

The Finite Element Method in Heat Transfer and Fluid Dynamics

Third Edition

**J. N. Reddy
D. K. Gartling**



CRC Press
Taylor & Francis Group

The Finite Element Method in Heat Transfer and Fluid Dynamics

Third Edition

CRC Series in
COMPUTATIONAL MECHANICS
and **APPLIED ANALYSIS**

Series Editor: J. N. Reddy
Texas A&M University

Published Titles

ADVANCED THERMODYNAMICS ENGINEERING
Kalyan Annamalai and Ishwar K. Puri

APPLIED FUNCTIONAL ANALYSIS
J. Tinsley Oden and Leszek F. Demkowicz

COMBUSTION SCIENCE AND ENGINEERING
Kalyan Annamalai and Ishwar K. Puri

CONTINUUM MECHANICS FOR ENGINEERS, Third Edition
Thomas Mase, Ronald E. Smelser, and George E. Mase

EXACT SOLUTIONS FOR BUCKLING OF STRUCTURAL MEMBERS
C. M. Wang, C.Y. Wang, and J. N. Reddy

**THE FINITE ELEMENT METHOD IN HEAT TRANSFER AND FLUID
DYNAMICS, Third Edition**
J. N. Reddy and D.K. Gartling

**MECHANICS OF LAMINATED COMPOSITE PLATES AND SHELLS: THEORY
AND ANALYSIS, Second Edition**
J. N. Reddy

NUMERICAL AND ANALYTICAL METHODS WITH MATLAB®
William Bober, Chi-Tay Tsai, and Oren Masory

PRACTICAL ANALYSIS OF COMPOSITE LAMINATES
J. N. Reddy and Antonio Miravete

**SOLVING ORDINARY and PARTIAL BOUNDARY VALUE PROBLEMS
in SCIENCE and ENGINEERING**
Karel Rektorys

STRESSES IN BEAMS, PLATES, AND SHELLS, Third Edition
Ansel C. Ugural

The Finite Element Method in Heat Transfer and Fluid Dynamics

Third Edition

J. N. Reddy
D. K. Gartling



CRC Press
Taylor & Francis Group

Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

© 2010 by Taylor and Francis Group, LLC
CRC Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works

Printed in the United States of America on acid-free paper
10 9 8 7 6 5 4 3 2 1

International Standard Book Number-13: 978-1-4200-8599-0 (Ebook-PDF)

This book contains information obtained from authentic and highly regarded sources. Reasonable efforts have been made to publish reliable data and information, but the author and publisher cannot assume responsibility for the validity of all materials or the consequences of their use. The authors and publishers have attempted to trace the copyright holders of all material reproduced in this publication and apologize to copyright holders if permission to publish in this form has not been obtained. If any copyright material has not been acknowledged please write and let us know so we may rectify in any future reprint.

Except as permitted under U.S. Copyright Law, no part of this book may be reprinted, reproduced, transmitted, or utilized in any form by any electronic, mechanical, or other means, now known or hereafter invented, including photocopying, microfilming, and recording, or in any information storage or retrieval system, without written permission from the publishers.

For permission to photocopy or use material electronically from this work, please access www.copyright.com (<http://www.copyright.com/>) or contact the Copyright Clearance Center, Inc. (CCC), 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400. CCC is a not-for-profit organization that provides licenses and registration for a variety of users. For organizations that have been granted a photocopy license by the CCC, a separate system of payment has been arranged.

Trademark Notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation without intent to infringe.

Visit the Taylor & Francis Web site at
<http://www.taylorandfrancis.com>

and the CRC Press Web site at
<http://www.crcpress.com>

To our wives
Aruna and *Laura*

Contents

Preface to the Third Edition	xvii
Preface to the Second Edition	xix
Preface to the First Edition	xxi
About the Authors	xxiii
1. Equations of Heat Transfer and Fluid Mechanics	1
1.1 Introduction	1
1.1.1 Heat Transfer	1
1.1.2 Fluid Mechanics	2
1.2 Present Study	3
1.3 Mathematical Preliminaries	3
1.3.1 Vectors and Tensors	3
1.3.2 Index Notation and Summation Convention	5
1.3.3 The Del Operator and Calculus of Vectors and Tensors	7
1.4 Governing Equations of a Continuum	10
1.4.1 Introduction	10
1.4.2 Conservation of Mass; the Continuity Equation	10
1.4.3 Conservation of Momenta	11
1.4.4 Conservation of Energy	12
1.4.5 Equation of State	13
1.4.6 Constitutive Equations	14
1.4.7 Divergence and Advection Forms	15
1.5 Governing Equations in Terms of Primitive Variables	16
1.5.1 Vector Form	16
1.5.2 Cartesian Component Form	17
1.5.3 Cylindrical Component Form	17
1.5.4 Closure	19
1.6 Porous Flow Equations	19
1.7 Low-Speed Compressible Flow Equations	20
1.8 Auxiliary Transport Equations	22
1.9 Chemically Reacting Systems	23
1.10 Boundary Conditions	26
1.10.1 Viscous Flow Boundary Conditions	26
1.10.2 Porous Flow Boundary Conditions	29
1.10.3 Thermal and Transport Boundary Conditions	30
1.10.4 Initial Conditions	31

