u_h} = \sum_{i=0}^{N} u_i \vec{v}_i \tag{5.9.1}$$

and expressing the weighted residuals as a vector element

$$F_j = \mathrm{Re} \int_\Omega \vec{v}_j \cdot (\vec{u} \cdot \nabla)\vec{u}\, d\Omega + \int_\Omega 2\nu(s(D(\vec{u})))D(\vec{u}) : D(\vec{v}_j)\, d\Omega \tag{5.9.2}$$

the Jacobian follows formally as

$$\begin{aligned}
\frac{\partial F_j}{\partial u_i}(\vec{u}) &= \mathrm{Re} \int_\Omega \vec{v}_j \cdot [(\vec{u} \cdot \nabla)\vec{v}_i + (\vec{v}_i \cdot \nabla)\vec{u}]\, d\Omega \\
&\quad + \int_\Omega 2\nu(s(D(\vec{u})))D(\vec{v}_i) : D(\vec{v}_j)\, d\Omega \\
&\quad + \int_\Omega 4\frac{\partial \nu}{\partial s}(s(D(\vec{u})))\frac{1}{s(D(\vec{u}))}(D(\vec{u}) : D(\vec{v}_i))(D(\vec{u}) : D(\vec{v}_j))\, d\Omega
\end{aligned} \tag{5.9.3}$$

The full inexact Newton step to $\vec{u}^{(k)}$ is found by using a Krylov subspace solver to find an approximate solution to the linearized problem at $\vec{u}^{(k-1)}$:

$$\frac{\partial F_j}{\partial u_i}\left(\vec{u}^{(k-1)}\right) \cdot \left(u_i^{(k-1)} - u_i^{(k)}\right) \quad = \quad F_j\left(\vec{u}^{(k-1)}\right) \tag{5.9.4}$$

In practice, if $\vec{u}^{(k-1)}$ is too far from the exact zero of F, this step may not reduce the residual. To make convergence more likely without requiring more expensive linear solves, if $F^{(k)} \cdot F^{(k)} \geq F^{(k-1)} \cdot F^{(k-1)}$ then a line search in the direction of $u^{(k)} - u^{(k-1)}$ is used to find a point with reduced residual, as described in Section 4.4. With viscosity models for which F is twice differentiable, a line search based on an exact linear solver is guaranteed to find a reduced residual, and near the algebraic exact solution u the method is guaranteed to converge quadratically. In our current results, the Newton's Method solver converges much faster than successive approximation for Powell-Eyring fluids, but (in the exact formulation above) it fails to converge for simulations with Williamson fluids.

For Williamson fluids, the appearance of $\frac{\partial \nu}{\partial s}(s)\frac{1}{s}$ in the Jacobian is a source of difficulty: this ratio of viscosity derivative to strain is unbounded as fluid strain goes to zero, and so the system residual F is not as smoothly differentiable as it first appeared. Altering the Jacobian (by putting a cap on the unbounded term, for example) does produce a system which converges to the correct solution, but the convergence is now linear and has no obvious benefits over successive approximation.

Even exact Newton's method is not guaranteed to converge from starting iterates too far from the exact solution of the nonlinear problem. To ensure the Newton solver's convergence in Navier-Stokes problems, we employ the same Reynold's number continuation algorithm which was used with successive approximation, finding approximate algebraic solutions at gradually increasing Reynolds numbers to stay within the Newton region of convergence.

Again, only the solutions attempted at $\mathrm{Re}_{\max}$ need to be evaluated to full tolerance; one or two Newton steps at lower Reynolds numbers is sufficient to give a good starting iterate for the higher Reynolds number problems. A useful strategy may be to begin with successive approximation for robustness at low Reynolds numbers, then switch to a Newton-Krylov solver to obtain the final high accuracy solution more efficiently.

5.10 Time-dependent flow

To solve the time-dependent flow system via a semidiscrete formulation, finite differencing the time derivative term gives a spatial boundary value problem at each timestep. Standard "theta methods" then advance a known solution $\vec{u}^{(n)}$ through a timestep to the next solution $\vec{u}^{(n+1)}$.

By using a first order finite difference evaluation of $\frac{\partial \vec{u}}{\partial t}$, and evaluating the time derivative at an arbitrary fraction θ of the step between t^n and t^{n+1}, we can write down a single theta method which includes Forward Euler ($\theta = 0$), Backward Euler ($\theta = 1$), and Crank-Nicolson ($\theta = \frac{1}{2}$). Defining a new unknown $\delta\vec{u} \equiv \vec{u}^{(n+1)} - \vec{u}^{(n)}$ gives:

$$\begin{aligned}\int_\Omega \frac{\delta\vec{u}}{\delta t}\vec{v}\,d\Omega \;=\; & -\mathrm{Re}\int_\Omega \vec{v}\cdot((\vec{u}^{(n)}+\theta\delta\vec{u})\cdot\nabla)(\vec{u}^{(n)}+\theta\delta\vec{u})\,d\Omega \\ & -\int_\Omega 2\nu(s(\vec{u}^{(n)}+\theta\delta\vec{u}))D(\vec{u}^{(n)}+\theta\delta\vec{u}):D(\vec{v})\,d\Omega \qquad (5.10.1) \\ & +\int_{\partial\Omega} 2\nu(s(\vec{u}^{(n)}+\theta\delta\vec{u}))D(\vec{u}^{(n)}+\theta\delta\vec{u})\vec{v}\cdot\vec{n}\,dS\end{aligned}$$

Forward Euler With $\theta = 0$, the only term to involve the unknown $\delta\vec{u}$ is the mass matrix. Discretizing this equation with divergence-free elements gives a mass matrix which in 2D resembles a Laplacian matrix on the streamfunction, since

$$\int_\Omega \delta\vec{u} \cdot \vec{v}\, d\Omega \;\equiv\; \int_\Omega \nabla\delta\psi \cdot \nabla\phi\, d\Omega \tag{5.10.2}$$

To make this matrix non-singular, some boundary condition must be added. At the least, in two dimensions one streamfunction node must be "pinned" to constrain the constant non-zero streamfunction solution mode and force $\delta\vec{u}$ to be unique.

One advantage of Forward Euler is that the mass matrix is time-constant, and so it can be factored (or an expensive preconditioner constructed) only at the beginning of a simulation and after any remeshing, then those factorizations or preconditioners can be reused repeatedly. The nonlinear terms are evaluated explicitly, so no nonlinear solution method is required.

The disadvantages of Forward Euler solves are quickly made evident by experimentation. Even for Stokes flow with Newtonian fluids, stability depends strongly on δt. Unless $\delta t < \mathcal{O}\left(h^2\right)$, high frequency errors are amplified and the solution diverges. For Navier-Stokes flow or for flow of non-Newtonian fluids, solution stability also depends on how strong the equation nonlinearities are. At even moderate Reynolds numbers ($Re = 200$), Forward Euler streamfunction solutions appear to diverge on any mesh regardless of time step size.

Crank-Nicolson With $\theta = \frac{1}{2}$, the timestepping scheme is $O(\delta t^2)$ accurate, but requires a nonlinear solution for $\delta\vec{u}$ at each timestep. The nonlinear system resembles the nonlinear steady state problem, but because the steady state nonlinear functional F is weighted and added to the mass matrix M, the Jacobian and residual of the system as a whole are adjusted accordingly:

$$\begin{aligned} M\delta\vec{u} - \delta t F(\vec{u}^{(n)} + \theta\delta\vec{u}) &= 0 \qquad (5.10.3) \\ \left(M - \theta\delta t \frac{\partial F}{\partial \vec{u}}\left(\vec{u}^{(n)}\right)\right)\left(\delta\vec{u}^{(k)} - \delta\vec{u}^{(k+1)}\right) &= M\delta\vec{u}^{(k)} - \delta t F(\vec{u}^{(n)} + \theta\delta\vec{u}^{(k)}) \end{aligned}$$

In practice, the advantages of Crank-Nicolson can be obtained without the expense of a fully convergent Newton solve by using an incomplete Newton method: a single Newton step from an initial guess of $\delta\vec{u} = 0$ appears to be sufficient to give quadratic time accuracy and unconditional stability. Additional accuracy could be achieved using a predictor-corrector method: the result of a cheap Forward Euler step can serve as the initial guess in the implicit solve, simultaneously reducing the error in an incomplete Newton method with a fixed number of steps or reducing the number of steps required for a Newton solve with a fixed nonlinear tolerance.

5.11 Unsteady Flow and Transport

Among the motivations for finding strongly divergence-free velocity fields is their suitability as advection terms in transport equations. Let us consider the scalar single-component transport equation,

$$\frac{\partial c}{\partial t} + \vec{u} \cdot \nabla c = \nabla \cdot (D \nabla c) \qquad (5.11.1)$$

Taking weighted residuals with a test function v and integrating by parts gives the weak form of the equation,

$$\int_\Omega \frac{\partial c}{\partial t} v \, d\Omega = \int_\Omega D\Delta c v \, d\Omega - \int_\Omega \vec{u} \cdot \nabla c \, v \, d\Omega \tag{5.11.2}$$

$$\tag{5.11.3}$$

or

$$\left(\frac{\partial c}{\partial t}, v \right)_\Omega = -(D\nabla c, \nabla v)_\Omega + (D\nabla c \cdot \vec{n}, v)_{\partial\Omega} - (\vec{u} \cdot \nabla c, v)_\Omega \tag{5.11.4}$$

For problems with Neumann-only boundary condtions, the constant function $v = 1$ is contained in the space of test functions. With this v,

$$\begin{aligned} \frac{\partial}{\partial t} \int_\Omega c \, d\Omega &= \int_{\partial\Omega} D\nabla c \cdot \vec{n} \, dS - \int_\Omega \vec{u} \cdot \nabla c \, d\Omega && (5.11.5) \\ &= \int_{\partial\Omega} D\nabla c \cdot \vec{n} \, dS + \int_\Omega c\nabla \cdot \vec{u} \, d\Omega - \int_{\partial\Omega} c\vec{u} \cdot \vec{n} \, dS && (5.11.6) \end{aligned}$$

In the exact solution to an incompressible flow and transport problem, $\nabla \cdot \vec{u} = 0$, and so the rate of change of global concentration $\frac{\partial}{\partial t} \int_\Omega c \, d\Omega$ will be exactly balanced by the two boundary terms, which correspond to diffusive and convective inflow respectively. For this global conservation to hold true in a numerical approximation, it is sufficient that the advection velocity field $\vec{u}$ satisfies $(c, \nabla \cdot \vec{u})_\Omega = 0$ for any value c in the concentration function space.

5.12 Divergence-free Flow Example Results

As an interesting divergence-free flow test, we simulate shear-thinning Navier-Stokes flow in the 2D lid-driven cavity. Incompressible flow is modeled in a square domain, enforcing zero velocity on three sides and a constant lateral velocity along the top. In Figure 5.2 and Figure 5.3 we see typical results for Newtonian flows, at Reynolds numbers of 400 and 1000 respectively, computed on uniform 32x32 meshes of squares divided into HCT 3-split triangles. The limitations of the 2048 element uniform meshes become apparent in the vorticity plots, in which the solution and postprocessing projection have led to distinct numerical oscillations.

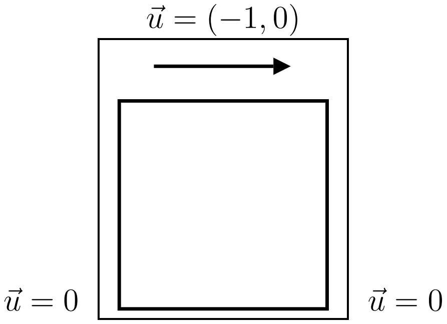

Figure 5.1: Domain and boundary conditions for lid-driven cavity flow

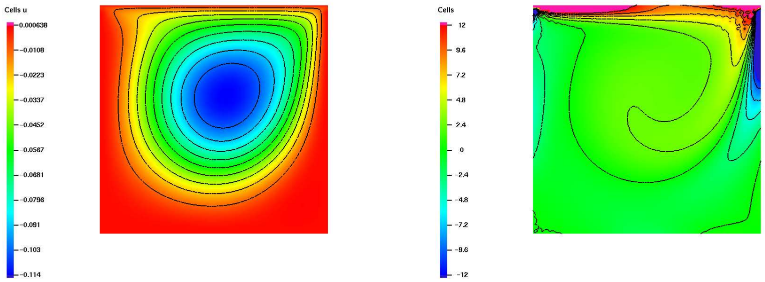

Figure 5.2: Steady-state streamfunction and vorticity plots for Newtonian lid-driven cavity flow at Reynolds number 400

Because the velocity boundary conditions are discontinuous at the corners, the first derivative of the streamfunction is also discontinuous and its

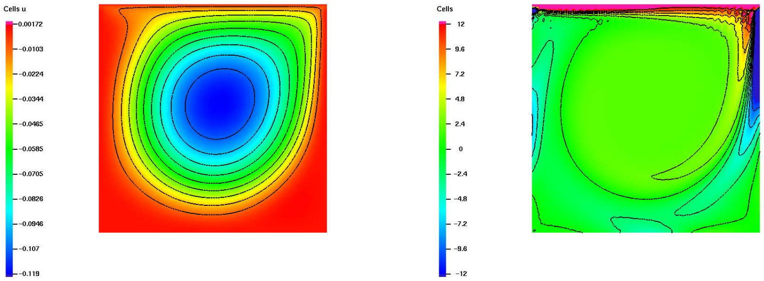

Figure 5.3: Steady-state streamfunction and vorticity plots for Newtonian lid-driven cavity flow at Reynolds number 1000

second derivatives are singular. As $r \to 0$,

$$\psi(r,\theta) \sim \frac{r}{\pi^2-4}\left(-\pi^2\sin(\theta)+2\pi\theta\sin(\theta)+4\theta\cos(\theta)\right) \quad (5.12.1)$$

$$\omega(r,\theta) \sim \frac{1}{(\pi^2-4)r}\left(4\pi\cos(\theta)-8\sin(\theta)\right) \quad (5.12.2)$$

The error encountered in trying to resolve this singularity on a coarse mesh can lead to noticeable pollution well into the interior of the domain. Over-refinement can limit the problem, but we can find such irregular solutions more efficiently by using adaptive h refinement to grade the mesh into the points of reduced regularity. Using the Laplacian jump error estimator on the streamfunction, adaptive refinement focuses in on the singularities but still tracks the trail of vorticity that has been convected into the interior. A vorticity solution on an example adapted mesh is shown in Figure 5.4.

The result of this process is that the sequence of adapted meshes recovers some of the convergence rate which has been lost to the uniform refinement sequence, as shown in Figure 5.5.

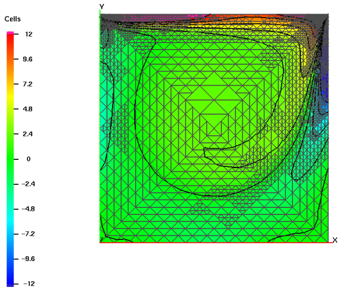

Figure 5.4: Steady-state vorticity plot for Extended Williamson lid-driven cavity flow at Reynolds number 500 and viscosity ratio 0.1.

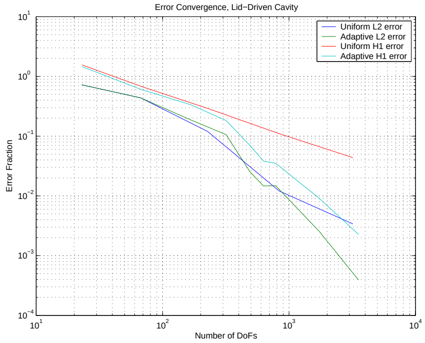

Figure 5.5: Error convergence, plotted as L_2 and H^1 error norms against number of unconstrained degrees of freedom, for both uniformly and adaptively refined meshes.

5.13 Transport Results

Using the lid-driven cavity flow results as an advection velocity, we can demonstrate the mass conservation properties for a decoupled species transport problem.

This system corresponds to flow of a Newtonian fluid at Reynolds number $\Re = 500$, and transport of a species with diffusivity $D = .01$ Backward Euler integration is used with timesteps of fixed length $\delta t = 10^{-3}$. The initial concentration varies linearly from 0 (plotted as blue) on the left of the domain to 1 (plotted as red) on the right.

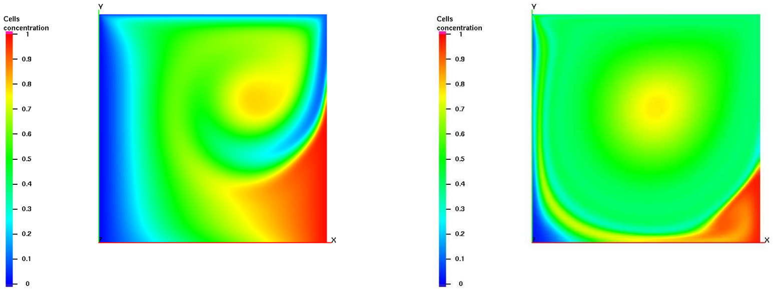

Figure 5.6: Concentration plots for Newtonian lid-driven cavity flow at Reynolds number 500, at nondimensionalized times $t = 5$ and $t = 25$

Chapter 6

Surface Tension Driven Flow of Thin Films

6.1 Introduction

Consider the flow of a layer of liquid, constrained below by a heated flat solid surface and bounded above by gas underneath a cooled flat plate. In the general case, flow in the liquid region interior is modeled with the 3-D Navier-Stokes equations for velocity and pressure, coupled to an advection-diffusion equation for temperature, with appropriate boundary conditions. For thin liquid layers, however, the three dimensional problem begins to exhibit essentially two-dimensional behavior, as the potential energy associated with surface tension begins to exceed the gravitational, inertial, and thermal energy of the fluid interior. For extremely thin fluid layers, surface tension effects dominate the flow, and a fourth-order two dimensional PDE can correctly model those effects. The goal of this part of the research is to develop new formulations for exploring the behavior of this reduced fourth-order model.

Although there is a stationary flow solution to this heat transfer problem, the stability of that solution depends on the parameters of the particular system.

The most familiar instability in fluids heated from below is buoyancy driven convection in the fluid. The effect of buoyancy depends on the nondimensional Raleigh number, $\mathrm{Ra} \equiv \alpha_T g \delta T d^3/\nu k$, which depends on the fluid thermal expansion coefficient α_T, dynamic viscosity ν, thermal conductivity k, and average depth d, as well as gravity g and the applied temperature difference $\delta T \equiv T_{\text{bottom}} - T_{\text{top}}$. Above a critical Raleigh number, warmer fluid at the bottom of a convection cell expands and becomes less dense, is buoyed up, cools at the fluid surface away from the heated plate, then sinks again at the sides of the cell. The strength of this effect decreases as the cube of d, and so it is less important than surface tension effects in sufficiently thin layers.

For thinner liquid layers, convection cells are still observed to arise from thermocapillary effects on the surface [111]. The development of these Bénard-Marangoni cells depends on the Marangoni number, $\mathrm{M} \equiv \sigma_T \delta T d / \rho \nu k$, which nondimensionalizes the derivative of surface tension with respect to temprature, σ_T, in terms of parameters described above as well as on liquid density ρ. Above a critical Marangoni number, reduced temperature at the surface leads to increased surface tension σ in cool spots, which draw liquid horizontally. The displaced liquid is forced down to the heated plate and drawn up again in hot spots. The Marangoni number also decreases with decreasing d, but only linearly.

In extremely thin heated liquid layers, the primary instability observed is a deformation in the liquid surface, also driven by thermocapillary effects but with a much longer wavelength [128]. Onset of this instability is determined primarily by the inverse dynamic Bond number, $D \equiv \sigma_T \delta T / \rho g d^2$. Cool spots on the surface draw in liquid horizontally, and the liquid layer thickness increases. The increase in surface height causes the surface to be more insulated from the heated plate below and less insulated from the cool plate above, cooling the liquid. and adding positive feedback to the system. For sufficiently large D, this destabilizing effect can exceed the stabilizing effect of gravity, causing long-wavelength perturbations in the liquid height to grow.

It is this final flow regime that can be modeled by the fourth-order thermocapillary flow equation. We will demonstrate the use of C^1 finite elements to construct conforming solutions to the Galerkin approximation of this model.

It is possible to add additional physical stability to the system by introducing a surfactant layer to the liquid surface. Areas of increased surface tension draw in surrounding fluid horizontally, and the incoming convection of surfactant increases the local surfactant concentration, lowering the surface tension back toward equilibrium. For related experiments see [98]. The surfactant transport equation can be simulated fully coupled with the flow equation with a similar fourth-order model.

6.2 Flow and transport equations

With slight modification, we follow the derivation used in [133] and [129]. The derivation of the simplified fourth-order model begins from the incompressible Navier-Stokes equations for Newtonian fluid flow, coupled to transport equations for temperature and surfactant. The flow velocity $\vec{u}$, pressure P, and temperature T evolve in time; we normalize these variables based on the average liquid thickness d, density ρ, thermal diffusivity $\kappa \equiv \frac{k}{\rho c_p}$, and viscosity ν as follows: length is scaled by layer depth d, time by d^2/κ, velocity by κ/d, pressure by $\rho\nu\kappa/d^2$, and temperature by δT. The resulting equations are

$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla) \vec{u} = \Pr(-\nabla P + \Delta \vec{u} - \mathrm{G} \hat{e}_z) \tag{6.2.1}$$

$$\nabla \cdot \vec{u} = 0 \tag{6.2.2}$$

$$\frac{\partial T}{\partial t} + (\vec{u} \cdot \nabla) T = \Delta T \tag{6.2.3}$$

where $\Pr \equiv \nu/\kappa$ is the Prandtl number, and $\mathrm{G} \equiv \frac{gd^3}{\nu\kappa}$ is the Galileo number.

The non-dimensionalized height u of the liquid layer evolves with the flow. Because u is a function only of x and y, its gradient is a vector in the $x - y$ plane. Defining the two dimensional gradient $\nabla_\perp \equiv \hat{e}_x \frac{\partial}{\partial x} + \hat{e}_y \frac{\partial}{\partial y}$, we can say that $\nabla_\perp u = \nabla u$. Defining the vertical velocity component $w \equiv \vec{u} \cdot \hat{e}_z$,

$$\frac{\partial u}{\partial t} + \vec{u} \cdot \nabla u = w \tag{6.2.4}$$

At the bottom flat plate, the temperature is held constant (and set here to 0 in the scaled system for convenience), and the no-slip, no-penetration velocity conditions apply. That is,

$$T(z = 0) = 0 \tag{6.2.5}$$

$$\vec{u}(z = 0) = \vec{0} \tag{6.2.6}$$

Define the standard nondimensional Biot number as $\mathrm{H} \equiv \frac{k_g d}{k d_g}$, based on the average thicknesses d, d_g and thermal conductivities k, k_g in the liquid and gas layers, the temperature on the liquid surface can be expressed as

$$T(z = 1 + d_g/d) = -\frac{1 + \mathrm{H}}{\mathrm{H}} \tag{6.2.7}$$

A surface tension boundary condition also applies on the liquid surface. Let the subscripts s and n denote directions of vectors within and perpendicular to the surface plane respectively, and let R_1 and R_1 be the local radii of curvature of the surface. Defining a dimensionless surface tension $S \equiv \frac{\sigma d}{\rho \nu \kappa}$ (the inverse of the nondimensional Crispation number), the associated boundary condition on the free surface is

$$\begin{aligned} \nabla_s S &= \nabla_s \vec{u}_n + \partial_{\vec{n}} \vec{u}_s && (6.2.8) \\ P - S\left(\frac{1}{R_1} + \frac{1}{R_2}\right) &= 2\partial_{\vec{n}} \vec{u}_n && (6.2.9) \end{aligned}$$

To derive the two-dimensional thin film equations, first expand the solution variables in terms of the wave number $q = 2\pi d/L$, as shown in [48], and neglect higher order terms. The lowest order equations in the interior, as given by [133], are

$$\begin{aligned} \frac{\partial^2 \vec{u}_\perp}{\partial z^2} &= \nabla_\perp P && (6.2.10) \\ \frac{\partial P}{\partial z} &= -\mathrm{G} && (6.2.11) \\ \frac{\partial^2 T}{\partial z^2} &= 0 && (6.2.12) \\ \nabla_\perp \cdot \vec{u}_\perp + \frac{\partial w}{\partial z} &= 0 && (6.2.13) \end{aligned}$$

On the liquid surface at $z = h$, and again using the notation $\nabla \equiv \nabla_\perp$ to refer to the gradient of two-dimensional functions in the $x - y$ plane,

$$\begin{aligned} \frac{\partial u}{\partial t} + \vec{u}_\perp \cdot \nabla u &= w && (6.2.14) \\ \nabla u \cdot \nabla S - \nabla u \cdot \frac{\partial \vec{u}_\perp}{\partial z} &= 0 && (6.2.15) \\ \left(\nabla u \times \nabla S - \nabla u \times \frac{\partial \vec{u}_\perp}{\partial z}\right) \cdot \hat{e}_z &= 0 && (6.2.16) \\ P + S\Delta u &= 0 && (6.2.17) \end{aligned}$$

Equation (6.2.12) implies that the temperature can be modeled as a pure conduction problem. Solving this problem with a constant heat flux $k\frac{\partial T}{\partial z}$

at the liquid/gas interface gives a surface temperature in terms of the two-layer Biot number $F \equiv \frac{d/d_g - \mathrm{H}}{1+\mathrm{H}}$.

$$T = \frac{-u}{1+F-Fu} \tag{6.2.18}$$

Integrating equation (6.2.11) in z gives

$$P = -S\Delta u + \mathrm{G}(u-z) \tag{6.2.19}$$

and substituting this into (6.2.10),

$$\frac{\partial^2 \vec{u}_\perp}{\partial z^2} = -\nabla S\Delta u + \mathrm{G}\nabla u \tag{6.2.20}$$

Next, integrating with respect to z twice, applying boundary conditions from (6.2.6) and (6.2.15),

$$\vec{u}_\perp = \left(-\nabla\left(S\Delta u\right) + \mathrm{G}\nabla u\right)\frac{z^2}{2} + \left(u\nabla S\Delta u - u\mathrm{G}\nabla u + \nabla S\right)z \tag{6.2.21}$$

Integrating (6.2.13) vertically from the no-slip boundary condition at $z = 0$,

$$w = \left(-\Delta\left(S\Delta u\right) + \mathrm{G}\Delta u\right)\frac{z^3}{6} + \left(\nabla\cdot\left(u\nabla S\Delta u\right) - \mathrm{G}\nabla\cdot\left(u\nabla u\right) + \Delta S\right) \tag{6.2.22}$$

We can then solve for $\frac{\partial u}{\partial t}$ in terms of u and S:

$$\begin{aligned}
\frac{\partial u}{\partial t} &= w(u) - \vec{u}_\perp(u)\cdot\nabla u && (6.2.23)\\
&= w(u) - \nabla\cdot\left(u\vec{u}_\perp(u)\right) + \nabla\cdot\vec{u}_\perp(u) && (6.2.24)\\
&= w(u) - \nabla\cdot\left(u\vec{u}_\perp(u)\right) - \left.\frac{\partial w}{\partial z}\right|_{z=h} && (6.2.25)\\
&= \nabla\cdot\left(\frac{u^3}{3}\nabla\left(S\Delta u\right) - \frac{u^2}{2}\nabla S + \frac{\mathrm{G}u^3}{3}\nabla u\right) && (6.2.26)
\end{aligned}$$

Let c_s denote the concentration of a monolayer surfactant on the liquid surface. The transport of this surfactant is modeled by an advection equation. Let us assume that the surface is relatively flat to justify the simplification $\nabla_\perp \approx \nabla_s$. Because surfactant diffusion based on surface tension effects will greatly exceed surfactant molecular diffusivity, we neglect the latter to obtain:

$$\frac{\partial c_s}{\partial t} + \nabla \cdot (c_s \vec{u}_\perp) = 0 \tag{6.2.27}$$

Using equation (6.2.21) for $\vec{u}_\perp$ gives an evolution equation analogous to (6.2.26):

$$\frac{\partial c_s}{\partial t} + \nabla \cdot \left(c_s \frac{u^2}{2} \left(-\nabla \left(S \Delta u \right) + \mathrm{G} \nabla u \right) + \left(u^2 \nabla S \Delta u - u^2 \mathrm{G} \nabla u + u \nabla S \right) \right) = 0 \tag{6.2.28}$$

To complete the model, the nondimensionalized surface tension S is described in terms of the film thickness u and surfactant concentration c_s. In the present work as in [134], we use a linear “dilute surfactant model” to describe the dependence of surface tension on surfactant c_s, and we also assume a linear dependence of surface tension on temperature. In terms of the nondimensionalized surface tension S, this model is:

$$S = S_0 - \mathrm{M} \delta T - \alpha E S_0 c_s \tag{6.2.29}$$

Where α and E are nondimensional coefficients in the nonlinear model for $S(c_s)$, and S_0 is the unperturbed surface tension at $T = 0$ and $c_s = 0$.