1.11 Change of Phase	32
1.12 Enclosure Radiation	34
1.13 Summary of Equations	36
Problems	37
References for Additional Reading.....	40
2. The Finite Element Method	43
2.1 Introduction	43
2.2 Model Differential Equation	44
2.3 Finite Element Approximation.....	45
2.4 Weighted-Integral Statements and Weak Forms	47
2.4.1 Preliminary Comments.....	47
2.4.2 Weak Form Development	47
2.5 Finite Element Model	50
2.6 Interpolation Functions.....	52
2.6.1 Properties of Approximation Functions.....	52
2.6.2 Linear Triangular Element	52
2.6.3 Linear Rectangular Element.....	54
2.6.4 Evaluation of Boundary Integrals.....	55
2.7 Assembly of Elements	55
2.8 Time-Dependent Problems	58
2.8.1 Introduction	58
2.8.2 Semidiscretization	58
2.8.3 Temporal Approximation	60
2.9 Axisymmetric Problems	61
2.10 Library of Finite Elements	63
2.10.1 Introduction	63
2.10.2 Triangular Elements	63
2.10.3 Rectangular Elements.....	65
2.11 Numerical Integration	66
2.11.1 Preliminary Comments	66
2.11.2 Coordinate Transformations	68
2.11.3 Integration over a Master Rectangular Element.....	70
2.11.4 Integration over a Master Triangular Element	71
2.12 Modeling Considerations	72
2.12.1 Mesh Generation	72
2.12.2 Representation of Boundary Flux	74
2.12.3 Imposition of Boundary Conditions	74
2.13 Illustrative Examples.....	75
2.13.1 Example 1.....	76
2.13.1.1 Problem description	76
2.13.1.2 Solution by linear triangular elements.....	77
2.13.1.3 Solution by linear rectangular elements	79
2.13.1.4 Discussion of the results	79

2.13.2 Example 2	81
2.13.3 Example 3	82
Problems	83
References for Additional Reading.....	86
3. Conduction Heat Transfer	87
3.1 Introduction	87
3.2 Semidiscrete Finite Element Model	88
3.3 Interpolation Functions.....	90
3.3.1 Preliminary Comments.....	90
3.3.2 Hexahedral (Brick) Elements.....	91
3.3.3 Tetrahedral Elements	92
3.3.4 Prism Elements.....	93
3.3.5 Pyramid Elements	94
3.4 Numerical Integration	95
3.5 Computation of Surface Flux	96
3.6 Semidiscrete Finite Element Model	99
3.7 Solution of Nonlinear Equations	100
3.7.1 Preliminary Comments	100
3.7.2 Steady-State Problems	100
3.7.3 Transient Problems	102
3.7.3.1 General formulation	102
3.7.3.2 Predictor-corrector methods	105
3.7.3.3 Time step control	106
3.7.3.4 Initialization	107
3.7.3.5 Linear multi-step methods.....	107
3.7.3.6 Convergence and stability	108
3.7.3.7 Mode superposition methods	111
3.8 Radiation Solution Algorithms	113
3.9 Variable Properties.....	118
3.9.1 Temperature-Dependent Properties	118
3.9.2 Phase Change Properties	119
3.9.3 Anisotropic Properties	121
3.10 Post-Processing Operations.....	122
3.10.1 Heat Flux	122
3.10.2 Heat Flow Function	124
3.11 Advanced Topics in Conduction	125
3.11.1 Introduction	125
3.11.2 Specialty Elements	126
3.11.3 Computational Boundary Conditions.....	129
3.11.3.1 Contact boundary conditions.....	129
3.11.3.2 Multipoint constraints	132
3.11.3.3 Partially covered surfaces	133
3.11.4 Bulk Nodes.....	134
3.11.5 Reactive Materials.....	136

3.11.6 Material Motion	138
3.12 Example Problems	139
3.12.1 Introduction	139
3.12.2 Element Convergence	139
3.12.3 Conduction/Radiation Solution	141
3.12.4 Temperature-Dependent Conductivity	143
3.12.5 Anisotropic Conductivity	143
3.12.6 One-Dimensional Stefan Problem	145
3.12.7 Drag Bit Analysis	147
3.12.8 Brazing and Welding Analysis	149
3.12.9 Investment Casting	152
Problems	154
References for Additional Reading	155
4. Flows of Viscous Incompressible Fluids	161
4.1 Introduction	161
4.1.1 Background	161
4.1.2 Governing Equations	161
4.2 Mixed Finite Element Model	164
4.2.1 Weak Form	164
4.2.2 Finite Element Model	165
4.3 Penalty Finite Element Models	167
4.3.1 Introduction	167
4.3.2 Penalty Function Method	168
4.3.3 Reduced Integration Penalty Model	171
4.3.4 Consistent Penalty Model	171
4.4 Finite Element Models of Porous Flow	172
4.5 Computational Considerations	174
4.5.1 Properties of the Matrix Equations	174
4.5.2 Choice of Interpolation Functions	175
4.5.2.1 Quadrilateral elements (2-D)	176
4.5.2.2 Triangular elements (2-D)	179
4.5.2.3 Hexahedral elements (3-D)	179
4.5.2.4 Tetrahedral elements (3-D)	180
4.5.3 Evaluation of Element Matrices in Penalty Models	180
4.5.4 Pressure Calculation	181
4.5.5 Traction Boundary Conditions	184
4.6 Solution of Nonlinear Equations	186
4.6.1 General Discussion	186
4.6.2 Fully Coupled Solution Methods	189
4.6.2.1 Picard method	189
4.6.2.2 Newton's method	190
4.6.2.3 Modified and quasi-Newton methods	192
4.6.2.4 Continuation methods	192
4.6.3 Pressure Correction/Projection Methods	194

4.7 Time-Approximation Schemes	196
4.7.1 Preliminary Comments	196
4.7.2 Forward/Backward Euler Schemes	197
4.7.3 Adams–Bashforth/Trapezoid Rule	198
4.7.4 Implicit Integration and Time Step Control	198
4.7.5 Explicit Integration	199
4.8 Stabilized Methods	200
4.8.1 Preliminary Comments	200
4.8.2 Galerkin/Least-Squares Formulation	202
4.8.3 Polynomial Pressure Projection	204
4.8.4 Variational Multiscale Methods	205
4.9 Least-Squares Finite Element Models	212
4.9.1 Introduction	212
4.9.2 Governing Equations	215
4.9.3 Least-Squares Formulation	216
4.9.4 Finite Element Model	217
4.9.5 Computational Aspects	218
4.10 Post-Processing	219
4.10.1 Stress Computation	219
4.10.2 Stream Function Computation	221
4.10.3 Particle Tracking	223
4.11 Free Surface Flows	224
4.11.1 Preliminary Comments	224
4.11.2 Time-Independent Free Surfaces	224
4.11.3 Time-Dependent Free Surfaces	229
4.12 Turbulence	235
4.12.1 Preliminary Comments	235
4.12.2 Governing Equations	236
4.12.3 General Turbulence Models	237
4.12.3.1 Correlations	238
4.12.3.2 Integral methods	238
4.12.3.3 One-point closure	238
4.12.3.4 Two-point closure	238
4.12.3.5 Large eddy simulation	238
4.12.3.6 Direct numerical simulations (DNS)	239
4.12.4 One-Point Closure Turbulence Models	239
4.12.4.1 Zero-equation model	240
4.12.4.2 One-equation model	240
4.12.4.3 Two-equation model	241
4.12.5 Finite Element Modeling of Turbulence	242
4.12.5.1 Zero-equation model	242
4.12.5.2 Two-equation model	243
4.12.6 Variational Multiscale (VMS) Turbulence Modeling	244
4.13 Numerical Examples	247
4.13.1 Preliminary Comments	247
4.13.2 Fluid Squeezed between Parallel Plates	248

4.13.3 Flow of a Viscous Lubricant in a Slider Bearing	250
4.13.4 Wall-Driven 2-D Cavity Flow	252
4.13.5 Wall-Driven 3-D Cavity Flow	254
4.13.6 Evaluation of the EBE Iterative Solvers	256
4.13.7 Backward Facing Step	258
4.13.8 Flow Past a Submarine	260
4.13.9 Crystal Growth from the Melt	262
4.13.10 Mold Filling	263
4.13.11 Examples Using Least-Squares Finite Element Models	267
4.13.11.1 Kovasznay flow	267
4.13.11.2 Flow over a backward-facing step	269
4.13.11.3 Flow past a cylinder at low Reynolds number	270
Problems	273
References for Additional Reading	275
5. Coupled Fluid Flow and Heat Transfer	285
5.1 Introduction	285
5.2 Nonisothermal Incompressible Flows	286
5.2.1 Governing Equations	286
5.2.2 Boundary Conditions	288
5.3 Mixed Finite Element Model	289
5.4 Penalty Finite Element Model	293
5.4.1 Preliminary Comments	293
5.4.2 Reduced Integration Penalty Model	294
5.4.3 Consistent Penalty Model	295
5.5 Finite Element Models of Porous Flow	295
5.6 Nonisothermal, Low-Speed, Compressible Flows	297
5.6.1 Governing Equations	297
5.6.2 Boundary Conditions	299
5.6.3 Mixed Finite Element Model	299
5.7 Solution Methods	302
5.7.1 General Discussion	302
5.7.2 Newton's Method	303
5.7.3 Segregated Equation Methods	304
5.8 Convection with Change of Phase	306
5.9 Convection with Enclosure Radiation	308
5.10 Post-Computation of Heat Flux	308
5.11 Turbulent Heat Transfer	310
5.12 Chemically Reacting Systems	311
5.12.1 Preliminary Comments	311
5.12.2 Finite Element Modeling of Chemical Reactions	311
5.13 Numerical Examples	312
5.13.1 Preliminary Comments	312
5.13.2 Concentric Tube Flow	312