After substituting in our solution for $T(u)$, this constitutive model enables the calculation of surface tension and surface tension gradients in terms of only the height, surfactant concentration, and known material constants:

$$\begin{aligned} S &= S_0 + \frac{\mathrm{M}u}{1+F-Fu} - \alpha E S_0 c_s && (6.2.30)\\ \nabla S &= \frac{\mathrm{M}(1+F)}{(1+F-Fu)^2}\nabla u - \alpha E S_0 \nabla c_s && (6.2.31) \end{aligned}$$

Substituting these models for S and ∇S into (6.2.26) and (6.2.28), the resulting system of thin film flow equations becomes:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nabla \cdot \bigg(\frac{u^3}{3}\nabla\bigg(\bigg(S_0 + \frac{\mathrm{M}u}{1+F-Fu} - \alpha E S_0 c_s\bigg)\Delta u\bigg) - && (6.2.32)\\ &\quad \frac{u^2}{2}\frac{\mathrm{M}(1+F)}{(1+F-Fu)^2}\nabla u + \frac{u^2}{2}\nabla c_s + \frac{\mathrm{G}u^3}{3}\nabla u\bigg) \\ \frac{\partial c_s}{\partial t} &= \nabla \cdot \bigg(\bigg(\frac{u^2}{2}\nabla\bigg(\bigg(S_0 + \frac{\mathrm{M}u}{1+F-Fu} - \alpha E S_0 c_s\bigg)\Delta u\bigg) + && (6.2.33)\\ &\quad \frac{\mathrm{M}(1+F)u}{(1+F-Fu)^2}\nabla u - \alpha E S_0 u \nabla c_s\bigg) c_s\bigg) \end{aligned}$$

Finally, we rescale the domain. Recalling the inverse dynamic Bond number $D \equiv \mathrm{M}/\mathrm{G}$, x and y lengths are rescaled by L/d and time by $\frac{3L^2}{\mathrm{G}D^2}$, to obtain the thin film flow equations:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nabla \cdot \bigg(\frac{u^3}{B}\nabla\bigg(\bigg(1 + \frac{Dud^2}{(1+F-Fu)L^2} - D_s\frac{d^2}{L^2}c_s\bigg)\Delta u\bigg) - \\ &\quad \frac{3u^2}{2}\frac{D(1+F)}{(1+F-Fu)^2}\nabla u + \frac{u^2}{2}\nabla c_s + \frac{\mathrm{G}u^3}{3}\nabla u\bigg) && (6.2.34)\\ \frac{\partial c_s}{\partial t} &= \nabla \cdot \bigg(\bigg(\frac{3u^2}{2B}\nabla\bigg(\bigg(1 + \frac{Dud^2}{(1+F-Fu)L^2} - D_s\frac{d^2}{L^2}c_s\bigg)\Delta u\bigg) + \\ &\quad \frac{D(1+F)u}{(1+F-Fu)^2}\nabla u - D_s u \nabla c_s\bigg) c_s\bigg) && (6.2.35) \end{aligned}$$

Where $D_s \equiv \frac{\alpha E \sigma_0}{\rho g d^2}$ is the dimensionless parameter corresponding to the ratio of gravitational and surfactant forces.

Dilute surfactant model PDE In the analysis of double-layer thin film flow by Van Hook et. al. [129], as well as in its extension to surfactant problems by Wang and Carey [133], some terms in the derived equations are simplified by the assumption $S \approx S_0$. Above, we avoid this assumption, but we can see that the extra terms in this new derivation can be safely neglected based on the behavior of the nondimensionalized equations for $d/L << 1$. The numerical experiments here use these final equations, where terms with d^2/L^2 have been dropped:

$$\begin{aligned} \frac{\partial u}{\partial t} &= \nabla \cdot \left(\left(u^3 - \frac{3D(1+F)u^2}{2(1+F-Fu)^2} \right) \nabla u - \right. \\ &\quad \left. \frac{u^3}{B} \nabla \Delta u + \frac{3D_s u^2}{2} \nabla c_s \right) \end{aligned} \tag{6.2.36}$$

$$\begin{aligned} \frac{\partial c_s}{\partial t} &= \nabla \cdot \left(\left(\frac{u^2}{2} - \frac{3D(1+F)u}{(1+F-Fu)^2} \right) c_s \nabla u - \right. \\ &\quad \left. \frac{c_s u^2}{2B} \nabla \Delta u + 3D_s c_s u \nabla c_s \right) \end{aligned} \tag{6.2.37}$$

6.3 Galerkin formulation

Previous finite element simulations of thin film flow [10, 135] have employed a mixed method to handle the fourth-order terms in the governing differential equations; introducing a secondary variable which includes Δu in its definition to break the fourth-order equation into a pair of coupled second order equations.

Here, we instead work directly in H^2 using C^1 elements to solve the fourth-order form. Taking the weighted residual of each PDE (against a test function v for the evolution equation of height u and a test function w for the evolution equation of surfactant c_s), then integrating each second order term by parts once and each fourth-order term twice,

$$\begin{aligned}
\left(\frac{\partial u}{\partial t}, v\right) &= \left(\left(\frac{3D(1+F)u^2}{2(1+F-Fu)^2} - u^3\right)\nabla u - \frac{3D_s u^2}{2}\nabla c_s, \nabla v\right)_\Omega + \\
&\quad \left(\left(u^3 - \frac{3D(1+F)u^2}{2(1+F-Fu)^2}\right)\partial_{\vec{n}} u + \frac{3D_s u^2}{2}\partial_{\vec{n}} c_s, v\right)_{\partial\Omega} - \\
&\quad \left(\frac{u^3}{B}\Delta u, \Delta v\right)_\Omega - \left(\frac{3u^2}{B}\Delta u \nabla u, \nabla v\right)_\Omega + \\
&\quad \left(\frac{u^3}{B}\partial_{\vec{n}}\Delta u, v\right)_{\partial\Omega} - \left(\frac{u^3}{B}\Delta u, \partial_{\vec{n}} v\right)_{\partial\Omega}
\end{aligned} \tag{6.3.1}$$

$$\begin{aligned}
\left(\frac{\partial c_s}{\partial t}, w\right) &= \left(\left(\frac{3D(1+F)u}{(1+F-Fu)^2} - \frac{u^2}{2}\right) c_s \nabla u - 3D_s c_s u \nabla c_s, \nabla w\right)_\Omega + \\
&\quad \left(\left(\frac{u^2}{2} - \frac{3D(1+F)u}{(1+F-Fu)^2}\right) c_s \partial_{\vec{n}} u + 3D_s c_s u \partial_{\vec{n}} c_s, w\right)_{\partial\Omega} - \\
&\quad \left(\frac{c_s u^2}{2B}\Delta u, \Delta w\right)_\Omega - \left(\Delta u \frac{2c_s u \nabla u + u^2 \nabla c_s}{2B}, \nabla w\right)_\Omega + \\
&\quad \left(\frac{c_s u^2}{2B}\partial_{\vec{n}}\Delta u, w\right)_{\partial\Omega} - \left(\frac{c_s u^2}{2B}\Delta u, \partial_{\vec{n}} w\right)_{\partial\Omega}
\end{aligned} \tag{6.3.2}$$

The nonlinear terms in equations (6.3.1) and (6.3.2) merit further comment. First, because the fourth-order terms each include a nonlinear coefficient "outside" three derivatives, integrating them by parts twice requires differentiating that coefficient, which creates an unusual additional interior integral term in each equation. Secondly, because of the appearance of $(1+F-Fu)^2$ in the denominator of several terms, the Galerkin functionals are not bounded on any Hilbert space. Solutions with nondimensionalized height $u(\vec{x}) > (1+F)/F$ are part of any linear function space, but are physically impossible in the experimental problem because of the barrier of the upper plate, and are precluded in the Galerkin functionals because the coefficients which become singular. The Galerkin functionals are only defined for u in the subset of $W^{2,5}$ constrained by $u < (1+F)/F - \varepsilon$.

6.4 Thin Film Flow Results

To demonstrate the coupling of surfactant concentration to thin film depth, this simulation adds a round droplet of surfactant to the surface of a film of initially uniform depth. The spreading of the surfactant layer entrains fluid with it away from the initial droplet location in an expanding wavefront.

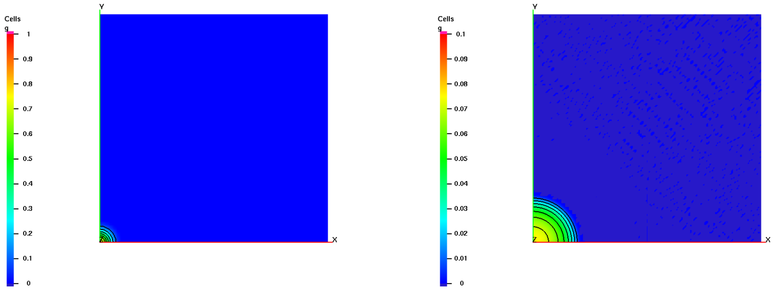

Figure 6.1: Initial surfactant concentration, and a thin film flow solution at $t = 0.2$.

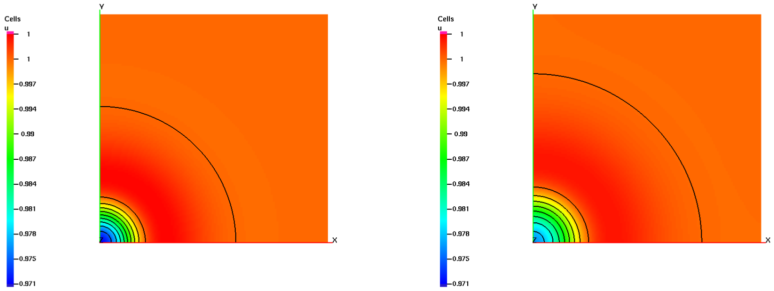

Figure 6.2: Film fluid depth solutions at $t = 0.1$ and $t = 0.2$.

Chapter 7

Cahn-Hilliard Phase Separation

7.1 Introduction

The Cahn-Hilliard treatment of interfacial free energy [26] has been extended to describe the evolution of material composition in many mixtures with diffuse interfaces, from microscale annealing to nanoscale void self-assembly [79, 119, 126, 144]. By adding additional variables for concentrations of additional components [60] and/or material phase [83], systems of Cahn-Hilliard like equations can be used to describe evolution of more complicated mixtures with known free energy functions. Even nominally immiscible fluid mixtures can be modeled by coupling the Cahn-Hilliard and Navier-Stokes equations [91]. In the sharp interface limit, the Cahn-Hilliard contributions to such a system are equivalent to classical surface tension models, but without the front tracking [145] required by other methods.

7.2 Cahn-Hilliard Equation

In a Cahn-Hilliard system, the material state is described by a continuous variable, rather than a discontinuous partition of the domain into regions of differing state. In the simplest binary systems [26], the material state is described by the the concentration c of one of the two components. The local free energy density f in a material is assumed to be a function of both c

and ∇c. For an isotropic material, the composition based free energy density will be a function of c and $||\nabla c||^2$ alone. The free energy is generally divided into two parts, $f = f_0 + f_\gamma$. The "configurational free energy" f_0 is present in mixtures at a homogenous concentration and the "surface free energy" or "gradient energy" f_γ is positive at material interfaces where a concentration gradient exists. The isotropic gradient energy term is described by

$$f_\gamma(\nabla c) \equiv \frac{\epsilon_c^2}{2} \nabla c \cdot \nabla c \tag{7.2.1}$$

where the parameter ϵ_c scales with the interfacial layer thickness.

The configurational free energy depends on the chemistry of a particular physical problem. The initial work by Cahn and Hilliard [26], as well as most of the subsequent material science literature, uses what we will call the "chemical" free energy function

$$f_{0c}(c) \equiv NkT\left(c\ln(c) + (1-c)\ln(1-c)\right) + N\omega c(1-c) \tag{7.2.2}$$

for mixtures involving small molecules. Here N is the molecular density, k is Boltzmann's constant, and ω is a scalar parameter related to fluid miscibility. Below a critical temperature $T < \frac{\omega}{2k}$, this free energy function is a double well which will lead to phase separation. In this formulation the phase c is a concentration with values varying from 0 to 1. This function is also called the "Flory-Huggins" free energy, due to its original derivation by Huggins [71] and Flory [63] in studies of polymer solutions.

In much of the mathematical analysis literature (see e.g. [56], a simplified function is often used to investigate the same qualitative behavior, by giving an equation that avoids the restricted range of and the singularities in the chemical free energy definition. This “mathematical“ free energy function is the quartic double well,

$$f_{0m}(c) \equiv \frac{1}{4}\left(c^2 - 1\right)^2 \tag{7.2.3}$$

In this mathematical free energy function, the phase c is the normalized difference in the two concentrations in a binary mixture, and so c is expected to vary in the range from -1 to 1. In fact f_{0m} is a simplified approximation, and depending on other parameters in the Cahn-Hilliard equation the phase c can take unphysical values outside of that range.

We relate the time evolution of c to the concentration flux $\vec{q}$, ensuring that concentration is a conserved variable.

$$\frac{\partial c}{\partial t} = -\nabla \cdot \vec{q} \tag{7.2.4}$$

Given the defined free energy $f_0 + f_\gamma$, a thermodynamic analysis provides a constitutive relation for the concentration flux. The flux $\vec{q}$ is proportional to the free energy gradient

$$\begin{aligned} \vec{q} &= -M_c \nabla \frac{df}{dc} && (7.2.5) \\ &= -M_c \nabla \left(f_0'(c) + f_\gamma'(c)\right) && (7.2.6) \end{aligned}$$

where the mobility M_c may be concentration dependent. M_c must be a positive semidefinite tensor-valued function of c, and is typically assumed to take non-negative scalar values. Of interest when using the chemical free energy function are the constant mobility function $M_c = M$ and the “degenerate” mobility function $M_c = Mc(1-c)$. Degenerate mobility functions are of special interest in the analysis of the Cahn-Hilliard equation with the mathematical free energy function, where they guarantee strict bounds on c [54].

When using the mathematical free energy function f_{0m}, the derivative f'_{0m} is also a simple well-behaved polynomial

$$f'_{0m} = c^3 - c \tag{7.2.7}$$

However, for the chemical free energy function f_{0c} the derivative f'_{0c} is

$$f'_{0c} = NkT\left(\ln(c) - \ln(1-c)\right) + N\omega(1-2c) \tag{7.2.8}$$

and becomes singular at the concentration limits $c = 0$ and $c = 1$.

Substituting $\vec{q}$ into the conservation equation completes the construction of the basic normalized Cahn-Hilliard equation:

$$\frac{\partial c}{\partial t} = \nabla \cdot M_c \nabla \left(f'_0(c) - \epsilon_c^2 \Delta c\right) \tag{7.2.9}$$

7.3 Lyapunov Energy Functional

The Cahn-Hilliard equations can be interpreted in terms of the time evolution of the free energy functional

$$\begin{aligned} J(c) &\equiv \int_\Omega f(c)\, d\Omega && (7.3.1) \\ &= \int_\Omega \left[f_0(c) + \frac{\epsilon_c^2}{2} \nabla c \cdot \nabla c\right] d\Omega && (7.3.2) \end{aligned}$$

The time derivative of this functional

$$\frac{\partial J(c)}{\partial t} = \int_\Omega \left[f'_0(c) \frac{\partial c}{\partial t} + \epsilon_c^2 \nabla c \cdot \nabla \frac{\partial c}{\partial t}\right] d\Omega \tag{7.3.3}$$

can be integrated by parts, giving L^2 inner products over the domain and it's boundary

$$\begin{aligned}\frac{\partial J(c)}{\partial t} &= \left(f_0'(c) - \epsilon_c^2 \Delta c, \frac{\partial c}{\partial t}\right)_\Omega + \\ & \quad \left(\epsilon_c^2 \partial_{\vec{n}} c, \frac{\partial c}{\partial t}\right)_{\partial\Omega} \end{aligned} \tag{7.3.4}$$

If c satisfies the symmetry boundary condition $\partial_{\vec{n}} c = 0$ on $\partial\Omega$, the boundary term is zero; if c satisfies periodic boundary conditions on $\partial\Omega$ then boundary terms from matched boundaries cancel. In either case,

$$\frac{\partial J(c)}{\partial t} = \left(f_0'(c) - \epsilon_c^2 \Delta c, \frac{\partial c}{\partial t}\right)_\Omega \tag{7.3.5}$$

This time derivative is non-positive if c satisfies the Cahn-Hilliard equation on Ω along with its natural boundary condition $(M_c \nabla (f_0'(c) - \epsilon_c^2 \Delta c)) \cdot \vec{n} = 0$ on $\partial\Omega$.

$$\begin{aligned}\frac{\partial J(c)}{\partial t} &= \left(f_0'(c) - \epsilon_c^2 \Delta c, \nabla \cdot M_c \nabla \left(f_0'(c) - \epsilon_c^2 \Delta c\right)\right) && (7.3.6) \\ &= -\left(\nabla \left(f_0'(c) - \epsilon_c^2 \Delta c\right), M_c \nabla \left(f_0'(c) - \epsilon_c^2 \Delta c\right)\right)_\Omega && (7.3.7)\end{aligned}$$

The final inner product on the right hand side will always be non-negative so long as the mobility $M_c(c)$ is either a strictly non-negative scalar or a strictly positive semidefinite tensor for all c.

In both the mathematical and chemical free energy formulations, $f_0(c)$ is bounded from below by some f_{min}. Also, $\nabla c \cdot \nabla c$ is non-negative; therefore $J(c)$ is bounded from below in $H^1(\Omega)$.

$J(c)$ is non-increasing, bounded from below, C^1 on the set of admissible c, and radially unbounded, i.e. it fulfills the definition of a "smooth weak Lyapunov function in the large" for the Cahn-Hillard problem.

7.4 Galerkin Finite Element Approximation

Taking a weighted residual of the strong equation (7.2.9), integrate the second order terms by parts once and integrate the fourth-order term by parts twice. This yields

$$
\begin{aligned}
(\frac{\partial c}{\partial t}, \phi)_\Omega &\equiv \\
F(c, \phi) &= -\left(M_c \nabla f_0'(c), \nabla\phi\right)_\Omega - \epsilon_c^2 \left(\Delta c, \nabla \cdot M_c^T \nabla \phi\right)_\Omega \qquad (7.4.1) \\
&\quad + \left(\left(M_c \nabla \left(f_0'(c) - \epsilon_c^2 \Delta c\right)\right) \cdot \vec{n}, \phi\right)_{\partial\Omega} + \epsilon_c^2 \left(\Delta c, M_c^T \nabla \phi \cdot \vec{n}\right)_{\partial\Omega}
\end{aligned}
$$

and is defined on $W^{2,2}(\Omega) \cap W^{1,4}(\Omega)$ in the simplest case; in general the domain of the functional depends on M_c and $f_0(c)$.

As boundary conditions, we use symmetry conditions to approximate an infinite domain. To enforce symmetry, it is sufficient to require that $\left(M_c^T \nabla c\right) \cdot \vec{n} = 0$ and $(M_c \nabla (f_0'(c) - \epsilon_c^2 \Delta c)) \cdot \vec{n} = 0$ on $\partial\Omega$. The latter condition may be enforced weakly as a natural boundary condition by substituting 0 into the corresponding boundary integral in F. Although constraints on $\partial_{\vec{n}} c$ are familiar as natural boundary conditions in second order problems, in this fourth-order problem constraints on the normal flux are essential boundary conditions.

The homogeneous essential boundary condition on first derivatives may be conveniently enforced approximately via a penalty method, by adding the term

$$
\frac{1}{\epsilon} \left(M_c^T \nabla c \cdot \vec{n}, M_c^T \nabla \phi \cdot \vec{n}\right) \qquad (7.4.2)
$$

to the residual functional using some $\epsilon \ll 1$. Comparing this term with the second boundary integral in (7.4.1), it can be seen that the penalty method is equivalent to enforcing the mixed boundary condition $M_c^T \nabla \phi \cdot \vec{n} = \epsilon \epsilon_c^2 \Delta c$. As $\epsilon \to 0$, solutions obtained with this boundary condition should approach solutions obtained with the exact boundary condition.

In the past, approximation of the Cahn-Hilliard equation using finite element methods has often focused on mixed methods, introducing a separate scalar variable for Δc or for $f_0'(c) - \epsilon_c^2 \Delta c$, then solving the resulting system of equations on a C^0 finite element space. See e.g. [55, 80]. Other finite element approaches in the literature include semi-discontinuous and non-conforming methods [11, 53, 137].

By using C^1 continuous finite element spaces for our approximation space H^h, we can find conforming approximate solutions to the fourth-order Cahn-Hilliard equation without introducing the additional variable and degrees of freedom required by a mixed system, using spaces with fewer degrees of freedom and equations with fewer terms to integrate than would be required by a semi-discontinuous method. In this work, equation (7.4.1) is enforced directly for all test functions $\phi \in H^h$. This method has previously been used with a slightly different weak form on the one dimensional Cahn-Hilliard problem [52]; the present work extends the method to two and three dimensional problems, on adaptive meshes of tensor product spline elements or HCT macroelements.

7.5 Lyapunov Energy Functional and Spatial Discretization

When approximating partial differential equations such as the Cahn-Hilliard equation, we often desire that solutions to the discretized formulation retain the conservation and dissipation properties of the original equation.

Mass conservation, which may require special attention with other numerical methods [41], is a simple property to verify with our Galerkin scheme. Because the constant function $v = 1$ is contained within our finite element function space, we can substitute it for ϕ in (7.4.1). The left side of the equation simply becomes the time derivative of the integral of c over Ω, and the right side of the equation becomes zero.

In the numerical analysis literature, however, it is also often desired that an approximate solution exhibit the same free energy dissipation property as the true solution, whether for the original free energy functional as in [143] or for a discrete version of that functional as in [64].

Although the Galerkin formulation is merely a natural restriction of the weak Cahn-Hilliard equation to the finite space H^h, and although the free energy functional J is defined on that space, the arguments in Section 7.3 do not necessarily apply in the discrete case. With c replaced by c_h, equation (7.3.5) applies to the Galerkin solution, but equation (7.3.6) does not, except when $f_0'(c) - \epsilon_c^2 \Delta c_h$ is an admissible trial function in H^h. For standard finite element spaces H^h or for the chemical free energy function f_0, that will not be the case.

As our later numerical results demonstrate, for many problems the standard Galerkin formulation does produce a non-increasing free energy functional in practice. However, to guarantee such a result for all problems it is necessary to modify the weak formulation, as follows:

Define the L_2 projection operator $P_h : L_2(\Omega) \to H^h$ such that, for all $v_h \in H^h$ and all $u \in L_2(\Omega)$, P_h satisfies:

$$(P_h u, v_h)_\Omega = (u, v_h)_\Omega \tag{7.5.1}$$

Then we can modify the Galerkin formulation of the semidiscrete Cahn-Hillard equations in such a way that they preserve the Cahn-Hilliard free energy properties in the same manner as the continuous equations. The modified formulation includes the L_2 projection operator as follows:

$$\begin{aligned}
(\frac{\partial c_h}{\partial t}, \phi)_\Omega &\equiv F(c_h, \phi) \\
&= -\left(M_c \nabla P_h f_0'(c_h), \nabla\phi\right)_\Omega - \epsilon_c^2 \left(P_h \Delta c_h, \nabla \cdot M_c^T \nabla\phi\right)_\Omega \\
&\quad + \left(\left(M_c \nabla \left(P_h f_0'(c_h) - \epsilon_c^2 P_h \Delta c_h\right)\right) \cdot \vec{n}, \phi\right)_{\partial\Omega} \\
&\quad + \epsilon_c^2 \left(P_h \Delta c_h, M_c^T \nabla\phi \cdot \vec{n}\right)_{\partial\Omega}
\end{aligned} \tag{7.5.2}$$

Weakly applying the natural boundary condition $(M_c \nabla (f_0'(c) - \epsilon_c^2 \Delta c)) \cdot \vec{n} = 0$ on $\partial\Omega$, the first boundary integral term vanishes.