5.13.3 Tube Flow with Change of Phase	313
5.13.4 Heated Cavity – Boussinesq Model	314
5.13.5 Heated Cavity – Acoustically Filtered Model	316
5.13.6 Solar Receiver	317
5.13.7 Tube Bundle	320
5.13.8 Volumetrically Heated Fluid	322
5.13.9 Porous/Fluid Layer	322
5.13.10 Curing of an Epoxy	326
5.13.11 Heated Channel	329
5.13.12 Closure	331
References for Additional Reading	331
6. Flows of Non-Newtonian Fluids	335
6.1 Introduction	335
6.2 Governing Equations of Inelastic Fluids	336
6.2.1 Conservation Equations	336
6.2.2 Boundary Conditions	337
6.2.3 Constitutive Equations	338
6.2.3.1 Power-law model	339
6.2.3.2 Carreau model	340
6.2.3.3 Bingham model	340
6.3 Finite Element Models of Inelastic Fluids	341
6.3.1 Introduction	341
6.3.2 Mixed Model	341
6.3.3 Penalty Model	343
6.3.4 Matrix Evaluations	344
6.4 Solution Methods for Inelastic Fluids	346
6.5 Governing Equations of Viscoelastic Fluids	350
6.5.1 Conservation Equations	350
6.5.2 Constitutive Equations	351
6.5.2.1 Differential models	352
6.5.2.2 Integral models	355
6.5.3 Boundary Conditions	356
6.6 Finite Element Model of Differential Form	357
6.6.1 Preliminary Comments	357
6.6.2 Summary of Governing Equations	357
6.6.3 Finite Element Model	358
6.6.4 Solution Methods	362
6.7 Additional Models of Differential Form	363
6.7.1 Explicitly Elliptic Momentum Equation Method	364
6.7.2 Elastic Viscous Stress Splitting Method	365
6.8 Finite Element Model of Integral Form	367
6.9 Unresolved Problems	368
6.9.1 General Comments	368
6.9.2 Choice of Constitutive Equation	369

6.9.3 Uniqueness and Existence of Solutions	370
6.9.4 Numerical Algorithm Problems	370
6.9.5 Equation Change of Type.....	371
6.9.6 Closure.....	372
6.10 Numerical Examples	372
6.10.1 Preliminary Comments	372
6.10.2 Buoyancy Driven Flow in a Cavity.....	372
6.10.3 Driven Cavity Flow.....	374
6.10.4 Squeeze Film Flow.....	374
6.10.5 Time-Dependent Poiseuille Flow	377
6.10.6 Four-to-One Contraction Problem.....	380
Problems	381
References for Additional Reading.....	382
7. Multiphysics Problems	387
7.1 Introduction	387
7.2 Coupled Boundary Value Problems	387
7.3 Fluid Mechanics and Heat Transfer	388
7.3.1 Introduction	388
7.3.2 Continuum Equations.....	388
7.3.3 Finite Element Models.....	390
7.4 Solid Mechanics	390
7.4.1 Introduction	390
7.4.2 Kinematics of Deformation.....	391
7.4.2.1 Descriptions of motion	391
7.4.2.2 Displacement vector	393
7.4.2.3 Deformation gradient tensor	393
7.4.2.4 Green strain tensor	394
7.4.3 Kinetics	395
7.4.3.1 Stress measures.....	395
7.4.3.2 Equilibrium statements	395
7.4.4 Constitutive Relations	396
7.4.5 Boundary Conditions	397
7.4.6 Finite Element Models.....	397
7.4.7 Solution Methods.....	399
7.5 Electromagnetics	399
7.5.1 Introduction	399
7.5.2 Maxwell's Equations	400
7.5.2.1 Constitutive relations	400
7.5.2.2 Electromagnetic forces and volume heating.....	402
7.5.2.3 Quasi-static approximation	402
7.5.3 Electromagnetic Potentials.....	403
7.5.4 Boundary and Interface Conditions	405
7.5.5 Gauge Conditions.....	407