In this case, following the steps taken with the continuous model in equation (7.3.5), the free energy functional $J(c_h)$ for a discrete model has time derivative

$$\frac{\partial J(c_h)}{\partial t} = \left(f_0'(c_h) - \epsilon_c^2 \Delta c_h, \frac{\partial c_h}{\partial t}\right)_\Omega \tag{7.5.3}$$

Because $c_h(t) \in H^h$ at all times t, finite differences $(c_h(t + \delta t) - c_h)/\delta t$ are also in H^h. For c_h differentiable and H^h closed, this implies that $\frac{\partial c_h}{\partial t} \in H^h$.

The L_2 projection operator onto H^h therefore applies, giving

$$\frac{\partial J(c_h)}{\partial t} = \left(P_h f_0'(c_h) - \epsilon_c^2 P_h \Delta c_h, \frac{\partial c_h}{\partial t} \right)_\Omega \tag{7.5.4}$$

After this projection, $P_h(f_0'(c_h) - \epsilon_c^2 \Delta c_h)$ is an element of H^h, and can be substituted for ϕ in (7.5.2).

$$\begin{aligned} \frac{\partial J(c_h)}{\partial t} &= -\left(M_c \nabla P_h f_0'(c_h), \nabla P_h(f_0'(c_h) - \epsilon_c^2 \Delta c_h) \right)_\Omega - \\ & \quad \epsilon_c^2 \left(P_h \Delta c_h, \nabla \cdot M_c^T \nabla P_h(f_0'(c_h) - \epsilon_c^2 \Delta c_h) \right)_\Omega \\ & \quad + \epsilon_c^2 \left(P_h \Delta c_h, M_c^T \nabla P_h(f_0'(c_h) - \epsilon_c^2 \Delta c_h) \cdot \vec{n} \right)_{\partial\Omega} \end{aligned} \tag{7.5.5}$$

Integrating the second interior integral by parts cancels out the boundary integral. Combining similar terms simplifies the result to

$$\frac{\partial J(c_h)}{\partial t} = -\left(M_c \nabla P_h(f_0'(c_h) - \epsilon_c^2 \Delta c_h), \nabla P_h(f_0'(c_h) - \epsilon_c^2 \Delta c_h) \right)_\Omega \tag{7.5.6}$$

For a positive semidefinite mobility coefficient M_c, this implies that $J(c_h)$ will be non-increasing for the time-continuous Cahn-Hilliard problem with spatially discretized c_h.

7.6 Lyapunov Energy Functional and Crank-Nicolson Time Discretization

In order to guarantee a non-increasing free energy functional when a time discretization is introduced to produce a fully discrete approximation problem, additional care is necessary. For a Crank-Nicolson-like finite differencing time discretization scheme, for example, we would like to begin with the spatially discretized equation

$$\left(\frac{\partial c_h}{\partial t}, \phi \right)_\Omega = F(c_h, \phi) \tag{7.6.1}$$

and approximate the time derivative by a two-point difference scheme, while evaluating the right hand side in the middle of the timestep, at the interpolated solution $c_h^{(\theta)} \equiv (c_h^{(n)} + c_h^{(n+1)})/2$, to retain a higher order accuracy:

$$\left(\frac{c_h^{(n+1)} - c_h^{(n)}}{\delta t}, \phi \right)_\Omega = F((c_h^{(\theta)}, \phi) \tag{7.6.2}$$

When applying this formulation to the Cahn-Hilliard equation, however, information about the potentially highly nonlinear configurational free energy function f_0 is only introduced to the approximate formulation in the term $f_0'(c_h^{(\theta)})$. To preserve the non-increase of the total free energy functional, we can replace this analytic derivative f_0' with its own finite difference approximation which depends on our timestep.

Let $c_h^{(n+1)}$ be the solution of the fully discretized equation

$$\begin{aligned}
\left(\frac{c_h^{(n+1)} - c_h^{(n)}}{\delta t}, \phi \right)_\Omega &= \tilde{F}(c_h^{(n+1)}, c_h^{(n)}, \phi) \\
&= -\left(M_c \nabla P_h \tilde{f}_0(c_h^{(n+1)}, c_h^{(n)}), \nabla \phi \right)_\Omega \\
&\quad -\epsilon_c^2 \left(P_h \Delta c_h^{(\theta)}, \nabla \cdot M_c^T \nabla \phi \right)_\Omega \\
&\quad + \left(\left(M_c \nabla \left(P_h \tilde{f}_0'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 P_h \Delta c_h^{(\theta)} \right) \right) \cdot \vec{n}, \phi \right)_{\partial\Omega} \\
&\quad + \epsilon_c^2 \left(P_h \Delta c_h^{(\theta)}, M_c^T \nabla \phi \cdot \vec{n} \right)_{\partial\Omega}
\end{aligned} \tag{7.6.3}$$

where the approximate configurational energy derivative $\tilde{f}_0$ is defined as $\tilde{f}_0{}'(c_h^{(n+1)}, c_h^{(n)}) \equiv f_0'(c_h^{(n+1)})$ when $c_h^{(n+1)} = c_h^{(n)}$ and otherwise as

$$\tilde{f}_0{}'(c_h^{(n+1)}, c_h^{(n)}) \equiv \frac{f_0(c_h^{(n+1)}) - f_0(c_h^{(n)})}{c_h^{(n+1)} - c_h^{(n)}} \tag{7.6.4}$$

We again consider the case of a weakly enforced homogeneous natural boundary condition $(M_c \nabla (f_0'(c) - \epsilon_c^2 \Delta c)) \cdot \vec{n} = 0$ on $\partial\Omega$, in which the first boundary integral term vanishes.

Consider the change in the free energy functional J between two successive timesteps $t^{(n)}$ and $t^{(n+1)}$:

$$\begin{aligned} J(c_h^{(n+1)}) - J(c_h^{(n)}) &= \int_\Omega \left[f_0(c_h^{(n+1)}) - f_0(c_h^{(n)}) + \frac{\epsilon_c^2}{2} \left(\left| \nabla c_h^{(n+1)} \right|^2 - \left| \nabla c_h^{(n)} \right|^2 \right) \right] d\Omega \\ &= \int_\Omega \left[f_0(c_h^{(n+1)}) - f_0(c_h^{(n)}) + \epsilon_c^2 \nabla c_h^{(\theta)} \cdot \nabla (c_h^{(n+1)} - c_h^{(n)}) \right] d\Omega \end{aligned}$$

Multiplying and dividing the configurational free energy terms by $c_h^{(n+1)} - c_h^{(n)}$, then integrating the interfacial free energy terms by parts, the remaining terms are the inner products

$$\begin{aligned} J(c_h^{(n+1)}) - J(c_h^{(n)}) &= \left(\tilde{f}_0{}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 \Delta c_h^{(\theta)}, (c_h^{(n+1)} - c_h^{(n)}) \right)_\Omega + \\ & \quad \epsilon_c^2 \left(\partial_{\vec{n}} c_h^{(\theta)}), c_h^{(n+1)} - c_h^{(n)} \right)_{\partial\Omega} \end{aligned} \tag{7.6.5}$$

With the essential symmetry boundary condition $\partial_{\vec{n}} c_h = 0$, the boundary term disappears. Because $(c_h^{(n+1)} - c_h^{(n)})$ is an element of H^h, the projection operator P_h can be applied to left hand argument of the remaining inner product.

$$J(c_h^{(n+1)}) - J(c_h^{(n)}) = \left(P_h \tilde{f}_0{}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 P_h \Delta c_h^{(\theta)}, (c_h^{(n+1)} - c_h^{(n)}) \right)_\Omega \tag{7.6.6}$$

After dividing both sides of the equation by δt, the inner product is now in the same form as in the fully discretized equations,

$$\left(\frac{c_h^{(n+1)} - c_h^{(n)}}{\delta t}, \phi \right) \tag{7.6.7}$$

Applying those equations then gives

$$\begin{aligned}
\frac{J(c_h^{(n+1)}) - J(c_h^{(n)})}{\delta t} &= \tilde{F}(c_h^{(n+1)}, c_h^{(n)}, P_h \tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 P_h \Delta c_h^{(\theta)}) \\
&= -\Big(M_c \nabla P_h \tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}), \qquad (7.6.8) \\
&\quad \nabla \left(P_h \tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 P_h \Delta c_h^{(\theta)} \right) \Big)_\Omega - \\
&\quad \epsilon_c^2 \left(P_h \Delta c_h^{(\theta)}, \nabla \cdot M_c^T \nabla \left(P_h \tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 P_h \Delta c_h^{(\theta)} \right) \right)_\Omega \\
&\quad + \epsilon_c^2 \left(P_h \Delta c_h^{(\theta)}, M_c^T \nabla \left(P_h \tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 P_h \Delta c_h^{(\theta)} \right) \cdot \vec{n} \right)_{\partial\Omega}
\end{aligned}$$

Integrating the second interior integral by parts cancels out the boundary integral, leaving terms which simplify to

$$\begin{aligned}
\frac{J(c_h^{(n+1)}) - J(c_h^{(n)})}{\delta t} &= -\Big(M_c \nabla P_h (\tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 \Delta c_h^{(\theta)}), \qquad (7.6.9) \\
&\quad \nabla P_h (\tilde{f_0}'(c_h^{(n+1)}, c_h^{(n)}) - \epsilon_c^2 \Delta c_h^{(\theta)}) \Big)_\Omega
\end{aligned}$$

For a positive semidefinite mobility coefficient M_c, this value is never positive, and so we expect the free energy functional J to never increase from one timestep to the next.

7.7 Cahn Hilliard Evolution

In the results in Sections 7.8 and 7.9, a constant mobility $M_c = 1$ is used in the PDE. Backward Euler time integration was used, with uniform timestep lengths ranging from 0.001 to 0.004. Later results, including the 3D simulations and the parametric studies, use a variable mobility $M_c = c(1 - c)$. These results also use a trapezoidal rule time integrator with adaptive time step selection. In all of the Cahn-Hilliard simulations to follow, a penalty value of $\epsilon = 10^{-10}$ is used to apply Dirichlet boundary conditions where applicable, and direct algebraic constraints are used when periodic boundary conditions are applied.

7.8 Cross Interface

To examine the interface contraction effects exhibited by Cahn-Hilliard phase evolution, we can use a cross-shaped initial condition as suggested by [80]. Instead of using a discontinuous initial condition on a sharp-cornered cross, however, we choose a C^1 cubic initial interface function and a cross shape beveled by circular arcs.

In Figure 7.1 we can see the solution and adaptive mesh behavior for a cross benchmark using the mathematical free-energy model with a constant mobility. The adaptive refinement is controlled to maintain approximately 1024 elements in the solution; in so doing it produces an interface resolution equivalent to a uniform mesh of 4096 elements. Although this algorithm is capable of tracking the moving boundary very accurately, it does not adapt to the decreasing length of the boundary, and so in later time solutions we can see a few unnecessarily overrefined cells.

The interface first quickly diffuses from the arbitrary width specified in the initial conditions to the equilibrium interface width for the problem. Next, the process of free-energy minimization acts to reduce the interface length. For this initial condition, we know a priori that the minimum free energy will be achieved by a circular domain, and so this benchmark serves as a qualitative verification of the formulation and adaptive implementation. Some Cahn-Hilliard simulations can exhibit a non-physical "pinning" on coarse meshes [37], but we verify here that even the extremely coarse phase-interior elements created by proper adaptivity do not cause such numerical artifacts.

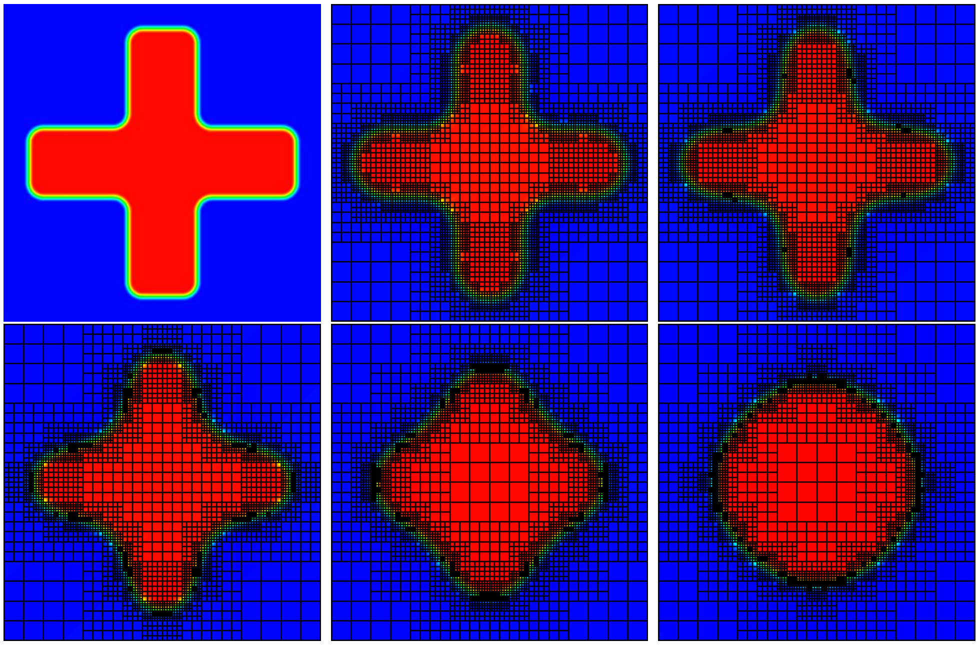

Figure 7.1: Cross benchmark initial conditions and Cahn-Hilliard solutions at $t = 0.025, 0.050, 0.100, 0.200, 0.400$. Hermite cubic elements are each plotted as four bilinear squares.

7.9 Spinodal Decomposition

A classic application case for the Cahn-Hilliard problem is the spinoidal decomposition of a random initial condition [25]. The first phase of the system evolution is anti-diffusionary, and rapidly divides much of the domain into two regions of nearly homogenous concentration, one at each of the binodal points of the configurational free energy function, thus minimizing the configurational free energy. With a random initial perturbation, however, these regions are heavily interleaved and the interface between them is long and convoluted. This stage of a numerical experiment, run on a uniform mesh of 3200 Hsieh-Clough-Tocher (HCT) triangles, is shown in Figure 7.2. In this example, an initial Cartesian grid of squares is defined, and each square cell is divided from lower left to upper right into two HCT triangles. The orientation of the triangle grid creates an anisotropic field for the initial conditions, which are generated by taking a homogeneous concentration field and then perturbing each value-based degree of freedom coefficient by a small random number. The initial data is shown in Figure 7.2, where the anisotropy is exaggerated by the GMV plotting software, which interpolates each piecewise-cubic HCT triangle by subdividing it once and interpolates the solution on that subdivision with four linear triangles for visualization. Even with these plots, solution results appear isotropic after only a few timesteps.

In the second phase of the system development, the surface free energy is more gradually reduced. The diffuse interfaces are shortened in an effect resembling surface tension on a sharp interface, and the topology of the material regions simplifies as nearby regions of matching composition merge.

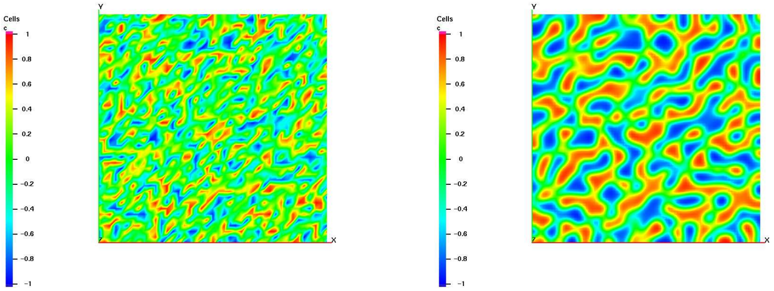

Figure 7.2: Initial conditions with random nodal values and zero edge fluxes, and a Cahn-Hilliard solution at $t = 0.001$.

Simulated spinodal decompositions using Clough-Tocher and Hermite tensor product elements demonstrate this behavior, as seen from the later stages of the Clough-Tocher experiment in Figure 7.3.

Finally, the Cahn-Hilliard system approaches a steady state solution whose precise location and size depends on the particular initial total concentration quantity and distribution. The steady-state solution cannot be found by solving the stationary form of the Cahn-Hilliard equation, and cannot be found efficiently by timestepping schemes with sufficiently small timesteps to resolve the initial transient behavior. The use of adaptive timestepping will be necessary for efficient longer simulations.

In 3D, the behavior is similar, although computations at a similar grid resolution are significantly more expensive. A solution on a uniform 32,768 hex mesh with a quarter million degrees of freedom is shown in Figure 7.4. This calculation requires minutes of clock time per timestep even on sixteen processors.

Qualitatively, this result exhibits some of the same behavior seen in the

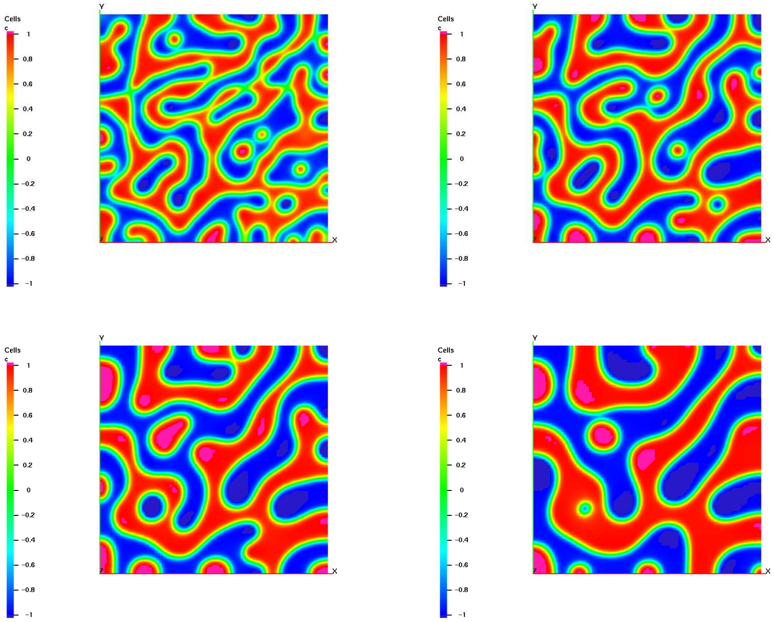

Figure 7.3: Cahn-Hilliard solutions at $t = 0.005, 0.01, 0.02, 0.05$. The interface widths are now effectively constant, and interface lengths progressively shorten.

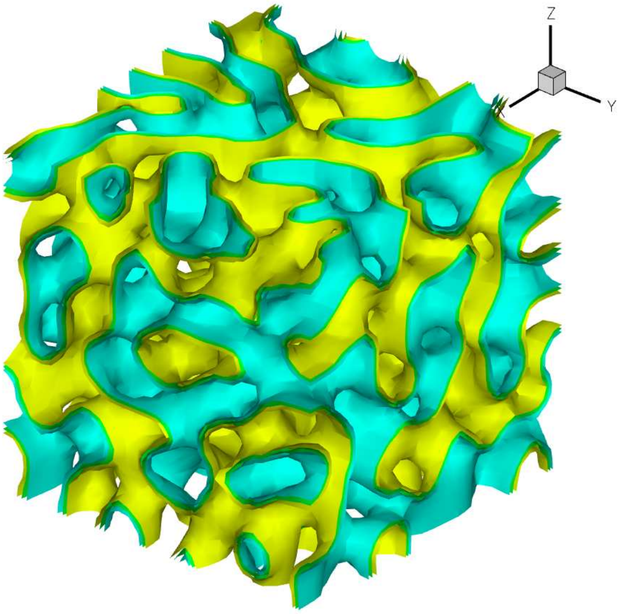

Figure 7.4: Concentration isosurfaces for a 3D spinodal decomposition problem at $t = 0.033$. Separation between the three parallel isosurface manifolds is approximately equal to interface width.

two dimensional studies; e.g., as time progresses, surface tension effects smooth and shorten the material interfaces. However, one of the most important differences in the three-dimensional problem is topological. In two dimensions, it is impossible for all four sides of a square domain to be connected by single-phase paths through each of two material components. In three dimensions, because material regions can pass unbroken over or under each other, it is not only possible but likely for all six sides of a cubic domain to be joined by connected regions of both materials. When estimating average properties of the mixture based on distinct material properties of each phase, this difference may have a large effect on the homogenized bulk values.

The transient behavior of the free energy functional for a 2D spinodal decomposition problem is plotted in Figure 7.5. As anticipated, the free energy of the discretization is monotonically decreasing with time.

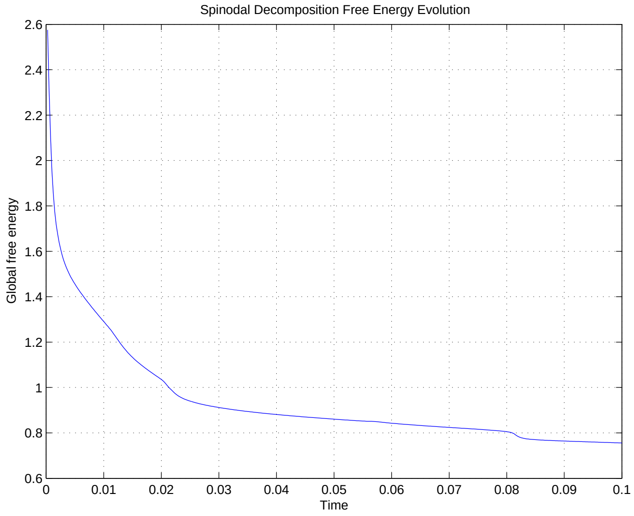

Figure 7.5: Integrated free energy over the square domain for the Galerkin approximation to a 2D spinodal decomposition problem.

7.10 Mesh Refinement

Because of the smoothing properties of the fourth-order operator in the Cahn-Hilliard equation, errors which occur on an individual time step in a transient simulation tend to decay rather than grow in future time steps. In Figure 7.6, the time evolution of the L_2 error for a typical transient spinodal decomposition problem is shown, for different spatial meshes and time steps with the same 1% initial perturbation conditions. The discretization sequence is generated by uniform refinement in space and time, and the finest discretization is used as a reference solution for error calculations. The coarsest solution is obtained with a uniform Hermite mesh with spacing $h_1 = 1/32$ and time step $\delta t = 1/4000$; the reference solution is obtained with $h_4 = 1/256$ and $\delta t = 1/32000$. The dissipation in the Cahn-Hilliard operator prevents exponential accumulation of the error with increasing time. Instead, as seen in Figure 7.7, the coarsest approximate solution develops topological differences to the reference solution, which lead to the slow but monotonic error growth in Figure 7.6. In the finer two approximations, the solution becomes first topologically and then visually indistinguishable from the reference solution, and the L_2 error varies but remains bounded as time increases.

In even a moderately developed Cahn-Hilliard system, the solution begins to be characterizable as a collection of single-phase regions with nearly constant concentration, separated by a maze of thin interfaces of rapidly varying concentration. Efficiently approximating such solutions is a natural task for adaptive h refinement, to allocate degrees of freedom along thin interfaces where they can have a significant effect on solution accuracy and to remove

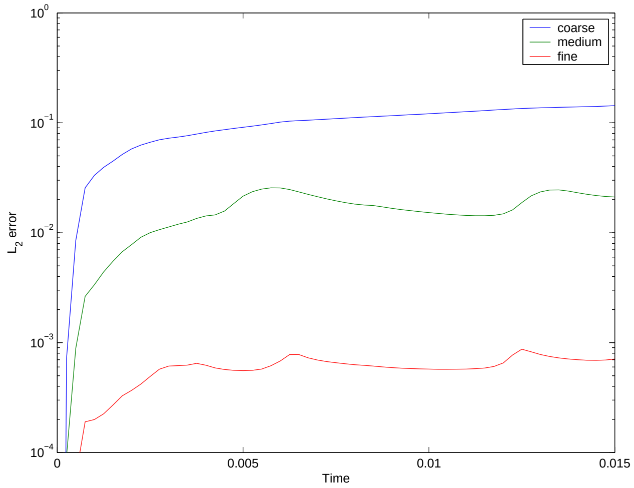

Figure 7.6: L_2 error compared to a reference solution, as a function of time, for a 2D spinodal decomposition approximated on a sequence of uniformly refined discretizations

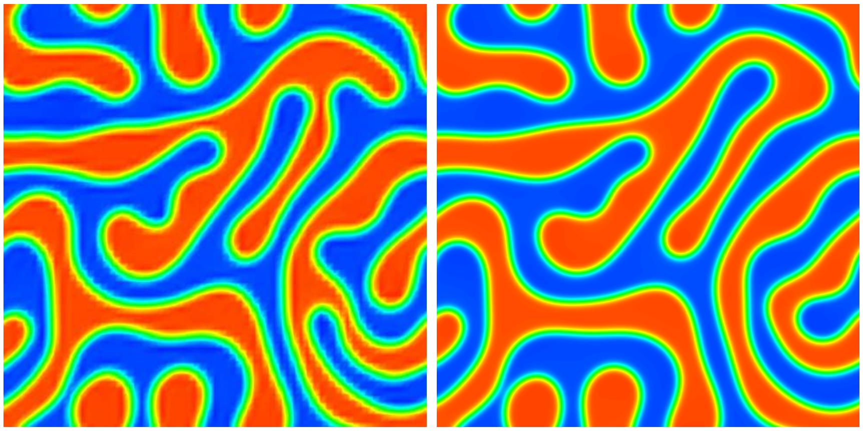

Figure 7.7: Solutions at $t = 0.015$ on h_1 and h_4 grids. The h_2 solution is slightly perturbed from the reference h_4 solution, and the h_3 solution is visually indistinguishable from h_4.

degrees of freedom from region interiors where they can needlessly increase computational complexity. The particular effectiveness of adaptive mesh refinement on the Cahn-Hilliard problem has recently been shown in the finite difference literature [38].

Because of the narrow aspect ratio of the interface regions and because those interfaces move through the domain with advancing time, finding an algorithm to automatically and efficiently track them with element refinement and coarsening is difficult. Initial experiments based on heuristic rules using a Laplacian jump error indicator are capable of correctly finding and refining into Cahn-Hilliard interfaces, but more work is needed before these methods will be responsive enough to reliably track rapidly changing interfaces. Another theoretical concern is the effect of adaptive coarsening on the free energy functional; an otherwise monotonic scheme can see local increases in the free energy functional when an element coarsening is performed.

Figure 7.8 displays one timestep of a numerical experiment using adaptive h refinement. The mesh refinement and coarsening strategy used here attempts to minimize the error at each timestep while maintaining a roughly constant number of active elements, by trading element coarsening for refinement whenever the error indicator suggests that such a trade would improve the final result.

7.11 Surface Patterning

The Cahn-Hilliard equation is useful for simulating phase decomposition in thin film patterning problems, such as are of interest in microelectron-

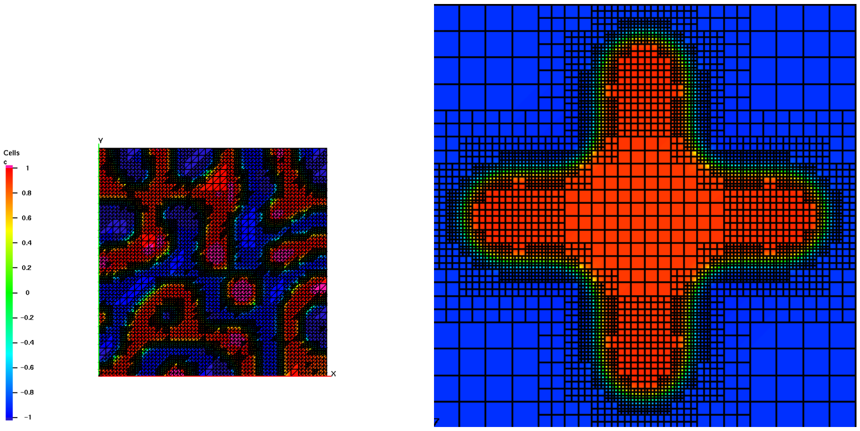

Figure 7.8: Left: Refined Clough-Tocher mesh and solution in an adaptive transient Cahn-Hilliard problem, tracking a random spinodal decomposition at $t = 0.4$. Right: Refined Hermite mesh and Cahn-Hilliard solution, progressing from a cross-shaped initial condition to $t = 0.15$.

ics [113]. Directed phase separation patterns may be realized by coupling the Cahn-Hilliard problem to anisotropic surface stress, bulk fluid flow, electrical fields, or surface chemistry [19,87,96]. An example of thin film pattern formation is pictured in Figure 7.9, the results of a binary polymer phase separation experiment reported by Karim et. al. [79]. We model the effect of the surface coatings by adding a spatially varying bias to the configurational free energy term in the Cahn-Hilliard equation. For a given bias frequency ν and bias amplitude B, define the adjusted local configurational free energy at point (x, y, z) as

$$f_{0b}(c) \equiv f_{0m}(c) + \frac{B}{\pi}\cos(2\pi\nu x)c \tag{7.11.1}$$

where $f_{0m}(c)$ is the unbiased free energy fenction from equation defined as (7.2.3). The biased Cahn-Hilliard equation to be solved remains of the same form as in equation (7.2.9) but with $f_0'(c) \equiv f_{0b}'(c)$. In the parametric studies to follow in Section 7.14, the same bias term is added, but to $f_{0c}(c)$ rather than $f_{0m}(c)$.

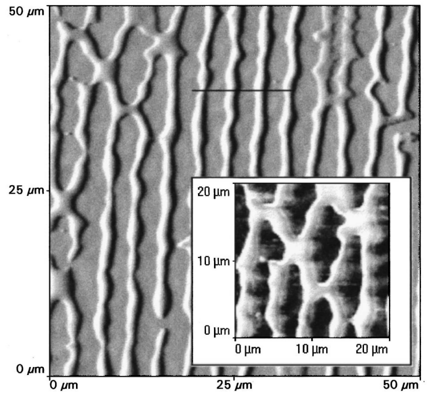

Figure 7.9: An experimental pattern of deuterated polystyrene and polybutadine on a monolayer substrate [79].

Because these phase decomposition experiments are being carried out on very thin film domains, it is natural to simulate the problem in two dimensions. As the thickness of the film becomes negligible compared to the thickness of the diffuse material interfaces and of the pattern characteristic length, the time scale during which the composition becomes homogenous in the vertical direction should become negligable compared to the time scales involved in interface motion in the horizontal directions.

In the numerical experiments pictured, the sinusoidally varying surface bias is given an amplitude B of between 4% and 32% of the total configurational free energy present in a fully blended mixture. A numerical spinodal decomposition is then run for each bias amplitude, with the same pseudorandom initial conditions.

The results for 4% and 8% bias amplitudes, pictured in Figure 7.10, are difficult to distinguish from an isotropic spinodal decomposition result. At 12% and 16% bias, pictured in Figure 7.11, anisotropic effects are clear, but the surface pattern is not duplicated in the adsorbed chemical pattern. At 20% and higher bias, pictured in Figure 7.12, the surface pattern is copied to the adsorbed chemical, with decreasing probability and density of defects.

It is interesting to note that, in these cases, the magnitude of the configurational free energy bias is not large enough to make the final patterned solution a global minimizer of the total free energy of the system. The long interfaces in the achieved pattern result in large values of interfacial free energy in the solution, larger than the reduction of configurational free energy which is enabled when a pattern conforms to the imposed bias. The patterning bias acts, not to force the system to reach a desired global minimum of free energy, but to encourage the system to reach a desired local minimum which is stable with regard to small perturbations. This is one reason why the transient behavior of a patterned phase decomposition is so important.

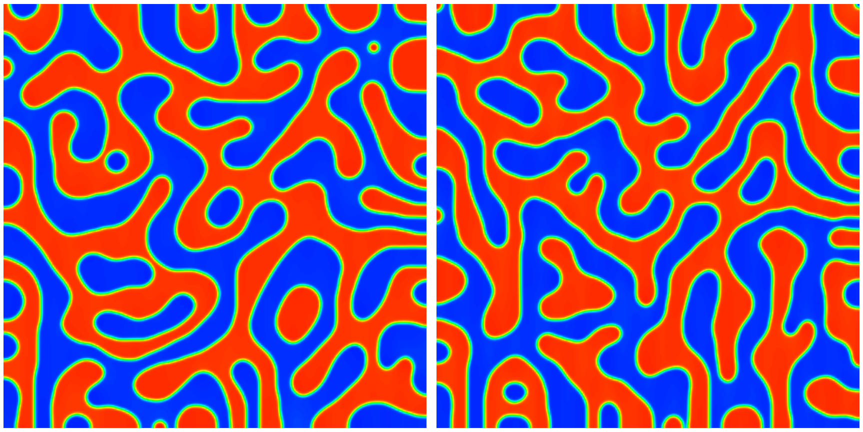

Figure 7.10: Patterned spinodal decomposition simulations run with a 4% and 8% configurational free energy bias amplitudes, at $t = 0.5$.

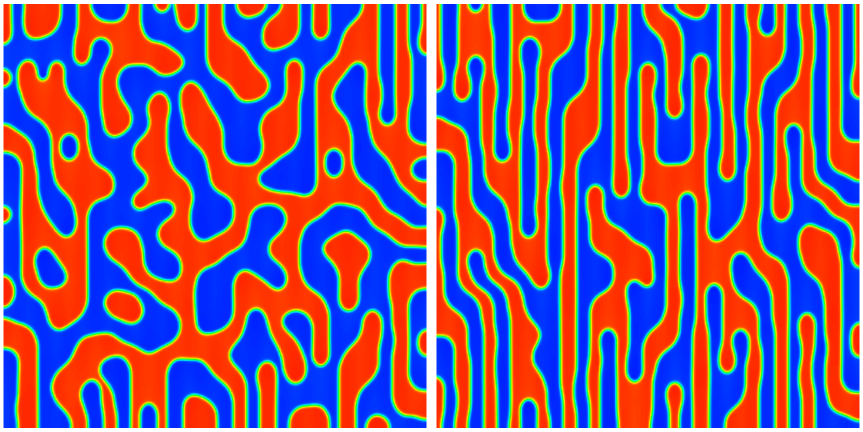

Figure 7.11: Patterned spinodal decomposition simulations run with a 12% and 16% configurational free energy bias amplitudes, at $t = 0.5$.

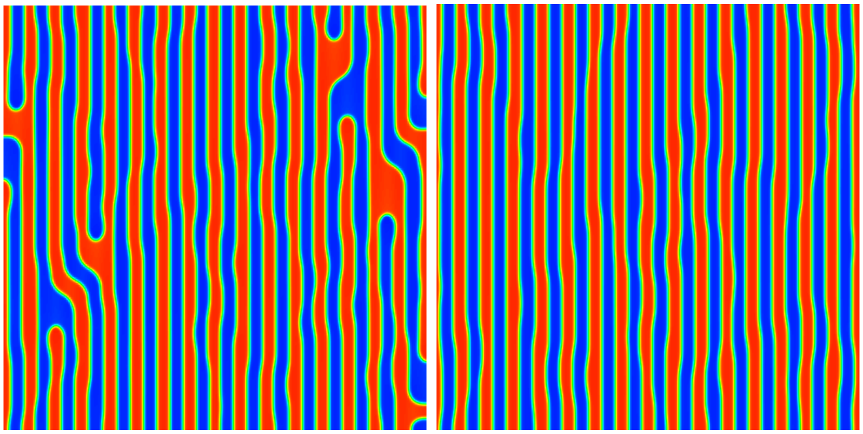

Figure 7.12: Patterned spinodal decomposition simulations run with a 20% and 24% configurational free energy bias amplitudes, at $t = 0.5$.

7.12 3D Thin Film Patterning

If the thin film layer is significantly thicker than the equilibrium Cahn-Hilliard material interface thickness, homogenization in the vertical direction may not be rapid enough for three dimensional behavior to be negligible. We can verify this supposition experimentally. In past studies, surface-directed chemistry has been shown to produce dominant wave vectors normal to a surface which preferentially attracts one of the separating materials in a phase separation process [78, 105]. To minimize the tendency for heterogeneous behavior, we apply a configurational energy bias which is uniform in z, as might be produced by an electric field applied from both sides of the thin film. Any solution variation in the z direction will then be entirely the result of initial perturbations in the solution, as amplified by the initial anti-diffusive behavior of a sudden phase decomposition problem.

In Figure 7.13, we examine a simulation with roughly 4% configurational free energy bias, in a 3D domain where the fluid thickness is equal to the wavelength of the imposed bias. In the early stages of its development, at $t = 0.125$, the solution is still clearly three-dimensional, with numerous horizontal material interfaces and threaded loops of single-phase regions.

Although this convoluted intermediate result would appear to make it less likely that the final state will form a defect-free pattern, in fact the reverse appears to be true. At time $t = 0.5$, the only two remaining defects are shrinking, and the solution interior has retained the imposed bias pattern far more faithfully than in the 2D examples. It seems that the additional connectivity in the fully 3D topology has prevented gaps from forming in the pattern lines in intermediate stages, and thus prevented permanent defects from forming in the long-term results.

In simulations undertaken with a stronger free energy bias, the rapid tendency of the material to flow into the prescribed pattern is evident, overwhelming three dimensional behavior. In Figure 7.14, a 3D simulation is plotted at the same times t in the same domain, but with a four times greater bias amplitude. Some shrinking ripples in the z direction can still be seen in the material interfaces, but the solution has still become essentially two dimensional.

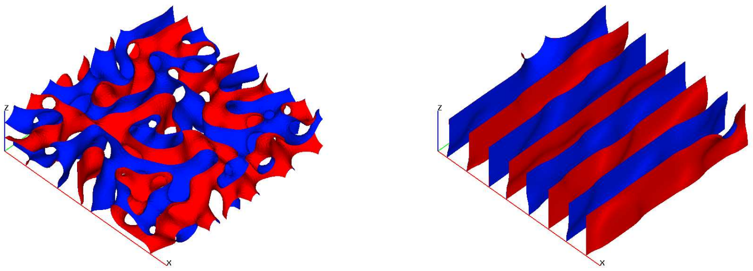

Figure 7.13: Concentration isosurfaces in a patterned 3D spinodal decomposition simulation run with a 4% configurational free energy bias amplitude, at $t = 0.125$ and $t = 0.5$.

The degree to which a three dimensional solution resembles it's two-dimensional counterpart depends on the ratios of the Cahn-Hilliard interface thickness, imposed bias wavelength, and domain thickness. In Figure 7.15, we see a simulation conducted with the same parameters as in Figure 7.13 but with one fourth of the domain thickness. The intermediate results show some three dimensional behavior, with regions of vertically layered material, but large areas of the domain are already essentially two dimensional. Unlike

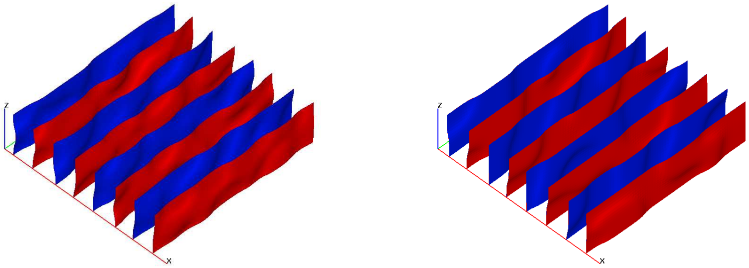

Figure 7.14: Concentration isosurfaces in a patterned 3D spinodal decomposition simulation run with a 16% configurational free energy bias amplitude, at $t = 0.125$ and $t = 0.5$.

the three dimensional results on thick domains, here material regions form at least a dozen disconnected sets rather than two. Some of the gaps which exist at this intermediate stage are stable and growing at $t = 0.5$, leading toward a final solution with three short-circuits and one break in the pattern.

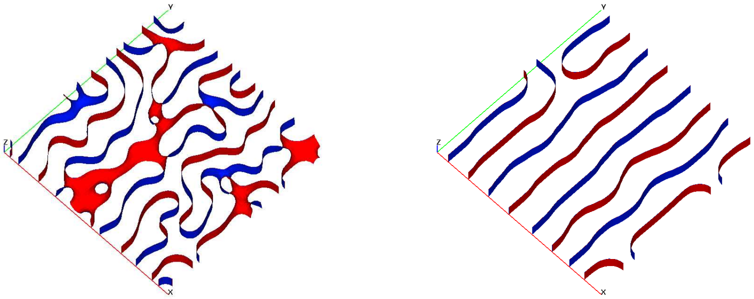

Figure 7.15: Concentration isosurfaces in a patterned thin 3D spinodal decomposition simulation run with a 4% configurational free energy bias amplitude, at $t = 0.125$ and $t = 0.5$.

7.13 Solution Postprocessing

For the spinodal phase decomposition problem, examination of the full 2D or 3D simulation at each timestep can be overwhelming due to the large quantity of data involved. To make it easier to draw conclusions about a simulation result, this data can be distilled into a few simple scalar results which describe some of the important features of the solution at a particular timestep.

First and most obviously, for the patterned phase decomposition problem it is important to know how well the solution conforms to the pattern at each timestep. The added bias function which causes a pattern to develop will not be the same as the desired pattern, due to the nonlinear antidiffusive properties of the Cahn-Hilliard function, so it is impossible to simply take a function space norm of the difference between the concentration function and some simple transformation of the pattern function. Instead, let us define a "defect count" for a concentration field as follows: Along each manifold where the bias function is at a local maximum or minimum, a correctly patterned concentration function should take values near the corresponding spinodal value, and a defect will be said to exist if anywhere along that manifold the concentration function is nearer to the wrong spinodal value; i.e. if $c < 0.5$ along manifolds where the bias is intended to generate a high concentration or if $c > 0.5$ where a low concentration is intended. For the sinusoidal bias function used to generate line patterns, for example, a bias with 8 peaks has 16 manifolds where a defect may occur, either due to features like "circuit breaks" where a hole exists in a high concentration line or like "short circuits" where two separate lines are bridged by a region of high instead of low concentration. The defect count is then an integer value (in this example between 0 and 16) reflecting how many pattern manifolds exhibit a defect. This definition is insensitive to slight variations in the final pattern, and counts only those deviations important enough to cause a topological change.

As a more sensitive means of exploring Cahn-Hilliard coarsening in both the isotropic spinodal decomposition and anisotropic phase decomposition problems, we examine the behavior of the correlation lengths. Coarsening of solution length scales is a common object of investigation in unpatterned phase separation problems [66, 122], and the effects of patterning on this process are of interest. For a concentration function $c(\vec{x})$, we define the corresponding correlation function $r(\vec{y})$ to be

$$r(\vec{y}) \equiv < c(\vec{x})c(\vec{x}+\vec{y}) > - < c(\vec{x}) >^2 \tag{7.13.1}$$

where $< f(\vec{x}) >$ denotes the average of an integrable function f over all $\vec{x}$ in the domain Ω,

$$< f(\vec{x}) > \equiv \frac{\int_\Omega f(\vec{x})\, d\Omega}{\int_\Omega 1\, d\Omega} \tag{7.13.2}$$

The correlation function is effectively a measure of how related two points in the domain are likely to be if they are separated by a distance $\vec{y}$. It takes a maximum at $\vec{y} = \vec{0}$ and generally decays with distance. In some analyses of spinodal decomposition problems, an analytic form of the expected correlation function can be found and experimental data fitted to it; the "correlation length" in this case simply refers to a length parameter in the fitted correlation function. It is dangerous to do data fitting with correlation functions on periodic or symmetry-bounded domains where $\vec{y}$ may be a significant fraction of the domain size, however, because in these cases $r(\vec{y})$ will also be periodic in space and will never completely decay to zero. To avoid this problem, and to define a correlation length in patterned decomposition scenarios where we do not want to make *a priori* assumptions about the form of the correlation function, we will define the correlation length l in the direction of unit vector $\vec{y}$ to be the distance at which $r(l\vec{y}) = 0.5r(\vec{0})$, as is done in [105]. In numerical calculations, the correlation function r in each direction will be evaluated on a uniform grid, and the correlation lengths will be calculated from a piecewise-linear interpolant of the grid point values of r.

For the unpatterned spinodal decomposition function, correlation lengths and average domain lengths have been of particular interest as a means of investigating the dynamic scaling of the system. For models with a conserved order parameter such as the concentration in the Cahn-Hilliard equation, the expected growth of intermediate length scales L over time t is the Lifshitz-Slyozov law, $L(t) \propto t^{1/3}$. In numerical and experimental results, such scale growth has been observed when L is intermediate between the lengths of the equilibrium interface thickness and the domain size [122, 123]. In patterned phase decomposition studies, we can expect the pattern to strongly influence the length scales involved; a phase decomposition solution which correctly matches a line pattern, for example, will have an infinite length scale in the direction parallel to the pattern and will have a length scale proportional to the wavelength in the direction perpendicular to the pattern.

In the parametric studies in Section 7.14, the defect count will be of the most importance as a final quantity of interest for patterning processes, while the correlation lengths are useful as a way to examine the development of solution coarsening and anisotropy in earlier stages of the transient simulation. The total free energy of the systems, as well as their rates of change, are also transient quantities of interest to be examined.

7.14 Parametric Studies

The behavior of the concentration function in the transient patterned spinodal decomposition problem depends on the set of physical parameters defining the problem. However, the exact nature of these dependencies can not often be predicted in advance from *a priori* arguments and analytic examination of the governing equations. To try to gain a better understanding of the complex processes at work here, we use parametric numerical studies to compare solutions generated from differing initial conditions or with differing physical constants.

Complicating this task is the fact that the solution to a spinodal decomposition problem depends not only on the physical parameters of the problem but also on the solution's particular initial perturbation away from the spinodal point. Because of the instability of the homogeneous initial concentration solution in this problem, two Cahn-Hilliard solutions can differ significantly in their transient evolution, based solely on the particular features of the initial perturbation of each solution away from the unstable point. The question at hand, then, is not just "How do differing fixed parameters cause the developing Cahn-Hilliard solution to vary?" The question is also one of uncertainty propagation: "How much uncertain variance can be expected to arise from the aleatoric uncertainty in the perturbations in the initial solution?"

To investigate both aspects of this problem, our code employs the Monte Carlo Finite Element Method. Rather than wait for numerical noise to create an initial perturbation, we generate a perturbation in the initial conditions using a pseudorandom number generator (PRNG), and we repeat simulations with the same physical parameters but with different "seed" values initializing the PRNG. Specifically, for each Monte Carlo sample with a given initial perturbation amplitude A and average initial concentration c_0, each node n of a grid is assigned the value $c_0 + (2\theta_n - 1)A$, where θ_n is an independent pseudorandom number between 0 and 1. This field is generated for an initial grid of Hermite tensor product cells of size $h = 1/32$, and then projected onto the more refined mesh used for the finite element simulation. From this point the finite element simulation is deterministic; i.e. there is no further stochastic noise such as might be found in a Cahn-Hilliard-Cook model [44]. Analysis of functionals of the output data may then allow the examination of the mean, variance, etc. from this random sampling of the perturbation space. Even a few samples is sufficient in many of the following results to allow parametric variance to be distinguished clearly from uncertainty variance.

Initial Perturbation Magnitude Because this method depends on the proposition that an uncertain initial perturbation can be properly approximated by a deliberate perturbation of a small but finite magnitude, it is natural to first consider what effect the size of that perturbation has on the final result. In Figures 7.16 and 7.17, the effect of the perturbation strength on a typical simulation is displayed. For this simulation a line pattern bias of frequency 4 and amplitude 0.04 is imposed, and the Cahn-Hilliard equation is run with physical parameters $NkT = 0.6$, $N\omega == 1.8$, and $\epsilon_c = 0.01$, on the unit domain.

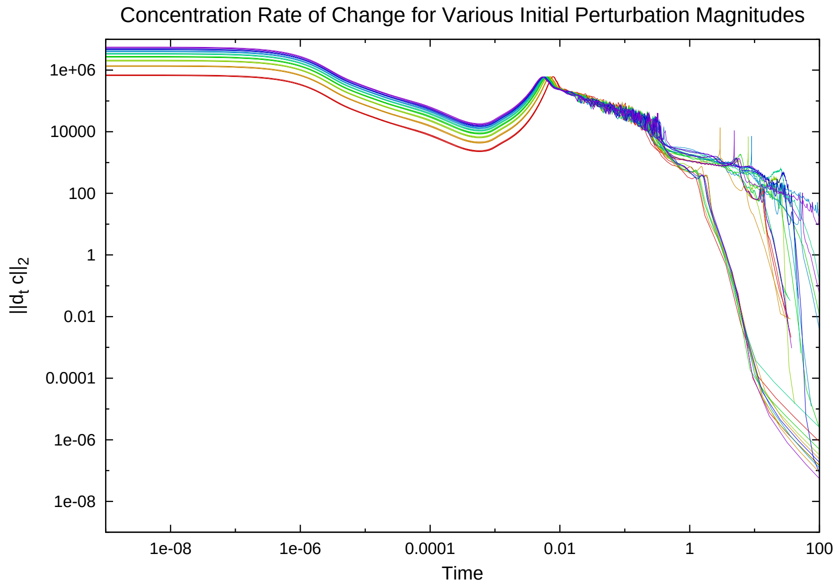

Figure 7.16: The rate of change of the concentration (in H^2 norm) plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple).

In Figure 7.16, the dependent variable being graphed is the rate of change $\frac{\partial ||c||_{H^2(\Omega)}}{\partial t}$. The axes here are both in log scale due to the extreme differences in timescales over the course of the simulation. The early time behavior here clearly depends far more on the perturbation magnitude than on the random details of that perturbation, with smaller perturbations resulting in much slower early solution development for $t < 0.001$. This slower development leads to a delay in the final separation phase, as can be seen by the lagging peaks around $t < 0.01$. However, once regions with concentrations near stable single-phase values begin to form, the differences between simulations with differing initial perturbation amplitudes largely vanish. In the late-time development of the simulation, for instance, the solutions are clearly segregated into four groups corresponding to the four Monte Carlo seeds used, and solutions of differing initial perturbation magnitude resemble each other more closely than they resemble solutions of differing initial perturbation "shape".

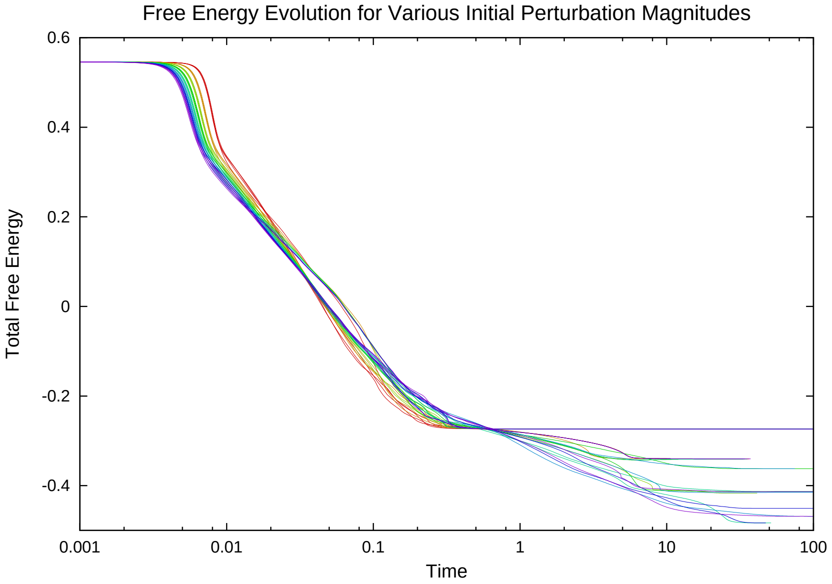

Figure 7.17: The free energy of the Cahn-Hilliard system plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple).

In Figure 7.17, a plot of the total free energy functional with respect to time for the same simulation, the reason for this insensitivity of long-term behavior seems to be clearer. Although the lower magnitude perturbation solutions seem to initially lag the higher magnitude solutions, the lower magnitude solutions "catch up" faster in general.

Figure 7.18 confirms this late-stage insensitivity. Behavior at $t = 1$

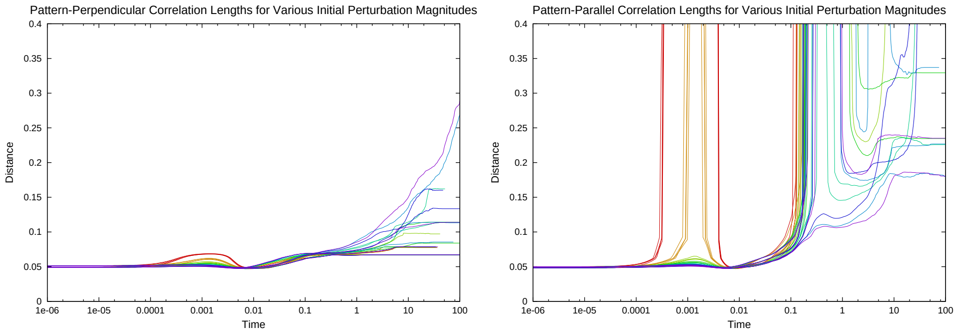

Figure 7.18: The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple).

and beyond appears uncorrelated with initial perturbation magnitude. Before $t = 0.01$, however, a startling effect of low perturbation magnitudes can be seen: the correlation length in the direction of the line pattern briefly jumps to infinity, as if the concentration solution had become completely aligned with the pattern just as phase separation had barely begun!