7.5.6 Static Field Problems	408
7.5.6.1 Electrostatics	408
7.5.6.2 Steady current flow	409
7.5.6.3 Magnetostatics	409
7.5.7 Finite Element Models for EM Fields	409
7.5.7.1 Quasi-static potential equations	409
7.5.7.2 Gauge condition	412
7.5.7.3 Static field equations	413
7.5.8 Solution Methods – EM Fields	414
7.6 Coupled Problems in Mechanics	415
7.6.1 Introduction	415
7.6.2 Heat Conduction – Viscous Fluid Interactions 1&2	416
7.6.3 Heat Conduction – Quasi-Static Solid Interactions 1&3	416
7.6.4 Heat Conduction – Electric Field Interactions 1&4	418
7.6.5 Heat Conduction – Electromagnetic Field Interactions 1&4&5 ..	418
7.6.6 Viscous Flow – Quasi-Static Solid Interactions 2&3	420
7.6.7 Viscous Flow – Electric Field Interactions 2&4	421
7.6.8 Viscous Flow – Electromagnetic Field Interactions 2&4&5 ..	422
7.6.9 Quasi-Static Solid – Electromagnetic Field Interactions 3&4&5 ..	423
7.7 Implementation of Coupled Algorithms	424
7.8 Numerical Examples	426
7.8.1 Introduction	426
7.8.2 Thermal-Stress Example	426
7.8.3 Thermal-Electromagnetic Example	428
7.8.4 Fluid-Solid Interaction Example	431
7.8.5 Fluid-Electromagnetic Example	432
References for Additional Reading	436
8. Parallel Processing	439
8.1 Introduction	439
8.2 Parallel Systems	440
8.2.1 Classification	440
8.2.1.1 Granularity of the processing elements	440
8.2.1.2 Topology of interconnections	440
8.2.1.3 Distribution of control across the processing elements ..	441
8.2.1.4 Memory access	442
8.2.2 Languages and Communication Utilities	442
8.2.3 Performance	443
8.2.3.1 Algorithmic efficiency	443
8.2.3.2 Actual/Beneficial efficiency	443
8.2.3.3 Scalability	444
8.3 FEM and Parallel Processing	444
8.3.1 Preliminary Comments	444
8.3.2 Generic FEM Steps	445
8.3.3 External Preprocessing	445
8.3.4 Internal Preprocessing	447

8.3.5 Solution Processing	447
8.3.5.1 Element matrices.....	447
8.3.5.2 Matrix solvers.....	448
8.3.5.3 Solution control.....	450
8.3.6 Internal Postprocessing	451
8.3.7 External Postprocessing.....	451
8.3.8 Other Parallel Issues.....	451
8.3.8.1 Nonlocal data.....	452
8.3.8.2 Multiphysics simulations	452
8.4 Summary	454
References for Additional Reading.....	454
Appendix A: Computer Program <i>FEM2DHT</i>	457
A.1 Introduction	457
A.2 Heat Transfer and Related Problems.....	457
A.3 Flows of Viscous Incompressible Fluids.....	458
A.4 Description of the Input Data.....	458
A.5 Source Listings of Selective Subroutines	469
Reference for Additional Reading	470
Appendix B: Solution of Linear Equations	477
B.1 Introduction	477
B.2 Direct Methods	478
B.3 Iterative Methods.....	479
B.3.1 General Comments.....	479
B.3.2 Solution Algorithms	479
References for Additional Reading.....	483
Appendix C: Fixed Point Methods and Contraction Mappings	485
C.1 Fixed Point Theorem	485
C.2 Chord Method.....	486
C.3 Newton's Method	487
C.4 The Newton–Raphson Method	488
C.5 Descent Methods	488
References for Additional Reading.....	489
Subject Index	491

Preface to the Third Edition

Computational fluid mechanics (CFD) and computational heat transfer (CHT) continue to evolve at a significant pace and have become an ever increasing presence in standard engineering design and analysis practice. The seemingly endless increase in computing power in both single processor and parallel environments has allowed realistic problems of significant complexity and fidelity to be routinely solved and utilized in technological advances. Commercial software has made rapid progress in providing a broad spectrum of analysis capabilities to a variety of industries. Though software is increasingly robust, accurate CFD and CHT simulations still require a knowledgeable user, with a background in both mechanics and numerical methods. The present edition of this book remains focused on providing the information required by an individual who is interested in good numerical methods for the study and understanding of fluid mechanics and heat transfer phenomena.

This book remains practical in scope and content with an emphasis on computational procedures that we have found effective on a wide spectrum of applications. Little, if any, material has been deleted from the second edition. New material has been added primarily in the first five chapters and reflects the research thrusts over the last eight years. Chapter 1 contains the general description of the boundary value problems of interest and has been expanded with a section on mathematical preliminaries and a section on low-speed compressible flows. Chapter 2 continues with the introduction of the finite element method and is essentially unchanged. The thermal conduction and radiation problem is discussed in great detail in Chapter 3, and has a new discussion of mode superposition methods and a more detailed account of radiation solution methods. The isothermal, viscous flow problem is the topic of Chapter 4. The section on stabilized methods has been expanded with more discussion of variational multiscale methods (VMM), and a new section on least-squares finite element models (LSFEM) has been added. Chapter 5 extends the finite element method to non-isothermal flows and now includes a section on the formulation of low-speed, compressible flows. Non-Newtonian flow problems, both inelastic and viscoelastic, are included in Chapter 6, which is largely unchanged. Chapter 7 remains focused on the formulations and algorithms for multidisciplinary problems involving fluid mechanics, heat transfer, solid mechanics, and electromagnetics; the last chapter is on parallel computing, including a general discussion of the parallel architecture and the implementation of finite element models. Organization of the text, equation numbering, references, and symbols retain the same style as used in the previous editions. References have been added as needed without the removal of older citations. We believe it is important to retain the historical record.