More details on this effect of initial perturbation amplitude can be seen in Figure 7.19, a plot of the number of pattern defects with respect to time for this simulation. At the frequency of the imposed pattern on this simulation, up to 8 defects can be seen. The majority of the simulations see the defect count eventually drop to 0 at these parameter settings, but for the simulation with the smallest initial perturbation, magnitude 0.001, the defect count also drops from 8 to 1-3 during the very start of the simulation by $t = 0.001$, only to rise back to 8 before $t = 0.005$. Examination of the detailed simulation results, such as the example in Figure 7.20 indicates what is really happening here: the initial drop in defect count precedes the full phase decomposition.

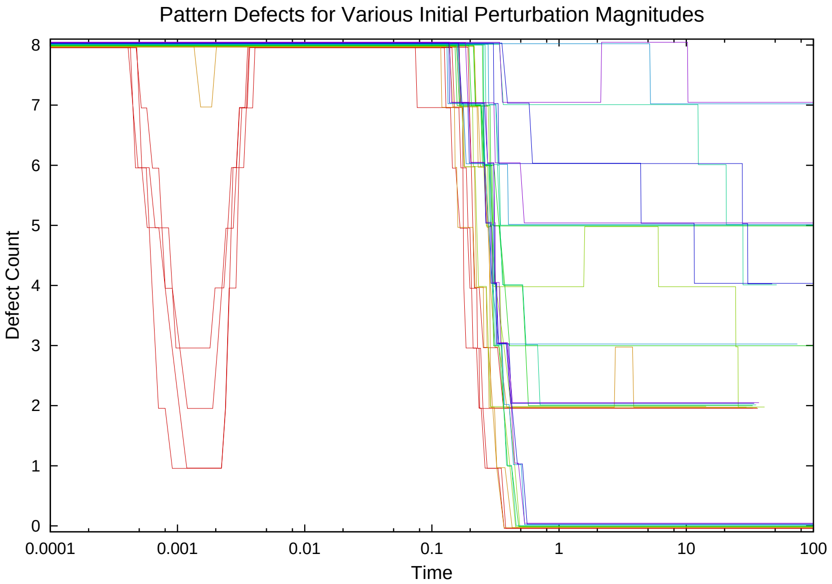

Figure 7.19: The defect count of the Cahn-Hilliard system plotted against time, for initial perturbation magnitudes ranging from 0.001 (red) to 0.008 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph.

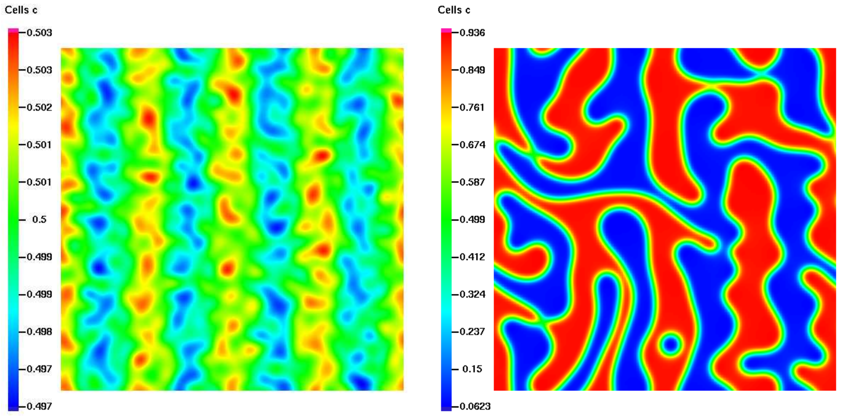

Figure 7.20: Plots of the concentration in a patterned phase decomposition, with an initial perturbation magnitude of 0.001, at time $t \approx 0.0008$ on the left and $t \approx 0.1$ on the right. The early-stage solution appears to fit the imposed pattern, but full decomposition has not yet occurred.

Because a “defect” is defined to be a greater than 0.5 concentration along the manifolds where a low concentration is desired or a lesser than 0.5 concentration where a high concentration is desired, a perturbation which is aligned with the imposed pattern bias is detected as a correct pattern even when that perturbation has not yet grown so far that the concentrations along the pattern have not approached the double-well minima. The bias of the imposed pattern briefly causes the concentration function to resemble that pattern, but when the higher frequency random perturbation grows then the pattern is again obscured, only to be recreated a second time in the late-time simulation. We only observe this effect for the smallest initial perturbation amplitude; at larger amplitudes, the brief pattern alignment is overwhelmed by even the small random variations in the initial conditions.

Film Thickness The results of the three-dimensional pattern self-assembly experiments in Section 7.12 suggested two conclusions. First, it appeared that for sufficiently thin films the concentration variable rapidly homogenized in the vertical direction, so that even the three-dimensional phase decomposition problem becomes essentially two-dimensional. Second, it was found that thicker films replicated an imposed pattern bias more reliably than thinner films, due to the increased topological complexity of the initial phase separation which left material connections across gaps that otherwise might develop into pattern defects.

To examine these conclusions in more detail, we simulate the same physical parameters as in the previous study, but on a three-dimensional domain with thickness ranging from 0.0625 to 0.25. Periodic boundary conditions are still imposed on the four sides of the square domain, and symmetry (no-flux) boundary conditions are applied on the top and bottom. Because these systems can involve over half a million degrees of freedom in 3D, the code here is run on 256 processors rather than our typical 16.

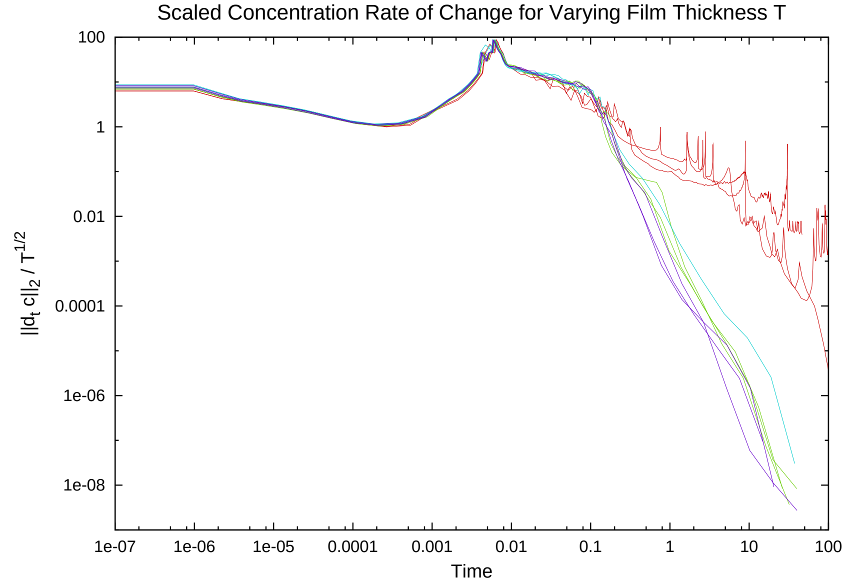

Figure 7.21: The rate of change of the concentration (in a scaled H^2 norm) plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple).

Figure 7.21, which plots scaled Hilbert norms of the time rate of change of the concentration solution, appears to show results all consistent with the earlier 3D behavior: the most interesting late-stage behavior occurs only on the thinnest domain, with thickness $T = 0.0625$. For thicker domains, the average rate of change of the system is greater up until approximately time $t = 0.1$, after which point the thick solutions appear to smoothly approach a steady state.

Because of the domain-dependent definitions of Hilbert norms and of the total energy integral (that is, the energy norm squared), we must scale these outputs to arrive at directly comparable numbers in Figures 7.21 and 7.22. For a domain of thickness T we scale the H^2 norm by dividing by $\sqrt{T}$, and we scale the energy functional by dividing by T.

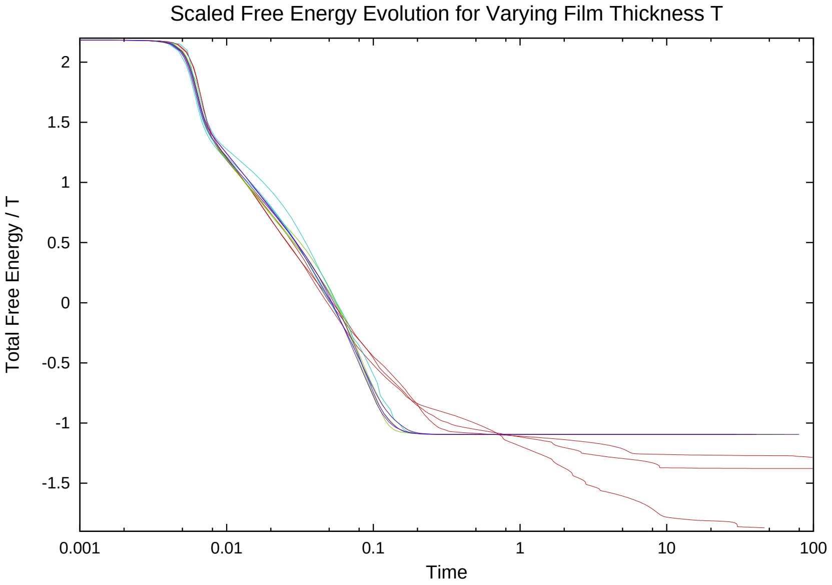

Figure 7.22: The free energy per unit volume of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple).

Figure 7.22, a plot of total free energy per unit volume, also confirms this behavior. The free energy in the thick domains falls more rapidly after $t = 0.1$, and quickly reaches a steady state.

In the correlation length plots in Figures 7.23 and 7.24, it is made clear that the underlying cause of these effects is due to the different reactions of thicker and thinner systems under the patterned bias. All of the thick sim-

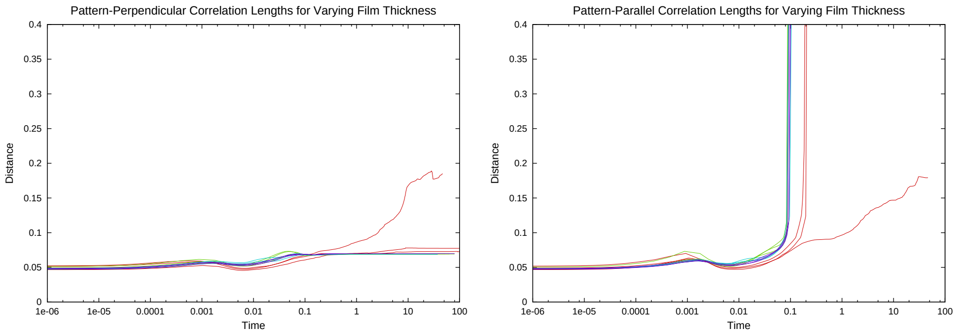

Figure 7.23: The horizontal and vertical (in-film directions) correlation lengths of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple).

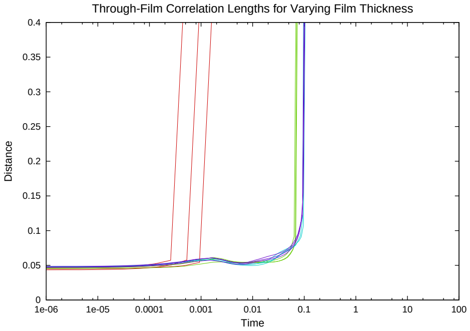

Figure 7.24: The thickness (through-film) direction correlation lengths of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple).

ulations see pattern-aligned correlation lengths grow to infinity by $t = 0.1$, whereas in the thinnest simulation this growth occurs later or not at all. Even for the instances where the thinnest simulations eventually align with the pattern imposed-direction, they do not necessarily adopt the desired pattern wavelength, as can be observed in the differing final pattern-perpendicular correlation lengths reached. A unique feature of the three-dimensional simulations is the existence of a correlation length in the third dimension, the through-film direction. In Figure 7.24 it can be seen that homogenization in this direction, although an inevitable feature of the end-state in any of the systems being studied, occurs at a time two orders of magnitude earlier in the thinnest domains, well before other differences in the simulation behavior become clear. It is clear that this early homogenization is driving the results.

The final quantity of interest, the evolving defect count, can be seen plotted in Figure 7.25. As expected from earlier results, the defect count is reduced to zero in sufficiently thick simulations, but for the nearly two-dimensional thin domain results defects have a high likelihood of being stuck in place in the final steady-state solution. This result confirms that our two-dimensional studies to follow are conservative in that sense, simulating the least reliable patterning behavior of very thin films.

Domain Size When solving a patterned phase decomposition problem, by necessity we do not solve for the concentration solution on the full physical domain of the problem. Just as engineering interest in the unpatterned Cahn-Hilliard problem stems from the desire to understand grain-scale coarsening behavior in macroscale systems, interest in directed pattern self-assembly typ-

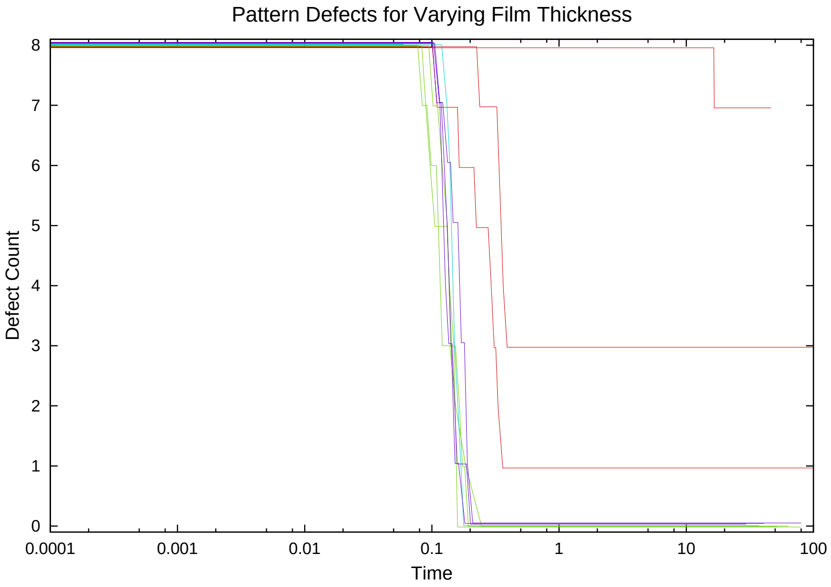

Figure 7.25: The defect count of the Cahn-Hilliard system plotted against time, for domain thicknesses ranging from 0.0625 (red) to 0.25 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph.

ically focuses on the ability to assemble fine scale patterns in large domains. The domain size in these problems typically vastly exceeds both the pattern length scale and the interfacial layer thickness. A simulation using a mesh of sufficient size and resolution to resolve all solution features throughout the entire domain would be intractable, so instead the bulk properties of these systems are investigated by artificially truncating the domain, then applying symmetry or periodic boundary conditions to approximate an infinite domain. However, as the solution to a Cahn-Hilliard problem coarsens, it is possible for self-interaction with periodic boundary conditions or boundary pollution from symmetry boundary conditions to have a noticeable effect on the system behavior.

To ensure that our simulations are being run on sufficiently large domains to make the effects of domain size negligible, a series of simulations is run with the same physical parameters as used in the previous studies, but on domains of size 1×1, 2×2, and 4×4. In Figures 7.26 and 7.27, it is quickly seen that the early development of the solution, as measured by the H^2 norm of the rate of change or by the total free energy of the system, is essentially unaffected by domain size. Note that we must again scale the Hilbert norms and the total energy integral to derive directly comparable output functionals. For a square domain of size $L \times L$ we divide the former by L and the latter by L^2.

One domain size dependent change in behavior can be seen: on the smallest domains, the solution is likely to be forced to an earlier steady state, as the solution change rate plummets and the system free energy levels off. The

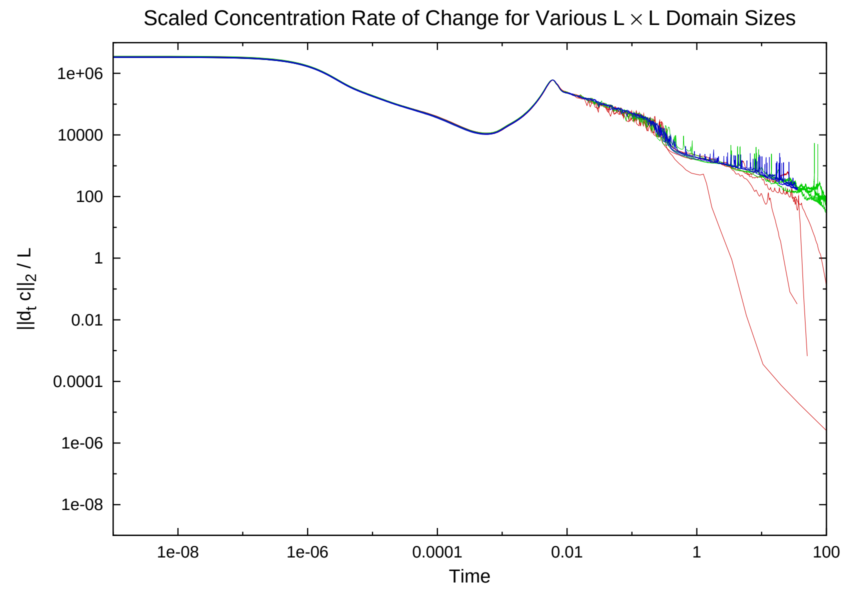

Figure 7.26: The rate of change of the concentration (in a scaled H^2 norm) plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue).

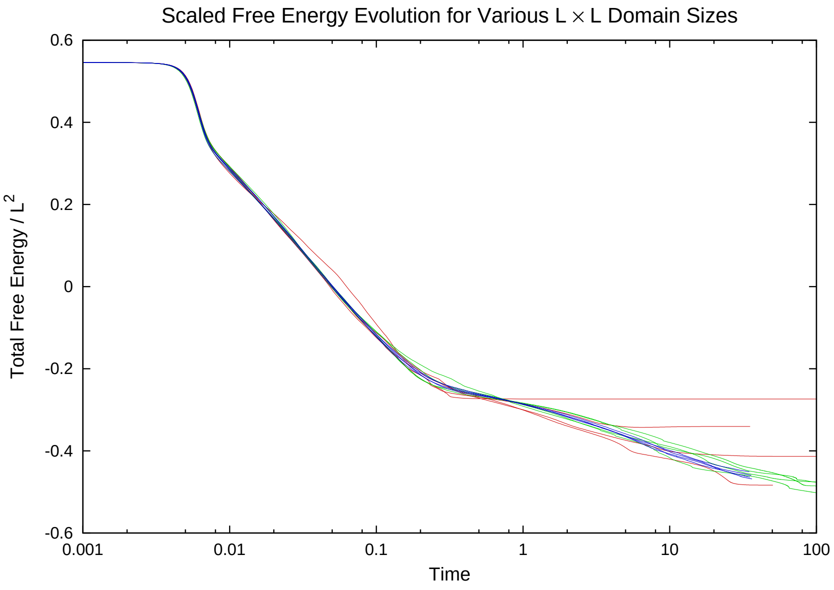

Figure 7.27: The free energy per unit area of the Cahn-Hilliard system plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue).

specific time at which this occurs depends on the particular initial conditions of each Monte Carlo iterate.

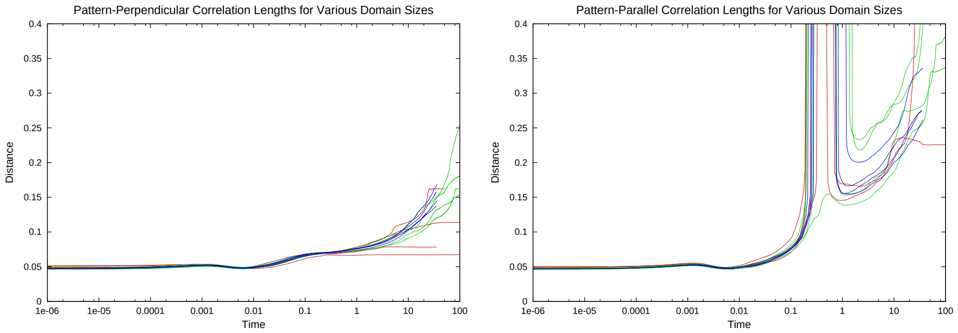

Figure 7.28: The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue).

The correlation lengths in Figure 7.28 require no scaling; the definition of correlation length in section 7.13 produces results which are directly comparable from one domain size to another. The correlation length results also show behavior dependent on domain size in the 1×1 simulations, where the pattern-perpendicular correlation lengths stop growing at relatively early times. The small domain size appears to reinforce the imposed pattern bias, which acts to limit the growth of length scales across the pattern lines.

The question of how to scale defect counts from one domain size to the next has no simple answer. For solutions in which defects are sparse, increasing the length and width of the domain by L would be expected to increase the number of pattern defects by nearly L^2, as the number of defects per unit area remained constant. For solutions in which defects are common, however, defects can "overlap", when two defects which break the same line in

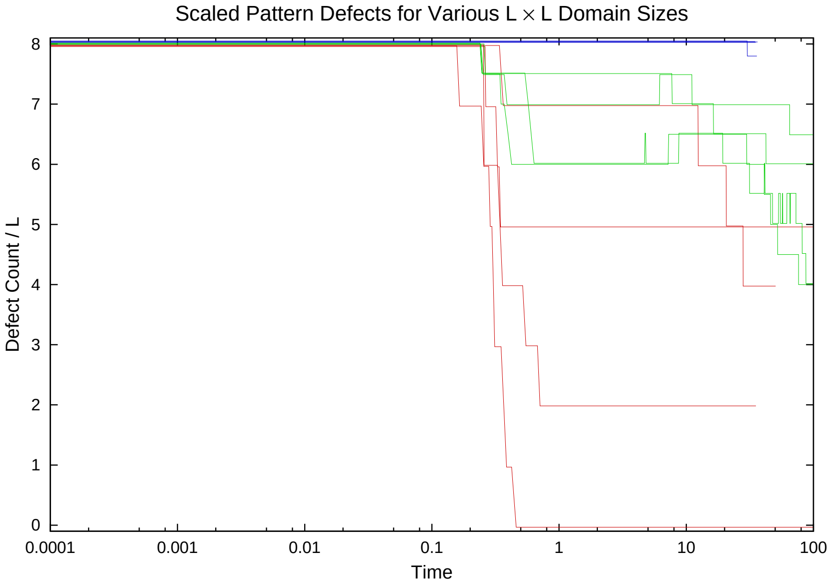

Figure 7.29: The defect count per unit length of the Cahn-Hilliard system plotted against time, for domain sizes ranging from 1×1 (red) to 4×4 (blue).

the imposed pattern are only counted as a single broken line, i.e. as only one defect. In this case, the defect count for a domain whose width in the pattern-perpendicular direction was increased by a factor of L would be expected to increase only proportionally with L, not L^2. For these studies, typically conducted with parameters which lead to a dense defect count as the initial perturbation grows but a sparse defect count as the solution aligns with the pattern bias, neither scaling remains valid throughout the entire course of a simulation.

Regardless of how defect counts are scaled, one facet of the domain size dependence of simulation behavior is clear: in the smallest domain, the defect count becomes fixed much earlier in the simulations, as the final topology of the solution becomes fixed.

Gradient Coefficient Physical values for NkT and $N\omega$ for physical phase separation problems are relatively common in the Cahn-Hilliard literature, and typical values are applied in these numerical studies. However, physical values for the interfacial energy term coefficient are harder to come by. In fact, the gradient coefficient ϵ_c is often artificially increased, to make the diffuse interfaces wider and therefore more tractable for numerical simulation.

Although ϵ_c may be safely artifically adjusted when it is a sufficiently small fraction of the physical length scales of interest, for the patterned spinodal decomposition problem, the choice of ϵ_c may become very physically significant due to non-local interfacial effects which may act to repair pattern defects. To illustrate this point, using the other fixed parameters as in the above studies, we vary the gradient coefficient from $\epsilon_c = 0.01$ to $\epsilon_c = 0.04$ to examine its effects on the results.

In Figure 7.30, the importance of the gradient coefficient becomes obvious. The behavior both of the early phase decomposition and the later interfacial coarsening depends strongly on the interfacial energy parameter.

In Figure 7.31, it can be seen that, although for these parameters the final state of the system is sensitive to the particular initial perturbation of the Monte Carlo sample, interesting behavior differences distinguished purely by the interfacial terms are dominant.

Figures 7.32 and 7.33 identify some of the specific behavior differences that can be seen with increasing gradient coefficients. In the red lines plotting defect counts for $\epsilon_c = 0.01$, the coefficient used in other parameter studies in this work, typical behavior is seen: a full defect count at early time, followed

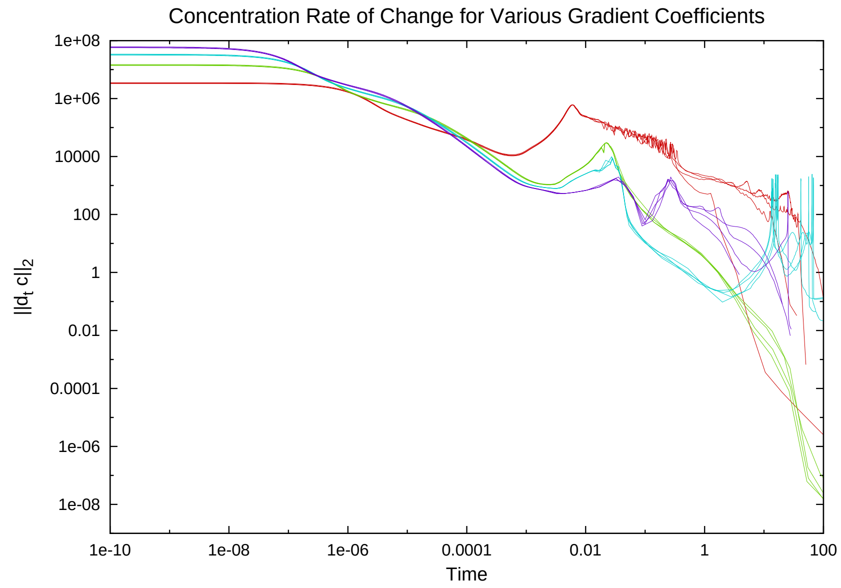

Figure 7.30: The rate of change of the concentration (in H^2 norm) plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple).

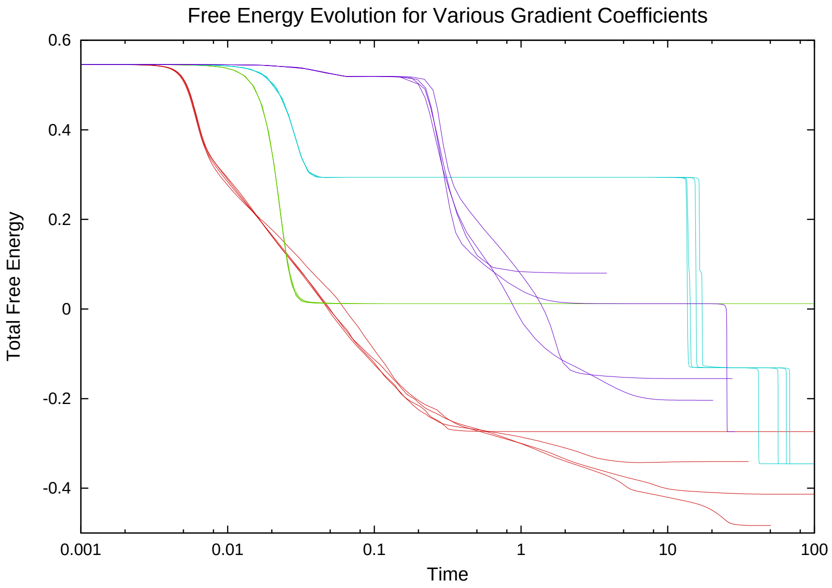

Figure 7.31: The free energy of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple).

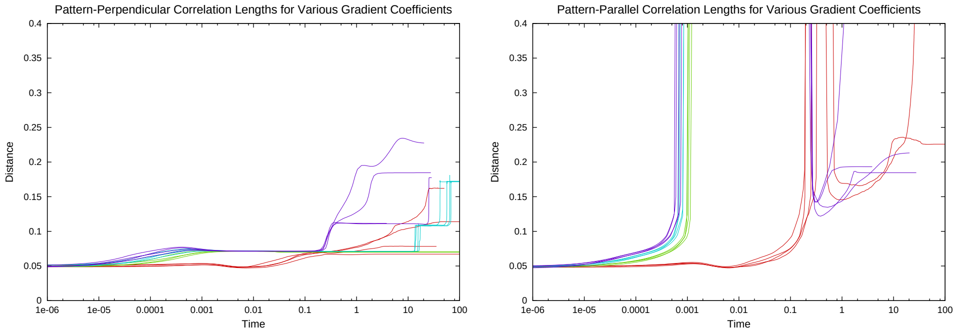

Figure 7.32: The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple).

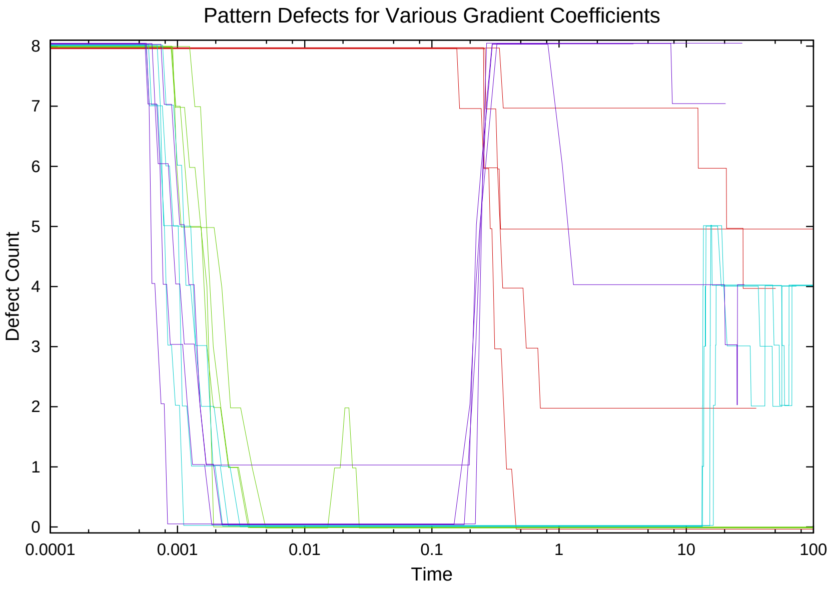

Figure 7.33: The defect count of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.01 (red) to 0.04 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph.

by a growth of self-correlation lengths, gradual defect reduction, and eventual alignment of correlation lengths with the pattern direction as Cahn-Hilliard coarsening completes. For higher gradient coefficients, surprisingly, the pattern defects appear to be completely eliminated in the early stages of phase separation, only to potentially reappear in later stages of the simulations. Examination of the final steady-state values for these simulations reveals the reason why: as ϵ_c increases, so does the equilibrium diffuse interface thickness of the simulation. As the diffuse interface thickness becomes a significant fraction of the pattern bias wavelength, the desired pattern becomes unstable. Even if in the early stages of coarsening a material stripe forms around each desired line in the pattern, adjacent stripes tend to merge, segregating material into a pattern with the desired directionality but an undesirably low frequency. For the largest gradient coefficient tested, $\epsilon_c = 0.04$, this instability is so pronounced that it even ruins the previously attained vertical symmetry of the solution, as seen in the graph of pattern-parallel correlation lengths. Concentration solution plots in Figure 7.34 demonstrate the detailed evolution of this instability. In the initial pattern-aligned, partly decomposed solution at left, the thickness of the interfacial layer is so great that two points in neighboring pattern lines interact and attract. The long-distance interactions from the merged shape affect the remaining material zones in the domain, and the combined zones eventually form a pattern which is aligned with the imposed bias but which has only half of the bias frequency.

Average Concentration Because the range of unstable concentration values is wide for the ratio of NkT to $N\omega$ used in these examples, it is possible

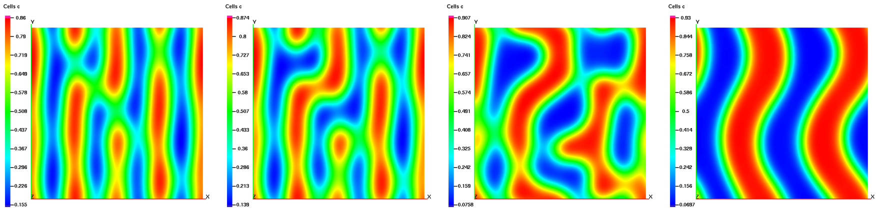

Figure 7.34: The development of an instability in a patterned spinodal decomposition problem with $\epsilon_c = 0.04$.

to get spinodal decomposition behavior from a wide range of initial concentrations. By varying the initial concentration from .25 to .5, we can see how the particular choice of initial concentration affects the final solution.

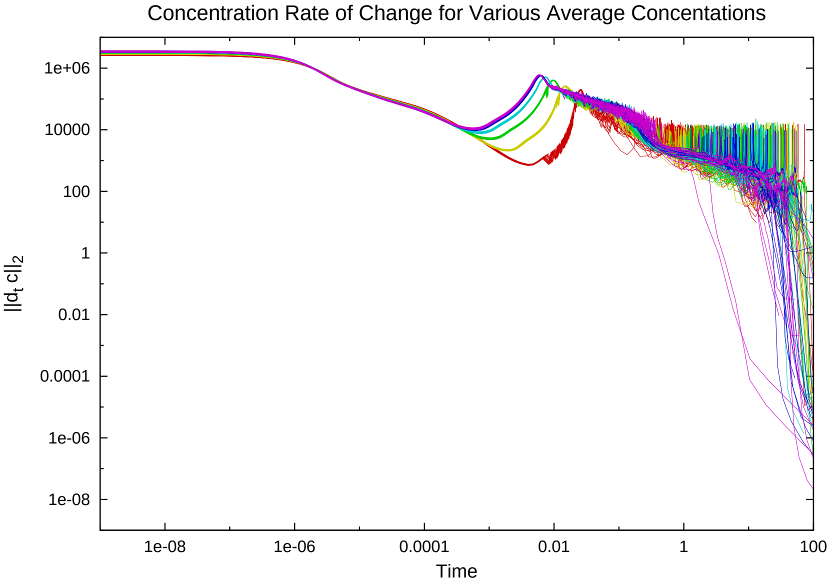

Figure 7.35: The rate of change of the concentration (in H^2 norm) plotted against time, for average concentrations ranging from 0.25 (red) to 0.5 (purple).

In Figure 7.35, the average concentration value has a clear impact on the early stage of phase decomposition. Binary mixtures with a more balanced average concentration tend to show earlier, faster decomposition than mixtures which are predominantly composed of one species or the other.

The effect of average concentration on late-time behavior, however, is not as clear here. As the "forest" of spikes from different Monte Carlo samples makes clear, the simulation parameters here make the particular timing of a given decomposition very sensitive to the shape of the initial perturbation determining that decomposition.

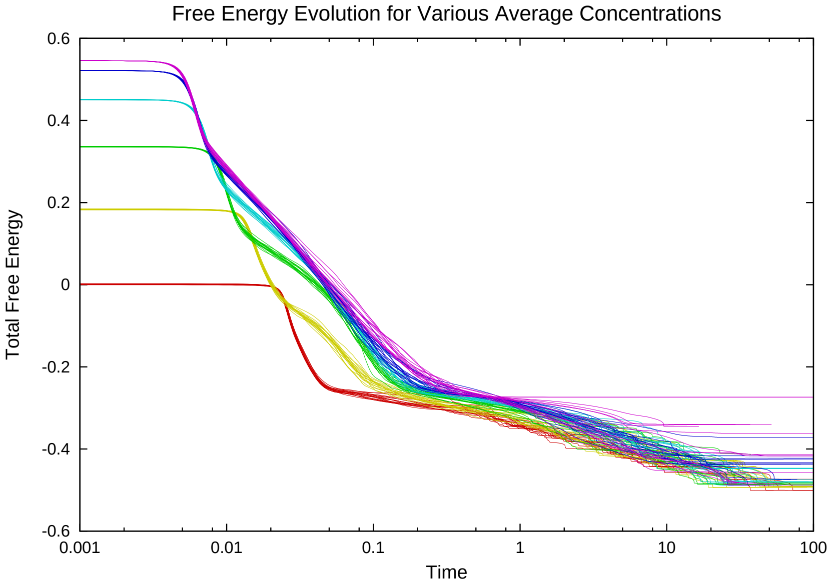

Figure 7.36: The free energy of the Cahn-Hilliard system plotted against time, for gradient coefficients ranging from 0.25 (red) to 0.5 (purple).

The reason for the differences in early phase separation behavior with different average concentrations is made clear by the graph of free energy in Figure 7.36. More balanced average concentrations begin closer to the peak in between the double wells in the chemical free energy diagram, and that free energy drives a faster decomposition of the mixture. In the late phases of the simulations, however, most material is in regions whose concentration is close to one or another of the stable binodal points, and because our chemical free energy function is symmetric except for the imposed patterned bias, the binodal points have roughly the same free energy densities, so the total free energy of the system is not affected by the question of whether or not the two stable regions include the same amount of material.

Bias Amplitude The most straightforward way to impose a directed pattern on self-assembling material more reliably is simply to increase the strength of the imposed pattern. We simulate patterns of increasing strength by varying the amplitude of the spatially varying bias term added to the chemical free energy equation. In this experiment, the bias amplitude is varied from 0 to 0.08, while other parameters are kept the same as in the preceding simulations.

In Figure 7.37, the rate of change of solutions with different patterned bias amplitudes is plotted. For low bias amplitudes, the behavior is familiar: gradually decreasing rates of change, intermittently punctuated by the short spikes of topological change events. For the strongest bias amplitudes, the simulation behavior differs qualitatively; the obviously stochastic early-time behavior is followed by a more gradual, relatively smooth, yet much steeper reduction in the rate of solution change. By forcing the solution topology to resemble the pattern topology, these largest imposed biases restrict the possibility of late-term topological changes, and so the late-term development of the concentration function is a relatively placid straightening of the pattern interfaces.

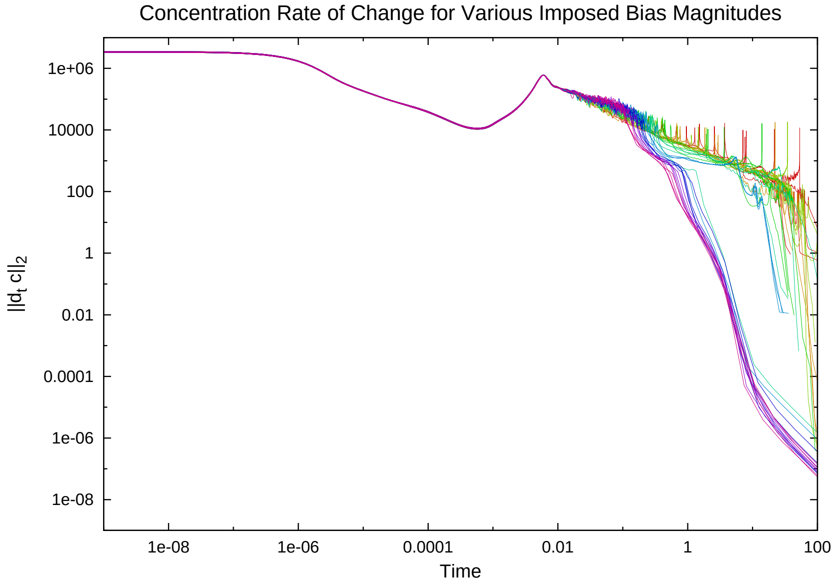

Figure 7.37: The rate of change of the concentration (in H^2 norm) plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple).

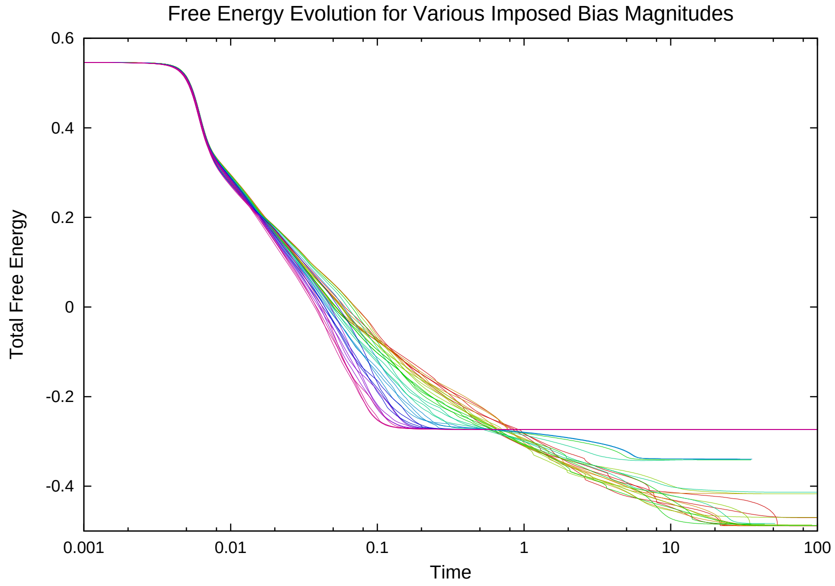

Figure 7.38: The free energy of the Cahn-Hilliard system plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple).

The effect of these stronger biases can also be seen in the effect of pattern bias strength on global free energy, plotted in Figure 7.38. The early-time development of the system is relatively insensitive to random perturbation details: in general, the stronger the imposed pattern bias, the faster the free energy of the system is reduced.

One other notable feature of the patterned Cahn-Hilliard problem is illustrated again by Figure 7.38: the global free energy of the solutions conforming to imposed patterns is actually higher than the free energy of the solutions which break those patterns and develop defects. Although the pattern bias allows for reduced configurational free energy in the phases of the system which conform to the imposed bias, the many long straight interfaces in the patterned solution inevitably lead to a higher interfacial free energy. The goal of patterning in a Cahn-Hilliard process is not to create a desired end state for the problem which is a global minimizer of the total free energy, it is to create desired states which are stable local minima of that free energy functional.

Pattern bias amplitude has a straightforward effect on the late-stage development of solution correlation lengths. Stronger patterning bias tends to control the growth of correlation lengths in the direction perpendicular to the pattern lines, as coarsening is discouraged across pattern boundaries, and stronger patterning bias encourages more rapid growth of correlation lengths in the direction parallel to the pattern lines, as material of the same phase is drawn preferentially into the aligned matching bias regions of the pattern. Surprisingly, it can also be seen in Figure 7.39 that pattern bias strength has

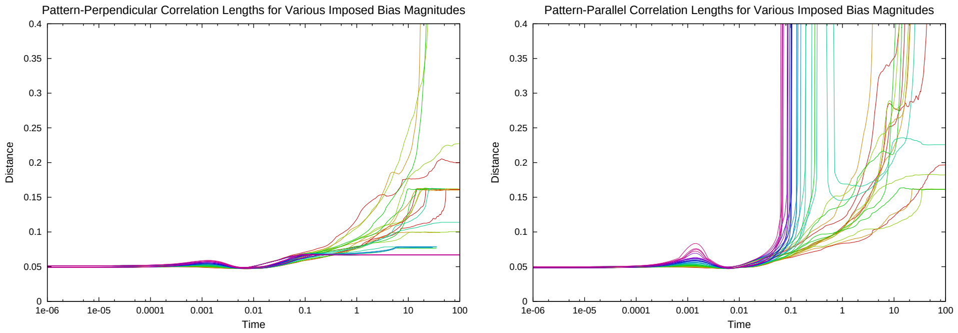

Figure 7.39: The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple).

a clear but temporary effect on correlation lengths in the early-stage solution development from $t = 0.0001$ to $t = 0.001$, where stronger pattern biases encourage growth of correlation lengths in all directions, albeit somewhat more strongly in the pattern-parallel direction. It may be that the spatially correlated, directed fluxes created by the patterning bias, combined with the diffusive interfacial terms in the system, temporarily promote development at a longer wavelength than the instabilities created by the anti-diffusive process of phase separation. In any case, these effects of phase separation start to grow exponentially as the solution perturbation away from an unstable homogeneous equibrium grows, and correlation lengths are again suppressed by the time $t = 0.01$, only to resume their growth as the separated phases reach their binodal concentration values and material region coarsening begins.

The effects of pattern bias amplitude on defect count are equally obvious, as can be seen in Figure 7.40. Stronger patterning bias leads on average to fewer defects in the end state, faster defect count reduction, and less stochastic

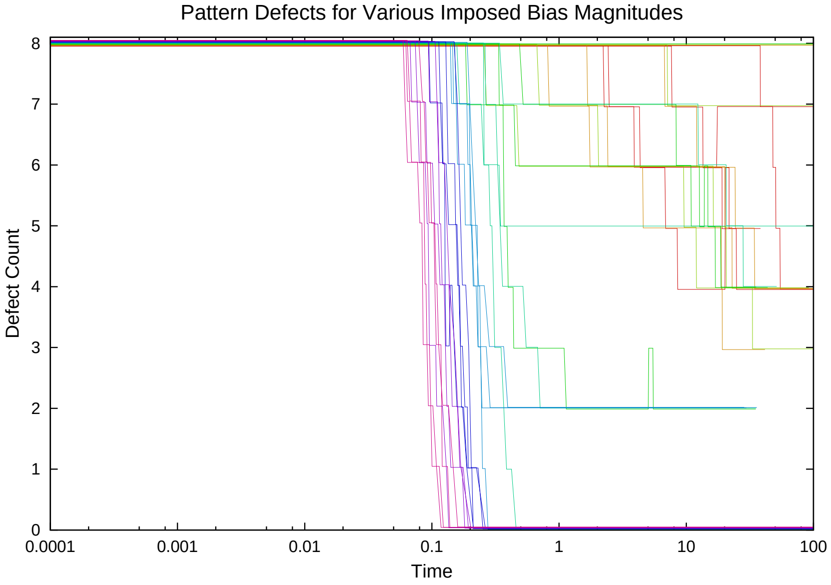

Figure 7.40: The defect count of the Cahn-Hilliard system plotted against time, for pattern bias amplitudes ranging from 0 (red) to 0.08 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph.

variation in the results. One interesting feature of this graph is that it can be seen that the defect count is not always monotonically decreasing, even in the late-time evolution of the problem. Weakly and moderately patterned simulations can be seen here to occasionally have new defects form, temporarily or even permanently.

Temperature The chemical free energy function for a Cahn-Hilliard decomposition problem is highly dependent on the temperature of the system. As shown in Figure 7.41, the total free energy available at spinodal points and the very double-well shape of the free energy diagram depend on temperature. At high temperatures, the free energy diagram becomes a single-well at which the most stable concentration is an even mixture. As the temperature is lowered, the thermodynamic impetus towards phase separation becomes greater and the final stable binodal concentration values become more distinct.

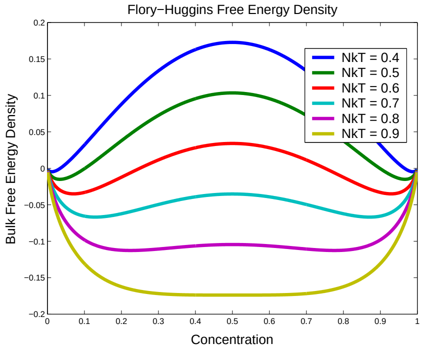

Figure 7.41: The chemical free energy density of the Cahn-Hilliard system plotted against local concentration, for various temperatures.

One effect of the different chemical free energy functions on the total free energy is clear: as seen in Figure 7.42, lower temperature systems initially

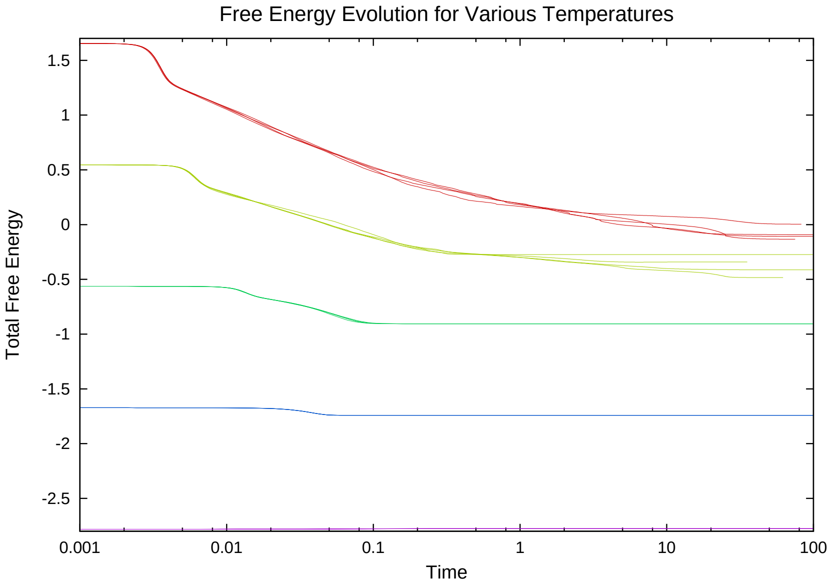

Figure 7.42: The free energy of the Cahn-Hilliard system plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple).

have much higher global free energy content, which provokes earlier phase separation and more dramatic free energy reduction. Lower temperature systems are also much more susceptible to stochastic effects, reaching significantly different end states from different initial perturbations.

The variation of system temperature has a dramatic effect on the rate of evolution of the system. In Figure 7.43, the low temperature experiments at $NkT = 0.5$ display the familiar phases of spinodal decomposition: the development of the initial perturbation into an unstable mode, the accelerating growth of that mode until binodal concentration values are reached, followed by a random but progressively slowing evolution of the system which is intermittently punctuated by higher speed topological change events. As the system temperature is increased, the observed evolution begins to differ. The

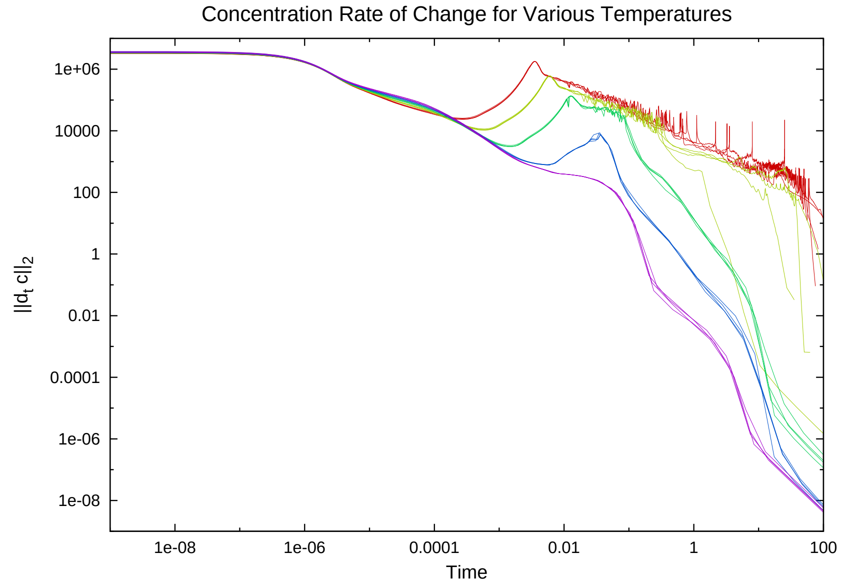

Figure 7.43: The rate of change of the concentration (in H^2 norm) plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple).

first differences are quantitative, at $NkT = 0.6$, where the phase decomposition is delayed, the random late-term evolution of the problem is shortened, and the systems reach a steady state earlier. Qualitative differences occur at $NkT = 0.7$, where the random "spikes" of sudden late-term topological change events are not seen at all, but where instead a short random development period is followed by a long, smooth decline in change rate. At $NkT = 0.8$, even that random development disappears, and the phase decomposition settles down to steady state entirely in a gradual manner. Finally, at $NkT = 0.9$, with our miscibility parameter $N\omega = 1.8$ the chemical free energy diagram is a single well. Here, no phase separation occurs, and the entire evolution of the system is a smooth readjustment of concentration to approach the spatially dependent equilibrium mixture values determined by the patterning bias.

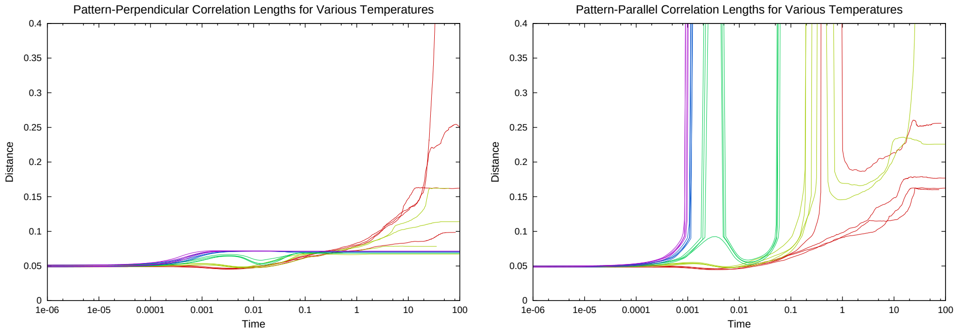

Figure 7.44: The horizontal and vertical correlation lengths of the Cahn-Hilliard system plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple).