The authors thank the publisher for the opportunity to prepare this third edition. The first author thanks colleagues Drs. Vinu Unnikrishnan and Ginu Unnikrishnan for their help with the scanning of figures from the first two editions and preparing the subject index, and Feifei Cheng for proofreading of the manuscript. The second author again thanks his numerous present and former colleagues at Sandia National Laboratories who continue to present challenging engineering applications in applied and computational mechanics. Specific acknowledgments must go to Drs. Mike Glass, Rick Givler, Charles Hickox, Roy Hogan, Mario Martinez and Phil Sackinger for collaboration and assistance in much of the algorithm development and demonstration simulations cited in this work. Portions of this book are taken from research work performed at Sandia National Laboratories under Contract No. DE-AC04-94AL85000 awarded by the U.S. Department of Energy and are used with permission. The authors dedicate this book to their wives, Aruna and Laura, who have graciously tolerated the authors' preoccupation with the writing of the book.

J. N. Reddy
College Station, Texas
jnreddy@tamu.edu

D. K. Gartling
Albuquerque, New Mexico
dkgartl@sandia.gov

Preface to the Second Edition

In the six years since the first edition of this book appeared some significant changes have occurred in the area of computational mechanics in general, and in computational fluid mechanics and heat transfer in particular. Foremost among these changes has been the extraordinary increase in performance in desktop computing platforms and the arrival in significant numbers of parallel computers. This widespread availability of capable computing hardware has predictably lead to the increased demand for computer simulation of products and processes during the engineering design and manufacturing process. Our original thesis that the finite element method was very well suited to general purpose and commercial software continues to hold true, as numerous programs are now available for the simulation of all types of applied mechanics problems. The range of applications of finite element analysis in fluid mechanics and heat transfer has become quite remarkable with complex, realistic simulations being carried out on a routine basis. The combination of hardware performance and reliable finite element algorithms has made these advances possible. Another significant change in computational mechanics is the increase in multidisciplinary (multiphysics) problems and their solution via finite element methods. Again, the increase in hardware performance has contributed to these types of computationally intensive problems. However, the inroads made by the finite element method in all areas of mechanics have also had a positive influence on coupled analysis. The commonality of finite element formulation, approximation and solution among the various boundary value problems in mechanics eases considerably the contemplation of multiphysics solutions and software. A final change in the numerical simulation arena comes from the implementation side of the finite element method. The demand for software capability and reliability has increased in step with the hardware performance. The use of parallel computers has added to the complexity of the implementation. All of these attributes lead to the conclusion that finite element implementation, if done well, will require some significant knowledge from areas in computer science. The time of general purpose codes being developed and maintained by one or two individuals is past and multi-talented teams now provide the most modern software.

Our focus for the present edition of this book remains the same — the education of the individual who is interested in good numerical methods for the study of fluid mechanics and heat transfer phenomena. The text remains practical in scope and content with an emphasis on computational procedures that we have found effective on a wide spectrum of applications. Little, if any, material has been deleted from the first edition. New material has been added in almost all chapters along with some rearrangement of topics to improve overall clarity and maintain the step-wise addition of increasingly complex material. Chapter 1 contains the general description of the boundary value problems of interest and has been augmented with a section on chemically reactive systems and additional discussion of change of phase. Chapter 2 continues with the introduction of the finite element method

and is essentially unchanged. The thermal conduction and radiation problem is discussed in great detail in Chapter 3, and it has new sections covering specialized finite elements and advanced topics in thermal analysis. The advanced topics section includes descriptions of difficult boundary conditions, such as multipoint constraints, contact and bulk nodes, material motion and kinematics, and methods for chemically reactive solids. The isothermal, viscous flow problem is the topic of Chapter 4. New sections in this chapter cover stabilized finite element methods and a general discussion of methods for free surface problems; the section on turbulence modeling has also been moved to this chapter. Chapter 5 extends the finite element method to non-isothermal flows and is largely unchanged. Non-Newtonian flow problems, both inelastic and viscoelastic, are now included in a revised and updated Chapter 6. A completely new Chapter 7 is focused on formulations and algorithms for multidisciplinary problems involving fluid mechanics, heat transfer, solid mechanics, and electromagnetics. This chapter outlines many of the possible types of coupling, describes the finite element equations for each mechanics area, and presents a number of realistic numerical examples. The last chapter on advanced topics is now devoted exclusively to a discussion of parallel computing including some general discussion of the parallel architecture and sections on parallel implementation of finite element models. Organization of the text, equation numbering, references, and symbols retain the same style as used in the first edition.

The second author thanks his numerous present and former colleagues at Sandia National Laboratories who continue to provide a wealth of challenging problems in applied and computational mechanics. Specific acknowledgments must go to Drs. Mike Glass, Rick Givler, Charles Hickox, Roy Hogan and Phil Sackinger for collaboration and assistance in much of the algorithm development and demonstration simulations cited in this work. Portions of this book are adapted from work performed at Sandia National Laboratories under Contract No. DE-AC04-94AL85000 awarded by the U.S. Department of Energy and are used with permission. The authors dedicate this book to their wives, Aruna and Laura, who have graciously tolerated the authors' preoccupation with the writing of the book.