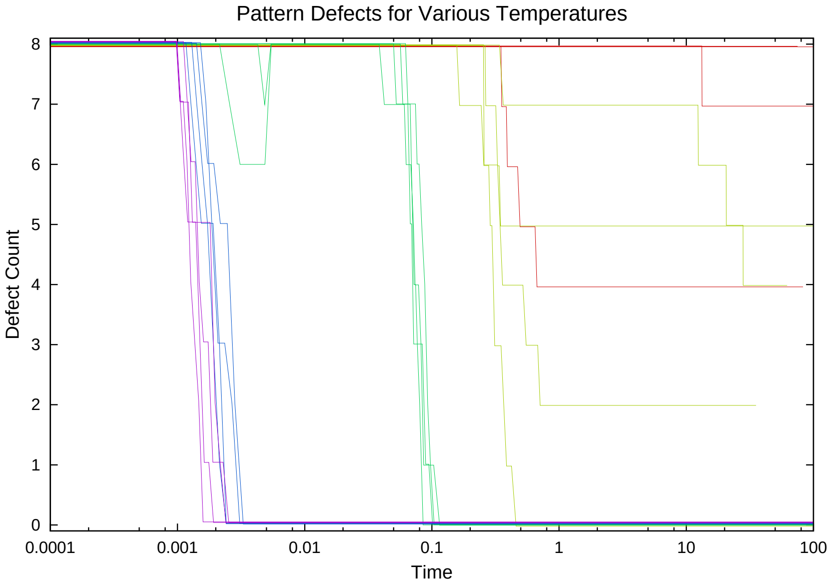

Figure 7.45: The defect count of the Cahn-Hilliard system plotted against time, for NkT ranging from 0.5 (red) to 0.9 (purple). Lines are slightly staggered away from integer values to avoid overlap on the graph.

The effect of these differences on the pattern is plotted in Figures 7.44 and 7.45. At lower temperatures, the more sudden phase decomposition causes defects to be "fixed" in place. Shorts and breaks in the pattern persist to the steady state, and the pathways left open between neighboring pattern lines allow relatively unrestrained growth of pattern-perpendicular correlation lengths. Higher temperatures give a more gradual decomposition, allowing concentration to migrate more reliably into the desired pattern; correlation lengths in the pattern-perpendicular direction are pinned by the pattern frequency, while correlation lengths following the grain of the pattern become infinite as the solution becomes constant in that direction. At the highest temperatures, however, the "desired pattern" is never fully reached; although the postprocessing of these solutions detects no shorts or breaks in the $NkT = 0.9$ pattern, there is also no phase decomposition in the $NkT = 0.9$ pattern; instead of using a weakly spatially varying imposed pattern to produce sharp lines of concentration, in these simulations a weakly varying imposed pattern produces a weakly spatially varying concentration.

One possibility that suggests itself from these results is to carry out phase decompositions at time-varying temperatures; instead of instantly quenching the mixed film to its final temperature, the temperature could be lowered gradually to the end state. This raises some interesting control possibilities. Through appropriate timing of the cooling process, the reliable patterning produced at high temperatures might be preserved while the pattern sharpening at lower temperatures is achieved. Even in the absence of deliberate control, the sensitivity of pattern formation to temperature suggests that the "instantaneous quench" assumption of spinodal decomposition should not be applied to the self-patterning problem, but that instead the continuous time history of quench temperature should be taken into consideration.

7.15 Simulation Performance

In many numerical spinodal decomposition studies in the literature, the simulation is "cut off" at some time, usually before the length scale of the coarsening material regions becomes large enough that the effect of the boundary conditions of the artificially truncated domain becomes a significant factor.

In patterned decomposition studies, however, the patterning acts to restrict the length scale of the evolving system, and if the domain size is a sufficient multiple of the pattern wavelength then it can make sense to consider the evolution of the physical system out to a steady state, as was done in the Section 7.14 simulations.

However, solving a Cahn-Hilliard system to steady state can be impractical with standard uniform time-stepping schemes. With the time-step sizes required to properly resolve the rapid solution evolution at the beginning of a spinodal decomposition, hundreds of thousands of steps might be needed to evolve even a patterned solution to steady state. The first steady state solutions for an unpatterned 3D Cahn-Hilliard problem are found in [68], where the gains afforded by adaptive time stepping make it feasible to track the slow late-stage development of the solution.

In these parametric studies, we found similar dramatic gains from the adaptive time stepping. Using a time integrator error tolerance to control time-step length as described in Section 3.5.1, we see in Figure 7.46 that, in a typical parametric study, the time-step lengths achieved can grow by several orders of magnitude over the course of the simulations. Each simulation occasionally needs to reduce time-step sizes to properly resolve sudden topological changes in the solution, such as merging material regions or evaporating droplets. These sudden changes are brief, with a general upward trend in time-step size until a steady state is reached. The specific results of a simulation depend on both the choice of physical parameters and on the random perturbation in the initial conditions. To give a better idea of the behavior for an entire parameteric study, Figure 7.47 plots the minimum, mean, and maximum time step sizes used by the set of simulations in the bias amplitude study from Section 7.14. The growth in the minimum time-step sizes used by these simulations is not as dramatic as the growth in the maximum step sizes attained. However, because the minimum step sizes are constrained by short intermittent reductions to track topological changes, the plot of average time-step growth resembles that for the maximums more than the minimums; an increase of six orders of magnitude in time-step size is observed over the course of the set of simulations.

The ability to take longer time steps via adaptive step size selection is not an unequivocal success. In Figure 7.48, the number of nonlinear solver iterations per time step is plotted. It can be seen that the developing simulation tends to require more nonlinear solver iterations at later time steps. This appears to be due in part to the increasing length of the time steps themselves, as the mass matrix component of the nonlinear algebraic system generated for the time step becomes less important. It can be seen that points in Figure 7.46

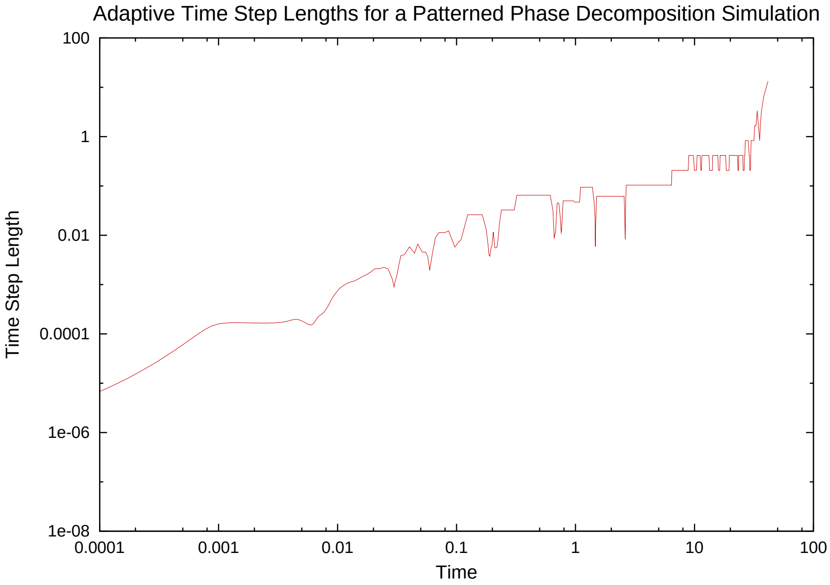

Figure 7.46: The time-step lengths at each point in simulation time for a single run of a patterned spinodal decomposition problem.

where the time-step length is reduced to meet an error tolerance correspond to local reductions in nonlinear solver iterations in Figure 7.48.

The overall increase in nonlinear solver steps in this simulation is not nearly as dramatic as the growth in time step size which it enables. In the worst case in late time steps, large time-step selection based solely on time integration error tolerances can lead to problems which are too nonlinear for our Newton-Krylov solver to quickly solve, and so the solver occasionally "wastes" its maximum number of iterations before deciding to reduce the time-step length and start over. Even with this cost, however, the number of nonlinear steps required does not seem to grow without bound. Figure 7.49 confirms this bounded behavior, in plots of the minimum, average, and maximum number of nonlinear solver steps per timestep for the entire set of simulations used in

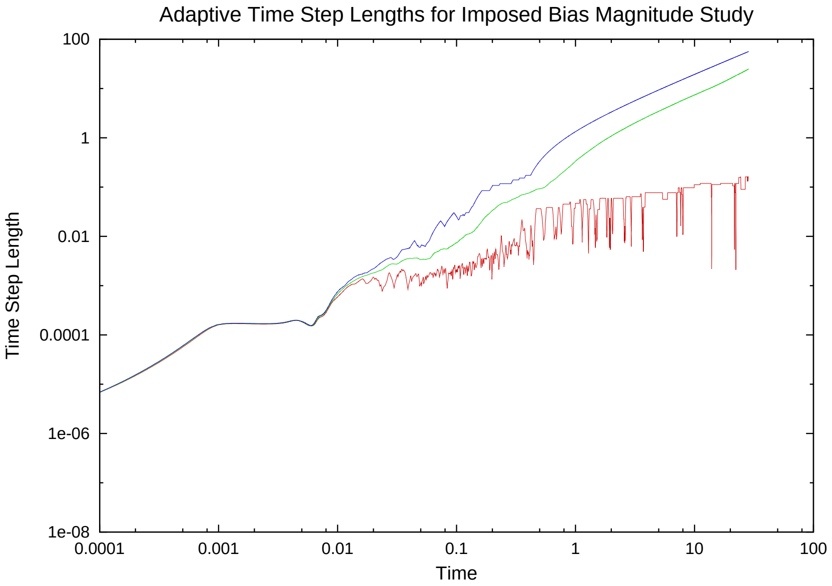

Figure 7.47: The minimum, mean, and maximum time-step length at each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study.

the bias amplitude study. There is a wide variation between minimum and maximum, but on average the solver requirements are well-behaved.

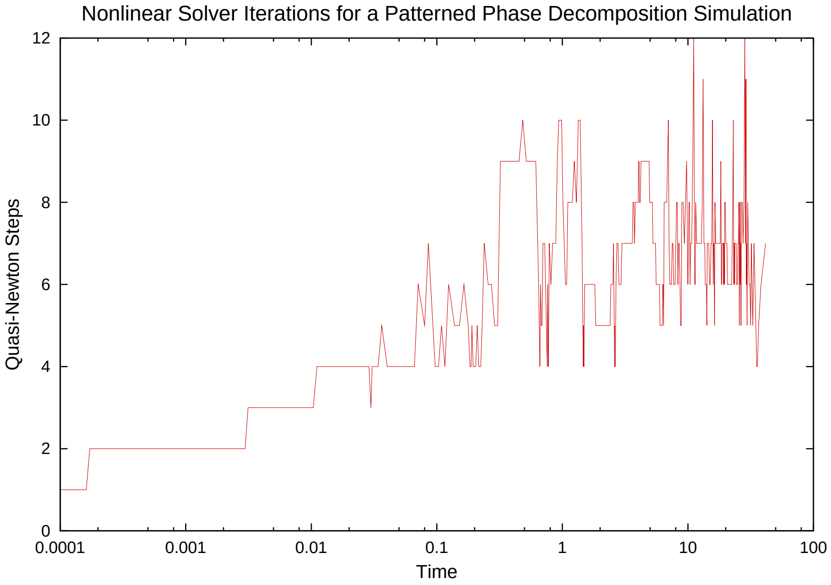

Figure 7.48: The number of inexact Newton steps taken at each point in simulation time for a single run of a patterned spinodal decomposition problem.

The increasing linear solver cost at later timesteps is not as well-behaved, for two reasons. First, in the early simulation times, there are sharp jumps in the linear iteration counts, corresponding to points where the nonlinear solver needs to add an additional inexact Newton step and all the additional Krylov steps that entails. Second, each nonlinear solver step takes as many linear solver iterations as is necessary to meet the specified residual reduction tolerance for that step. This is another source of increased computational cost. In the late stages of the simulations, the number of Krylov steps required per time step trends generally upward, far more than the growth in the number of inexact Newton steps. The linear solver performance will obviously be sen-

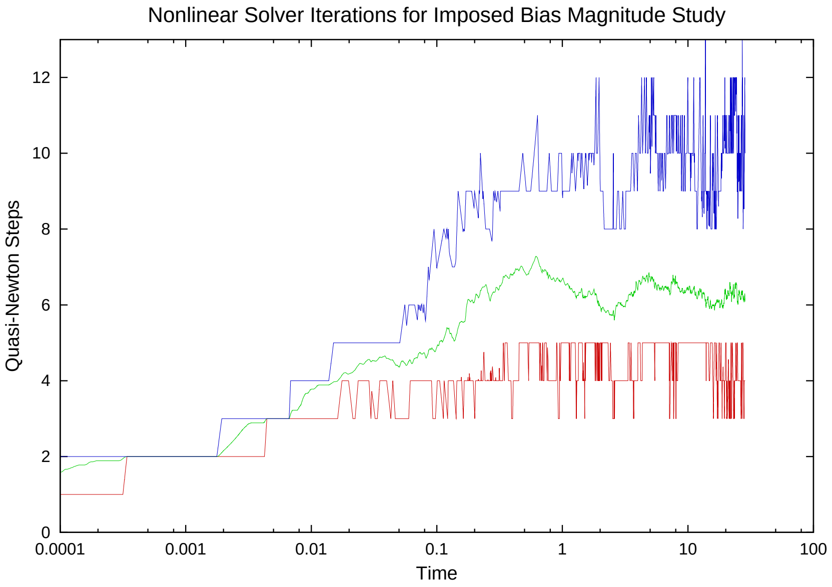

Figure 7.49: The minimum, mean, and maximum number of inexact Newton solver steps taken at time steps covering each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study.

sitive to particular choices of Krylov method and preconditioner type, but in general this upward trend is expected behavior, as the system Jacobians which resembled a mass matrix in the earliest time steps evolve to include more of the ill-conditioned nonlinear and fourth-order terms in late time steps.

Figure 7.51, another plot of minimum, average, and maximum values over an entire parametric study sample set, shows that this effect is typical. We should be concerned that increasingly expensive solves at each time step will act to offset the computational efficiency achieved with longer time steps.

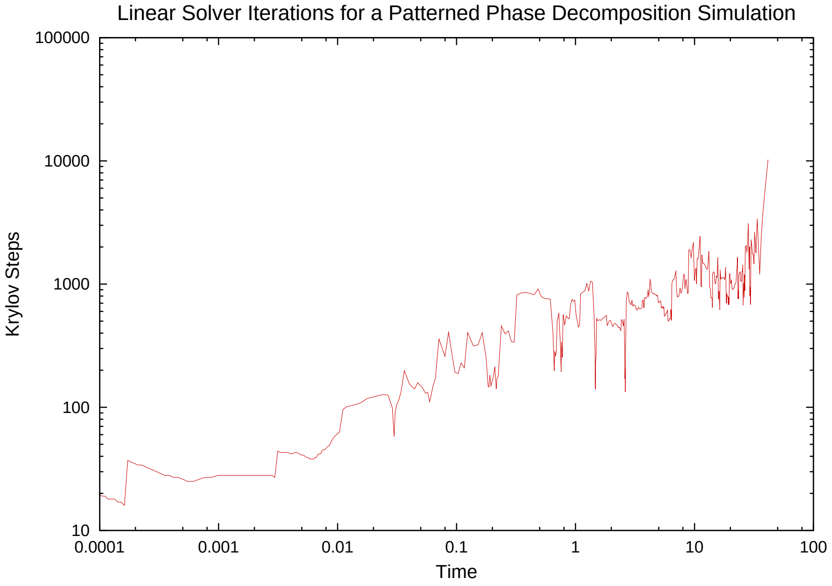

Figure 7.50: The number of Krylov steps taken at each point in simulation time for a single run of a patterned spinodal decomposition problem.

To discern the combined effect of these factors on our simulations, we define the “solve rate” to be the number of units of simulation time being calculated per second of wall clock time. If, during timestep n, the simulation is advanced from time t_n to $t_n + \delta t_n$ by a computer which takes s seconds

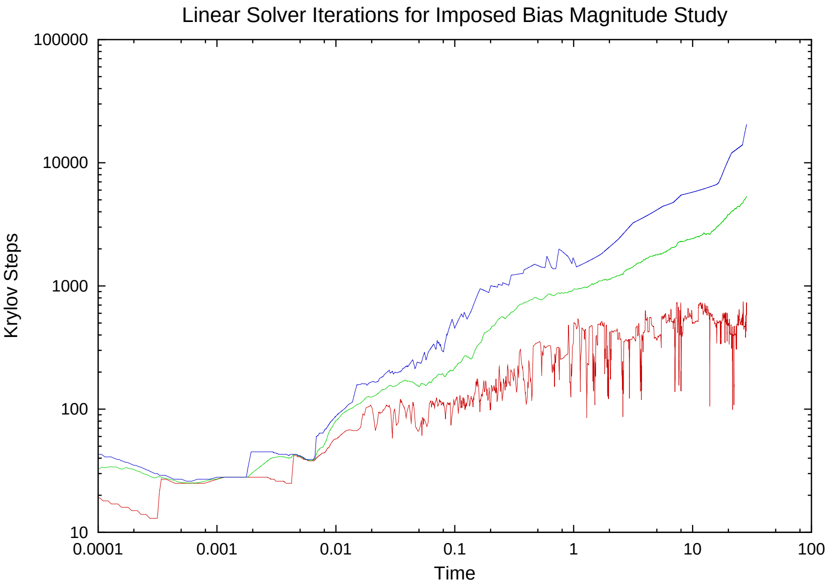

Figure 7.51: The minimum, mean, and maximum number of total Krylov solver steps taken during time steps covering each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study.

to perform all the necessary calculations, then the solve rate is $\frac{\delta t_n}{s}$ for that time step. Figure 7.52 plots the solve rate for the same single simulation as seen in our earlier example plots. Despite the countervailing increases in solver cost per time step, the predominant effect that can be seen is still a dramatic improvement in solver efficiency as time-step lengths increase. This improvement can be seen for an entire parametric study in the minimum, average, and maximum solve rate plots in Figure 7.53. The minimum solve rate is always constrained by the potential need to track a late topological change in a particular solution, but the average solve rate for the set of study samples still grows dramatically, with an improvement of more than five orders of magnitude over the course of the simulations.

This parameter study was executed on the Texas Advanced Computing Center's supercomputer, Lonestar, using four quad-processor nodes for each Monte Carlo sample, and running up to 40 samples at a time in parallel. Although the specific solve rates achieved depend on solver parameters and hardware choices, the general principle demonstrated is clear: with sufficiently robust solvers, adaptive time step selection allows phase decomposition simulations to be carried out to very late simulation times without correspondingly prohibitive cost.

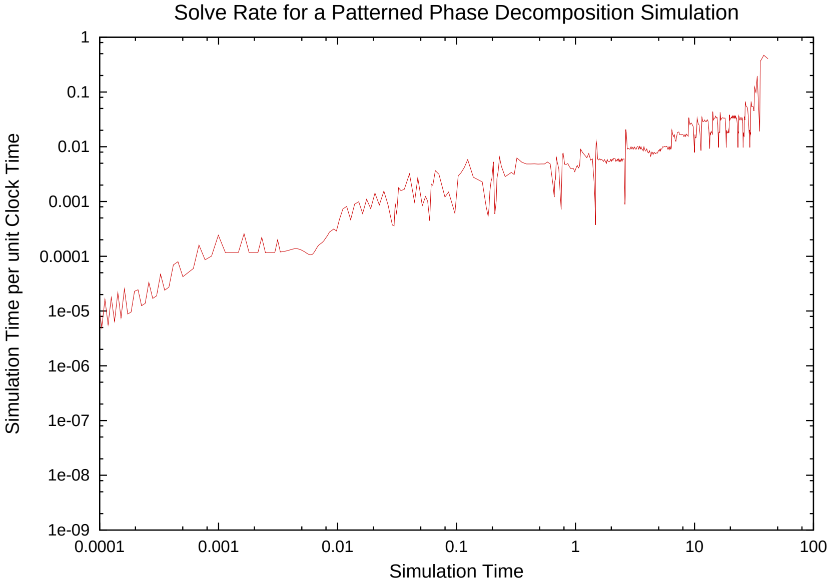

Figure 7.52: The solver efficiency (expressed as a ratio of simulation time elapsing to clock time elapsing) at each point in simulation time for a single run of a patterned spinodal decomposition problem.

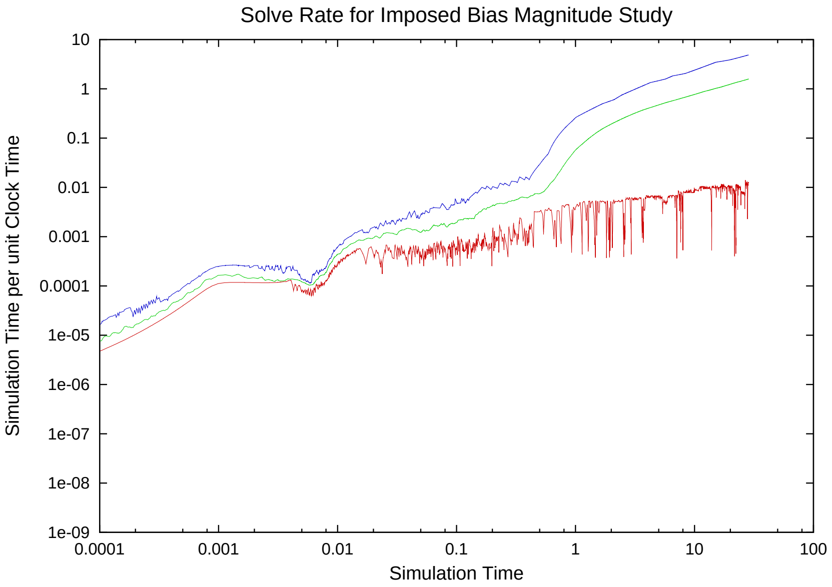

Figure 7.53: The minimum, mean, and maximum solver efficiency (expressed as a ratio of simulation time elapsing to clock time elapsing) during time steps covering each point in simulation time, for many Monte Carlo samples and parameter values in a pattern bias amplitude study.

Chapter 8

Concluding Remarks

The infrastructure developments which were a part of this work have been added to the public open source `libMesh` project, made available to the greater scientific and engineering community via the SourceForge software repository, where the `libMesh` library sees hundreds of downloads and thousands of web visits each month. The most generally applicable components, such as the new object-oriented boundary-value problem framework and its supporting classes, have already been used in recent research for other applications [102]. There is also a foundation here to build on for future work with fourth-order problems.

Tetrahedral Macroelements In two dimensions, the generation of a quality triangular mesh on an arbitrary domain is essentially a solved problem, and the triangular macroelements studied here are an excellent way to generate adaptable C^1 conforming bases on such a mesh. Quad meshes and hybrid quad/triangle meshes are also popular, and it would be straightforward to use a compatible quadrilateral macroelement such as the Lai quad [90] alongside the Clough-Tocher triangles implemented for this work.

The Hermite tensor product spaces used for our 3D results are less ideal; they provide efficient C^1 splines on hierarchically adapted Cartesian grids, and can be extended to meshes which are C^1 mappings of such grids, but not every domain can be discretized in this way. Even without such topology restrictions, quality hexahedral mesh generation is still a difficult problem, with promising techniques for restricted classes of domains, but open questions in general [17, 94, 120, 139].

On the other hand, robust algorithms and software are now available for automatic pure-tetrahedral mesh generation [20, 50, 67, 114], with element quality dependent only on the limitations of the domain to be meshed and on the steady improvements in the meshing literature. For a pure tetrahedral mesh, C^1 macroelement solutions exist which are compatible with the non-conforming hanging node continuity constraint techniques described in Section 3.1. The implementation and testing of these elements in a parallel adaptive code would require much work and debugging, but would provide a useful capability, with no real obstacles to reapplying the approaches from this work.

C^1 p Refinement Although macroelement construction is motivated by the challenge of creating a polynomial spline space with high levels of continuity but low polynomial degree p, for sufficiently smooth problems the improved convergence rates of high-p elements can be attractive, particularly when combined with mesh refinement in an hp method.

Although the constraint methods of Section 3.1 can be applied to adaptive p refinement problems, the constraint process can be made more elegant (or in purely p refined conforming meshes, avoided altogether) for hierarchic bases. A simple change of basis (and a corresponding change in degree of freedom functionals) would, for instance, turn the Clough-Tocher-Percell [100] quartic macrotriangle basis into a strict superset of the Clough-Tocher cubic basis. Extending to $p = 5$ and up is relatively straightforward.

Initial support for both adaptive p and hp refinement with p-hierarchic bases, as well as high p extensions of the Hermite elements, have already been added to `libMesh` for experimental purposes alongside the present research activity.

Coupled Flow/Decomposition For the polymer phase decomposition studies in this work, the bulk velocity of the material is assumed to be zero, because in typical experiments the decomposition rate is sufficiently low that coupling to hydrodynamic effects can be neglected. In more rapid physical processes, however, the Cahn-Hilliard equation may be coupled to the Navier-Stokes equations to examine flow in a fluid mixture. This case is found in the literature using finite rifference [75, 76] and lattice Boltzmann methods [23, 82]; it would be interesting to compare such work with the results of a conforming Finite Element formulation.

Self-Patterning Control Problems The results in Section 7.14 suggest a variety of ways in which the reliability of self-patterning phase decomposition processes can be influenced by the choice of the physical parameters governing the system. Some of these parameters, such as film thickness and material properties, cannot be easily manipulated during the course of an experiment, but others, such as quenching temperature, can be made subject to fine-tuned control by the experimenter. There may be tradeoffs between the more rapid phase decomposition which occurs at relatively low quenching temperatures versus the more reliable pattern self-assembly which we found to occur at relatively high temperatures. Transient simulations of gradually cooling mixtures could be instructive in determining the effectiveness of utilizing temperature control to stabilize the decomposition process.

Bibliography

[1] S. Adjerid. An a posteriori error estimate for fourth-order boundary value problems. *CMAME*, 191:2539–2559, 2002.

[2] M. Ainsworth and J. T. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. Wiley-Interscience, New York, 2000.

[3] P. Alfeld. A trivariate clough-tocher scheme for tetrahedral data. *Computer Aided Geometric Design*, 1:169–181, 1984.

[4] C. Amrouche, C. Bernardi, M. Dauge, and V. Girault. Vector potentials in three-dimensional non-smooth domains. *Math. Meth. Appl. Sci.*, 21:823–864, 1998.

[5] D. M. Anderson, G. B. McFadden, and A. A. Wheeler. Diffuse-interface methods in fluid mechanics. *Annual Rev. Fluid Mech.*, 30:139–165, 1998.

[6] G. Awanou and M. Lai. Quintic spline interpolation over tetrahedral partitions, 2002.

[7] I. Babuska, O. C. Zienkiewicz, J. Gago, and E. R. A. Oliv iera. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations.* Wiley, 1986.

[8] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc Web page. http://www.mcs.anl.gov/petsc, 2007.

[9] W. Bangerth and R. Rannacher. *Adaptive Finite Element Methods for Differential Equations.* Birkhäuser Verlag, 2003.

[10] J. W. Barrett, H. Garcke, and R. Nurnberg. Finite element approximation of surfactant spreading on a thin film. *SIAM J. Numer. Anal*, 41(4):1427–1464, 2003.

[11] J. W. Barrett, H. Garcke, and R. Nurnberg. Finite element approximation of a phase field model for surface diffusion of voids in a stressed solid. *Math. Comp.*, 75:7–41, 2006.

[12] W. Barth. *Simulation of Non-Newtonian Fluids on Workstation Clusters.* PhD dissertation, The University of Texas at Austin, 2004.

[13] W. Barth, G. Carey, S. Chow, and B. Kirk. Finite Element Modeling of Generalised Newtonian Flows. In *Proceedings of the 14th Australasian Fluids Conference, Adelaide*, December 2001.

[14] W. L. Barth and G. F. Carey. Extension of the cht-01 natural convection benchmark problem to non-newtonian fluids. In *Proceedings of CHT-04: ICHMT International Symposium on Advances in Computational Heat Transfer*, Norway, April 2004. ICHMT.

[15] B. Bejanov, J.-L. Guermond, and P. D. Minev. A locally DIV-free projection scheme for incompressible flows based on non-conforming finite

elements. *Int. J. Numer. Meth. Fluids*, 49:549–568, 2005.

[16] O. Bíró, K. Preis, and K. R. Richter. On the use of the magnetic vector potential in the nodal and edge finite element analysis of 3d magnetostatic problems. *IEEE Transactions on Magnetics*, 32(3):651–654, May 1996.

[17] T. D. Blacker and R. J. Meyers. Seams and wedges in plastering: A 3d hexahedral mesh generation algorithm. *Engineering With Computers*, 2(9):83–93, 1993.

[18] W. Boettinger, J. A. Warren, C. Beckermann, and A. Karma. Phase-field simulation of solidifaction. *Annual Review of Materials Research*, 32:163–194, 2002.

[19] M. Böltau, S. Walheim, J. Mlynek, G. Krausch, and U. Steiner. Surface-induced structure formation of polymer blends on patterned substrates. *Nature*, 391:877–879, February 1998.

[20] H. Borouchaki, F. Hecht, E. Saltel, and P. L. George. Reasonably efficient delaunay based mesh generator in 3 dimensions. In *Proceedings of the 4th International Meshing Roundtable*, pages 3–14, October 1995.

[21] R. J. Braun and B. T. Murray. Adaptive phase-field computations of dendritic crystal growth. *Journal of Crystal Growth*, 174(1):41–53, 1997.

[22] Richard P. Brent. *Algorithms for Minimization without Derivatives.* Prentice-Hall, Englewood Cliffs, N.J., 1973.

[23] A. J. Briant and J. M. Yeomans. Lattice Boltzmann simulations of contact line motion. II. Binary fluids. *Physical Review E*, 69:031603, 2004.

[24] P. N. Brown and Y. Saad. Hybrid Krylov Methods for Nonlinear Systems of Equations. *SIAM Journal on Scientific and Statistical Computing*, 11(3):450–481, 1990.

[25] J. W. Cahn. On spinodal decomposition. *Acta Materialia*, 9:795–801, 1961.

[26] J. W. Cahn and J. E. Hilliard. Free energy of a nonuniform system. i. interfacial free energy. *J. of Chem. Phys.*, 28(2):258–267, February 1958.

[27] G. F. Carey. A mesh refinement scheme for finite element computations. *Computer Methods in Applied Mechanics and Engineering*, 7(1):73–105, 1976.

[28] G. F. Carey. *Computational Grids.* Taylor& Francis, New York, 1997.

[29] G. F. Carey, M. Anderson, B. Carnes, and B. Kirk. Some aspects of adaptive grid technology related to boundary an d interior layers. *J. Comp. Appl. Math.*, 166(1):55–86, 2004.

[30] G. F. Carey and S. S. Chow. Numerical approximation of generalized Newtonian fluids using Powell-Sabin-Heindl Elements: I. Theoretical estimates. *Int. J. Numer. Meth. Fluids*, 41:1085–1118, 2003.

[31] G. F. Carey and J. T. Oden. *Finite Elements–An Introduction, Volume I.* Prentice Hall, Englewood Cliffs, NJ, 1981.

[32] G. F. Carey and J. T. Oden. *Finite Elements–Mathematical Aspects, Volume IV.* Prentice Hall, Englewood Cliffs, NJ, 1983.

[33] G. F. Carey and J. T. Oden. *Finite Elements: Fluid Mechanics, Volume VI.* Prentice Hall, Englewood Cliffs, NJ, 1983.

[34] G. F. Carey and M. Utku. Boundary penalty techniques. *J. Comp. Meth. Appl. Mech. Eng.*, 30:103–118, 1982.

[35] G. F. Carey and M. Utku. Penalty resolution of the Babuska-circle paradox. *J. Comp. Meth. Appl. Mech. Eng.*, 41:11–28, 1983.

[36] V. Carey. *A Posteriori Error Estimation for the Finite Element Method via Local Averaging.* PhD dissertation, Cornell University, 2005.

[37] C. Castellano and S. C. Glotzer. On the mechanism of pinning in phase-separating polymer blends. *J. Chem. Phys.*, 103(21):9363–9369, December 1995.

[38] H. D. Ceniceros and A. M. Roma. A nonstiff, adaptive, mesh refinement-based method for the Cahn-Hilliard equation. *J. Comput. Phys*, 225(2):1849–1862, 2007.

[39] A. Charbonneau, K. Dossou, and R. Pierre. A residual-based a posteriori error estimator for the Ciarlet-Raviart formulation of the first biharmonic problem. *Num. Meth. PDE*, 13:93–111, 1997.

[40] L.-Q. Chen. Phase-field models for microstructure evolution. *Annual Review of Materials Research*, 32:113–140, 2002.

[41] S. M. Choo and S. K. Chung. Conservative nonlinear difference scheme for the Cahn-Hilliard equation. *Computers Math. Applic.*, 36(7):31–39, 1998.

[42] P. J. Ciarlet. *The Finite Element Method for Elliptic Problems.* North-Holland, Amsterdam, 1978.

[43] R. Clough and J. Tocher. Finite element stiffness matrices for analysis of plates in blending. In *Proceedings of Conference on Matrix Methods in Structural Analysis*, 1965.

[44] H. E. Cook. Brownian motion in spinodal decomposition. *Acta Metallurgica*, 18(3):297–306, March 1970.

[45] H. L. De Cougny and M. S. Shephard. Parallel refinement and coarsening of tetrahedral meshes. *Int. J. Num. Meth. Eng.*, 46(7):1101–1125, 1999.

[46] S. D. Cramer and J. M. Marchello. Numerical evaluation of models describing non-Newtonian behavior. *AIChE Journal*, 14:980–983, November 1968.

[47] M. J. Crochet, A. R. Davies, and K. Walters. *Numerical Simulation of Non-Newtonian Fluid Flow*, volume 1. Elsevier, 1984.

[48] S. H. Davis. Rupture of thin liquid films. In R. E. Meyer, editor, *Waves on Fluid Interfaces*, pages 291–302. 1983.

[49] R. S. Dembo, S. C. Eisenstat, and T. Steihaug. Inexact Newton Methods. *SIAM Journal on Numerical Analysis*, 19(2):400–408, 1982.

[50] H. Edelsbrunner and N. R. Shah. Incremental topological flipping works for regular triangulations. *Algorithmica*, 15:223–241, 1996.

[51] S. C. Eisenstat and H. F. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing*, 17(1):16–32, 1996.

[52] C. M. Elliot and D. A. French. Numerical studies of the Cahn-Hilliard equation for phase separation. *IMA J. Appl. Math.*, 38:97–128, 1987.

[53] C. M. Elliott and D. A. French. A nonconforming finite-element method for the two-dimensional Cahn-Hilliard equation. *SIAM J. Numer. Anal.*, 26(4):884–903, August 1989.

[54] C. M. Elliott and H. Garcke. On the Cahn-Hilliard equation with degenerate mobility. *SIAM J. Math. Anal.*, 27(2):404–423, March 1996.

[55] C. M. Elliott and S. Larsson. Error estimates with smooth and nonsmooth data for a finite element method for the Cahn-Hilliard equation. *Mathematics of Computation*, 58(198):603–630, April 1992.

[56] C. M. Elliott and S. Zheng. On the Cahn-Hilliard equation. *Arch. Rational Mech. Anal.*, 96:339–357, 1986.

[57] H. Emmerich. *The Diffuse Interface Approach in Materials Science*. Springer, New York, 2003.

[58] G. Engel, K. Garikipati, T. J. R. Hughes, M. G. Larson, L. Mazzei, and R. L. Taylor. Continuous/discontinuous finite element approximations of fourth-order elliptic problems in structural and continuum mechanics with applications to thin beams and plates, and strain gradient elasticity. *Comp. Meth. App. Mech. Eng.*, 191:3669–3750, 2002.

[59] D. J. Estep, M. G. Larson, and R. D. Williams. *Estimating the Error of Numerical Solutions of Systems of Reaction-Diffusion Equations*, volume 146. 2000.

[60] D. J. Eyre. Systems of Cahn-Hilliard equations. *SIAM J. Appl. Math.*, 53(6):1686–1712, December 1993.

[61] F. Fairag. Numerical computations of viscous, incompressible flow problems using a two-level finite element method. *SIAM J. Sci. Comput.*, 24(6):1919–1929.

[62] J. E. Flaherty, P. J. Paslow, M. S. Shephard, and J. D. Vasilakis, editors. *Adaptive Methods for Partial Differential Equations*. SIAM, 1989.

[63] P. J. Flory. Thermodynamics of high polymer solutions. *J. of Chem. Phys.*, 9(8):660, August 1941.

[64] D. Furihata. Finite difference schemes for $\frac{\partial u}{\partial t} = \left(\frac{\partial}{\partial x}\right)^\alpha \frac{\delta g}{\delta u}$ that inherit energy conservation or dissipation property. *J. Comp. Phys.*, 156:181–205, 1999.

[65] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-

Wesley, 1995.

[66] H. Garcke, B. Niethammer, M. Rumpf, and U. Weikard. Transient coarsening behaviour in the Cahn-Hilliard model. *Acta Materialia*, 51:2823–2830, 2003.

[67] N.A. Golias and T.D. Tsiboukis. An approach to refining three-dimensional tetrahedral meshes based on Delaunay transformations. *Intl. J. Numer. Meth. Eng.*, 37:793–812, 1994.

[68] H. Gomez, V. M. Calo, Y. Bazilevs, and T. J. R. Hughes. Isogeometric analysis of the Cahn-Hilliard phase-field model. *Computer Methods in Applied Mechanics and Engineering*, submitted, 2007.

[69] T. Grätsch and K.-J. Bathe. A posteriori error estimation techniques in practical finite element analysis. *Computers and Structures*, 83:235–265, January 2005.

[70] P. M. Gresho and R. L. Sani. *Incompressible Flow and the Finite Element Method*. John Wiley and Sons, New York, 1998.

[71] M. L. Huggins. Solutions of long chain compounds. *J. of Chem. Phys.*, 9(5):440, May 1941.

[72] T. J. R. Hughes. *The Finite Element Method–Linear Static and Dynamic Finite Element Analysis*. Dover, Mineola, NY, 1987.

[73] Michael Imamura. Using Doxygen: A quick guide to getting started and using the Doxygen inline documentation system for documenting source code. Technical report, Linux Users Group at Georgia Tech, May 2002.

[74] S. Iqbal and G. F. Carey. Performance analysis of dynamic load balancing algorithms with variable number of processors. *J. Parallel and Distributed Computing*, 65(8):934–948, 2005.

[75] D. Jacqmin. Calculation of two-phase Navier-Stokes flows using phase-field modeling. *J. Comp. Phys.*, 155:96–127, 1999.

[76] D. Jacqmin. Contact-line dynamics of a diffuse fluid interface. *J. Fluid Mech.*, 402:57–88, 2000.

[77] R.-Q. Jia. Approximation order from certain spaces of smooth bivariate splines on a three-direction mesh. *Transactions of the American Mathematical Society*, 295(1):199–212, May 1986.

[78] R. A. L. Jones, L. J. Norton, E. J. Kramer, F. S. Bates, and P. Wiltzius. Surface-directed spinodal decomposition. *Phys. Review Letters*, 66(10):1326–1329, March 1991.

[79] A. Karim, J. F. Douglas, B. P. Lee, S. C. Glotzer, J. A. Rogers, R. J. Jackman, E. J. Amis, and G. M. Whitesides. Phase separation of ultrathin polymer-blend films on patterned substrates. *Physical Review E*, 57(6):R6273–R6276, June 1998.

[80] D. Kay and R. Welford. A multigrid finite element solver for the Cahn-Hilliard equation. *J. Comp. Phys.*, 212:288–304, 2006.

[81] D. W. Kelly, J. P. Gago, O. C. Zienkiewicz, and I. Babusk a. A posteriori error analysis and adaptive processes in the finite element method: Part I error analysis. *Int. J. Num. Meth. Engng.*, 19:1593–1619, 1983.

[82] V. M. Kendon, M. E. Cates, I. Pagonabarraga, J.-C. Desplat, and P. Bladon. Inertial effects in three-dimensional spinodal decomposition of a symmetric binary fluid mixture: a lattice Boltzmann study. *J. Fluid Mech.*, 440:147–203, 2001.

[83] D. Kessler, J.-F. Scheid, G. Schimperna, and U. Stefanelli. Study of a system for the isothermal separation of components in a binary alloy with change of phase. *IMA J. of Appl. Math.*, 69:233–257, 2004.

[84] E. Khain and L. M. Sander. Generalized Cahn-Hilliard equation for biological applications. *Physical Review E*, 77:051129, 2008.

[85] R. C. Kirby. Fiat: A new paradigm for computing finite element basis functions. *ACM Trans. Math. Software*, 30(4):502–516, December 2004.

[86] B. S. Kirk, J. W. Peterson, R. H. Stogner, and G. F. Carey. `libMesh`: A C++ Library for Parallel Adaptive Mesh Refinement/Coarsening Simulations. *Engineering With Computers*, 22(3):237–254, December 2006.

[87] G. Krausch, E. J. Kramer, M. H. Rafailovich, and J. Sokolov. Self-assembly of a homopolymer mixture via phase separation. *App. Phys. Letters*, 64(20):2655–2657, May 1994.

[88] O. A. Ladyzhenskaya. Sixth problem of the millenium: Navier-Stokes quations, existence and smoothness. *Russian Math. Surveys*, 58(2):251–286, 2003.

[89] M. Lai and L. Schumaker. Macro-elements and stable local bases for splines on Clough-Tocher triangulations. *Numer. Math*, 88:105–119, 2001.

[90] M.-J. Lai. Scattered data interpolation and approximation using bivariate c1 piecewise cubic polynomials. *Computer Aided Geometric Design*, 13:81–88, 1996.

[91] J. Lowengrub and L. Truskinovsky. Quasi-incompressible Cahn-Hilliard fluids and topological transitions. *Proc. R. Soc. Lond. A*, 454:2617–2654, 1998.

[92] Message Passing Interface Forum. MPI: A Message Passing Interface. In *Proceedings of Supercomputing '93*, pages 878–883. IEEE Computer Society Press, 1993.

[93] S. Meyers. *Effective C++*. Addison-Wesley, Reading, Mass., 1992.

[94] L. Mingwu, S. E. Benzley, G. D. Sjaardema, and T. Tautges. A multiple source and target sweeping method for generating all-hexahedral finite element meshes. In *5th International Meshing Roundtable*, pages 217–228, October 1996.

[95] P. Monk. A mixed finite element method for the biharmonic equation. *SIAM J. Numer. Anal.*, 24(4):737–749, August 1987.

[96] T. L. Morkved, M. Lu, A. M. Urbas, E. E. Ehrichs, H. M. Jaeger, P. Mansky, and T. P. Russell. Local control of micdomain orientation in diblock

copolymer thin films with electric fields. *Science*, 273:931–933, August 1996.

[97] B. T. Murray, A. A. Wheeler, and M. E. Glicksman. Simulations of experimentally observed dendritic growth behavior using a phase-field model. *Journal of Crystal Growth*, 154(3-4):386–400, 1995.

[98] V. X. Nguyen and K. J. Stebe. Patterning of small particles by a surfactant-enhanced Marangoni-Bénard instability. *Physical Review Letters*, 88(16):164501, 2002.

[99] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, New York, 1970.

[100] P. Percell. On cubic and quartic Clough-Tocher finite elements. *SIAM J. Numer. Anal.*, 13:100–103, March 1976.

[101] J. Petera and J. F. T. Pittman. Isoparametric Hermite elements. *Int. J. Numer. Meth. Eng.*, 37(20):3489–3519, October 1994.

[102] J. Peterson. *Parallel Adaptive Finite Element Methods for Problems in Natural Convection.* PhD dissertation, The University of Texas at Austin, 2008.

[103] M. J. D. Powell and M. A. Sabin. Piecewise quadratic approximations on triangles. *ACM Trans. Math. Software*, 3:316–325, 1977.

[104] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flann ery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, New York, NY, 1988.

[105] S. Puri and H. L. Frisch. Surface-directed spinodal decomposition: modelling and numerical simulations. *J. Phys.: Condens. Matter*, 9:2109–2133, 1997.

[106] A. Ralston. *A First Course in Numerical Analysis*. McGraw-Hill, New York, 1965.

[107] J. N. Reddy. *An Introduction to the Finite Element Method*. McGraw-Hill, New York, 1984.

[108] J. N. Reddy and D. K. Gartling. *The Finite Element Method in Heat Transfer and Fluid Dynamics*. CRC Press, London, 1994, 2001.

[109] J. Reinders. *Intel Threading Building Blocks: Outfitting C++ for Multi-core Processor Parallelism*. O'Reilly Media, 2007.

[110] P. J. Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa Press, 1998.

[111] M. F. Schatz, S. J. VanHook, W. D. McCormick, J. B. Swift, and H. L. Swinney. Onset of surface-tension-driven Bénard convection. *Physical Review Letters*, 75(10):1938–1941, September 1995.

[112] R. Scheichl. Decoupling three-dimensional mixed problems using divergence-free finite elements. *SIAM J. Sci. Comput.*, 23(5):1752–1776.

[113] R. F. Service. Patterning electronics on the cheap. *Science*, 278(5337):383–384, October 1997.

[114] H. Si and K. Gaertner. Meshing piecewise linear complexes by constrained delaunay tetrahedralizations. In *Proceedings of the 14th International Meshing Roundtable*, pages 147–163, September 2005.

[115] R. H. Stogner and G. F. Carey. Parametric numerical investigations of directed pattern self-assembly in phase decomposition of thin films. *in preparation.*

[116] R. H. Stogner and G. F. Carey. C^1 macroelements in adaptive finite element methods. *Int. J. Numer. Meth. Eng.*, 70(9):1076–1095, May 2007.

[117] R. H. Stogner, B. S. Kirk, and J. W. Peterson. Data structures and algorithms for distributed-memory parallel adaptive mesh refinement in the `libMesh` library. *in preparation.*

[118] R. H. Stogner, B. T. Murray, and G. F. Carey. Approximation of Cahn-Hilliard diffuse interface models using parallel adaptive mesh refinement and coarsening with C^1 elements. *Int. J. Numer. Meth. Eng., in press*, 2008.

[119] Z. Suo and W. Hong. Programmable motion and patterning of molecules on solid surfaces. *Proc. Nat. Academy of Sciences USA*, 101(21):7874–7879, May 2004.

[120] T. J. Tautges, T. Blacker, and S. A. Mitchell. The whisker weaving algorithm: A connectivity-based method for constructing all-hexahedral finite element meshes. *Int. J. Numer. Meth. Eng.*, 39:3327–3349, 1996.

[121] K. Thorton, J. Agren, and P. W. Voorhees. Modelling the evolution of phase boundaries in solids at the meso- and nano-scales. *Acta Materialia*, 51:5675–5710, 2003.

[122] R. Toral, A. Chakrabarti, and J. D. Gunton. Numerical study of the cahn-hilliard equation in three dimensions. *Physical Review Letters*, 60(22):2311–2314, May 1988.

[123] R. Toral, A. Chakrabarti, and J. D. Gunton. Effect of the morphology of patterns on the scaling functions: off-critical quenches. *Physical Review B*, 39(1):901–904, January 1989.

[124] T. Tu, D. R. O'Hallaron, and O. Ghattas. Scalable parallel octree meshing for terascale applications. In *Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, 2005.

[125] R. S. Tuminaro, H. F. Walker, and J. N. Shadid. On Backtracking Failure in Newton-GMRES Methods with a Demonstration for the Navier-Stokes Equations. *J. Computational Physics*, 180:549–558, 2002.

[126] R. L. J. M. Ubachs, P. J. G. Schreurs, and M. G. D. Geers. A nonlocal diffuse interface model for microstructure evolution of tin-lead solder. *J. Mech. Phys. Solids*, 52:1763–1792, 2004.

[127] A. M. P. Valli, G. F. Carey, and A. L. G. A. Coutinho. Control strategies for timestep selection in simulation of coupled viscous flow and heat transfer. *Comm. Num. Meth. in Eng.*, 18(2):131–139, 2002.

[128] S. J. VanHook, M. F. Schatz, W. D. McCormick, J. B. Swift, and H. L. Swinney. Long-wavelength instability in surface-tension-driven Bénard convection. *Physical Review Letters*, 75(24):4397–4400, December 1995.

[129] S. J. VanHook, M. F. Schatz, J. B. Swift, W. D. McCormick, and H. L. Swinney. Long-wavelength surface-tension-driven Bénard convection: experiment and theory. *J. Fluid Mech.*, 345:45–78, 1997.

[130] Ženíšek. Polynomial approximation on tetrahedrons in the finite element method. *J. Approx. Theory*, 7:334–351, 1973.

[131] J.-S. Wang and N. Ida. Eigenvalue analysis in electromagnetic caveties using divergence-free finite elements. *IEEE Transactions on Magnetics*, 27(5):3978–3981, September 1991.

[132] Shun-Lien Wang and Robert F. Sekerka. Computation of the dendritic operating state at large supercoolings by the phase field model. *Phys. Rev. E*, 53:3760–3776, 1996.

[133] X. Wang and G. F. Carey. Finite-element simulation of a heated thin fluid layer. *Numerical Heat Transfer, Part A*, 45:841–867, 2004.

[134] X. Wang and G. F. Carey. On Marangoni effects in a heated thin fluid layer with a monolayer surfactant. part I: model development and stability analysis. *Int. J. Numer. Meth. Fluids*, 48:1–16, 2005.

[135] X. Wang and G. F. Carey. On Marangoni effects in a heated thin fluid layer with a monolayer surfactant. part II: finite element formulation and numerical studies. *Int. J. Numer. Meth. Fluids*, 48:17–42, 2005.

[136] S. H. Weintraub. *Differential Forms: A Complement to Vector Calculus.* Academic Press, San Diego, 1997.

[137] G. N. Wells, E. Kuhl, and K. Garikipati. A discontinuous Galerkin method for the Cahn-Hilliard equation. *J. Comp. Phys.*, 218:860–877, 2006.

[138] A. A. Wheeler, B. T. Murray, and R. J. Schaefer. Computation of dendrites using a phase field model. *Physica D*, 66(1-2):243–262, 1993.

[139] D. R. White, L. Mingwu, S. E. Benzley, and G. D. Sjaardema. Automated hexahedral mesh generation by virtual decomposition. In *Proceedings of the 4th International Meshing Roundtable*, pages 165–176, October 1995.

[140] S. M. Wise, J. S. Lowengrub, H. B. Frieboes, and V. Cristini. Three-dimensional multispecies nonlinear tumor growth - i. model and numerical method. *J. Theoretical Biology, in press*, 2008.

[141] A. J. Worsey and G. Farin. An n-dimensional Clough-Tocher interpolant. *Constructive Approximation*, 3:99–110, 1987.

[142] A. J. Worsey and B. Piper. A trivariate Powell-Sabin interpolant. *Computer Aided Geometric Design*, 5:177–186, 1988.

[143] Y. Xingde and C. Xiaoliang. The Fourier spectral method for the Cahn-Hilliard equation. *Appl. Math. and Computation*, 171(1):345–357, December 2005.

[144] H.-C. Yu and W. Lu. Dynamics of the self-assembly of nanovoids and nanobubbles in solids. *Acta Materialia*, 53:1799–1807, 2005.

[145] P. Zhao, J. C. Heinrich, and D. R. Poirier. Fixed mesh front-tracking methodology for finite element simulations. *Int. J. Num. Meth. Eng.*, 61:928–948, 2004.

[146] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique. *Int. J. Num. Meth. Eng.*, 33:1331–1364, 1992.

[147] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery and a posteriori error estimates. part 2: Error estimates and adaptivity. *Int. J. Num. Meth. Eng.*, 33:1365–1382, 1992.

Vita

Roy Hulen Stogner was born in Albuquerque, New Mexico on 6 July 1979, the son of Steve A. Stogner and Jane M. Stogner. He received the degree of Bachelor of Science in Mechanical Engineering from Rice University in 2001, and entered graduate study in the Computational Fluid Dynamics Laboratory at the University of Texas at Austin with the CAM Fellowship. He is currently a lead developer of the open source `libMesh` finite element library. After graduation he will join the Center for Predictive Engineering and Computational Sciences (PECOS) as a Postdoctoral Fellow.

Permanent address: 7131 Wood Hollow Dr. #146
Austin, Texas 78731

This dissertation was typeset with LaTeX[†] by the author.

[†] LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.