J. N. Reddy
College Station, Texas

D. K. Gartling
Albuquerque, New Mexico

Preface to the First Edition

The numerical simulation of fluid mechanics and heat transfer problems has become a routine part of engineering practice as well as a focus for fundamental and applied research. Though there are still various topical areas where our physical understanding and/or ineffective numerical algorithms limit the investigation, a large number of complex phenomena can now be confidently studied via numerical simulation. Though finite difference methods have and will continue to play a major role in computational fluid dynamics (CFD) and heat transfer, finite element techniques have spurred the explosive development of “general purpose” methods and the growth of commercial software. The inherent strengths of the finite element method such as unstructured meshes, element-by-element formulation and processing, and the simplicity and rigor of boundary condition application are being coupled with modern developments in automatic mesh generation, adaptive meshing, and improved solution techniques to produce accurate and reliable simulation packages that are widely accessible. Improvements in computer hardware and system software (e.g., powerful workstations and window environments) have contributed significantly to streamlining the numerical simulation process. The finite element method in fluid mechanics and heat transfer has rapidly caught up with the well-established solid mechanics community in simulation capabilities.

As in any rapidly developing field, the education of the non-expert user community is of primary importance. The present text is an attempt to fill a need for those interested in using the finite element method in the study of fluid mechanics and heat transfer. It is a pragmatic book that views numerical computation as a means to an end—we do not dwell on theory or proof. Other fundamental and theoretical textbooks that cover these aspects are available or anticipated. The emphasis here is on presenting a useful methodology for a limited but significant class of problems dealing with heat conduction, incompressible viscous flows, and convection heat transfer.

The text has been developed out of our experience and course notes used in teaching graduate courses and continuing education courses to a wide spectrum of students. To gain the most from the book the student should have a reasonable background in fluid mechanics and heat transfer as would normally be found in most mechanical, aerospace, chemical or engineering mechanics curriculums. An introductory knowledge of finite element techniques would be very helpful but not essential; some familiarity with basic numerical analysis, linear algebra, and numerical integration would also be of assistance.

Our approach to the finite element method for fluid mechanics and heat transfer has been designed as a series of incremental steps of increasing complexity. In Chapter 1, the continuum boundary value problems that form the central focus of the book are described in some detail. We have tried to be as general as possible in describing the varied physical phenomena that may be encountered within the limits of non-isothermal, incompressible, viscous flows. Chapter 2

introduces the finite element method by application to a simplified, two-dimensional heat conduction problem. All of the necessary machinery for constructing weak forms of a partial differential equation and building a finite element model are introduced here and demonstrated by application. Chapter 3 recaps parts of Chapter 2 and extends the finite element method to three dimensions, time dependence, and practical applications in conduction heat transfer. Isothermal viscous fluid mechanics formulations are described in Chapter 4 along with a significant section on the solution of nonlinear equations developed from the flow problem. Chapter 5 extends the viscous flow problem to consider convective heat transfer formulations and applications. Inelastic non-Newtonian flows and free surface problems are covered in Chapter 6. The complex topic of viscoelastic flow simulation is surveyed in Chapter 7. The last chapter concludes the text with a survey of several advanced topics, including turbulence modeling. The coverage of each topic is sufficient to allow the reader to understand the basic methodology, use existing simulation software with confidence, and allow development of some simpler, special purpose computer codes. Example problems ranging from simple benchmarks to practical engineering solutions are included with each topical area. Adequate references to the relevant literature have also been included for those desiring a more encyclopedic coverage of a specific topic.

The text is organized into major sections within each chapter. Equations are numbered consecutively within each major section. Within a section, reference to an equation is by its sequential number; references to equations outside the current section have a full section, equation number citation. Vectors, tensors, and matrices are denoted by boldface letters. The vectors of interpolation (shape) functions in this book are denoted by Greek symbols (Ψ , Θ , Φ). Bibliographic information for literature cited in the text is numbered sequentially within each chapter and collected at the end of the chapter.

The first author would like to thank M. S. Ravisankar for his help with Chapter 8 topics and Praveen Gramma for reading the manuscript. The second author would like to thank his numerous present and former colleagues at Sandia National Laboratories who have provided a seemingly endless stream of challenging problems and taught him much about the practice of computational mechanics. Specific acknowledgments must go to Drs. Charles Hickox, Rick Givler, Roy Hogan, Phil Sackinger, Randy Schunk, Rekha Rao, and Steve Rottler for their suggestions and comments on early versions of the text. Portions of this book are adapted from work performed at Sandia National Laboratories under Contract No. DE-AC04-76DP00789 awarded by the U.S. Department of Energy and are used with permission.

J. N. Reddy
College Station, Texas

D. K. Gartling
Albuquerque, New Mexico

About the Authors

J. N. Reddy earned a Ph.D. in Engineering Mechanics from the University of Alabama in Huntsville, worked as a Postdoctoral Fellow at the University of Texas at Austin, was Research Scientist for Lockheed Missiles and Space Company during 1974–75, and taught at the University of Oklahoma from 1975 to 1980 and Virginia Polytechnic Institute & State University from 1980 to 1992. Currently, he is a Distinguished Professor and the inaugural holder of the Oscar S. Wyatt Endowed Chair at Texas A&M University, College Station. Dr. Reddy has published over 400 journal papers and 16 textbooks on theoretical formulations and numerical simulations of problems in solid and structural mechanics, computational fluid dynamics, numerical heat transfer, computational biology, geology and geophysics, mechanics of nanosystems, and applied mathematics.

Dr. Reddy is the recipient of numerous honors and awards, including the 1998 Nathan M. Newmark Medal from the American Society of Civil Engineers, the 2003 Computational Solid Mechanics award from the U.S. Association of Computational Mechanics, the 2004 Distinguished Research Award from the American Society for Composites, and an honorary degree (*Honoris Causa*) from the Technical University of Lisbon, Portugal (2009). Dr. Reddy is a fellow of the American Academy of Mechanics, the American Institute of Aeronautics and Astronautics, the American Society of Civil Engineers, the American Society of Mechanical Engineers, the American Society for Composites, International Association of Computational Mechanics, U.S. Association of Computational Mechanics, the Aeronautical Society of India, and the Institution of Structural Engineers, U.K. Dr. Reddy serves on the editorial boards of two dozen journals, and as the Editor-in-Chief of *Applied Mechanics Reviews*, *Mechanics of Advanced Materials and Structures*, *International Journal of Computational Methods in Engineering Science and Mechanics*, and *International Journal of Structural Stability and Dynamics*.

As a result of his extensive publications of archival journal papers and books, Dr. Reddy is recognized by ISI Highly Cited Researchers with over 10,000 citations and an H-index of over 40 to his credit. A more complete resume with links to journal papers can be found at <http://authors.isihighlycited.com/> or <http://www.tamu.edu/acml>.

David K. Gartling is a Senior Scientist in the Engineering Sciences Center at Sandia National Laboratories, Albuquerque, New Mexico. He earned his B.S. and M.S. in Aerospace Engineering at the University of Texas at Austin and completed the diploma course at the von Kármán Institute for Fluid Dynamics in Brussels, Belgium. After completion of his Ph.D. in Aerospace Engineering at the University of Texas at Austin, he joined the technical staff at Sandia National Laboratories. Dr. Gartling was a Visiting Associate Professor in the Mechanical Engineering Department at the University of Sydney, Australia, under a Fulbright Fellowship, and later he was a Supervisor in the Fluid and Thermal Sciences Department at Sandia National Laboratories. Dr. Gartling has published numerous papers dealing with finite element model development and finite element analysis of heat transfer and fluid dynamics problems of practical importance. He is the recipient of the 2001 Computational Fluid Dynamics Award from the U.S. Association of Computational Mechanics and is a fellow of the American Society of Mechanical Engineers. Dr. Gartling is presently a member of several professional societies, serves on the editorial boards of several journals, and is the Co-Editor of *International Journal for Numerical Methods in Fluids*.

Equations of Heat Transfer and Fluid Mechanics

1.1 Introduction

The disciplines of heat transfer and fluid dynamics are subjects that are closely coupled and form the core of many engineering and scientific studies. Areas as diverse as aerodynamics, meteorology, geology and geophysics, biology, material science and manufacturing all rely heavily on the ability to predict, understand and control complex fluid and thermal systems. In order to properly outline the scope of the present text, it is important to first provide brief definitions of the general areas of heat transfer and fluid dynamics.

1.1.1 Heat Transfer

Heat transfer is a branch of engineering that deals with the transfer of thermal energy from one point to another within a medium or from one medium to another due to the occurrence of a temperature difference. Heat transfer may take place in one or more of its three basic forms: conduction, convection, and radiation. The transfer of heat within a medium due to a diffusion process is called conduction heat transfer. The Fourier heat conduction law states that the heat flow is proportional to the temperature gradient. The coefficient of proportionality is a material parameter known as the *thermal conductivity* which may be a function of a number of variables.

Convection heat transfer is usually defined as energy transport effected by the motion of a fluid. The convection heat transfer between two dissimilar media is governed by Newton's law of cooling. It states that the heat flow is proportional to the difference of the temperatures of the two media. The proportionality coefficient is called the *convection heat transfer coefficient* or *film conductance*.

Thermal radiation is defined as radiant (electromagnetic) energy emitted by a medium and is due solely to the temperature of the medium. Radiant energy exchange between surfaces or between a region and its surroundings is described by the Stefan–Boltzmann law, which states that the radiant energy transmitted is proportional to the difference of the fourth power of the temperatures of the surfaces. The proportionality parameter is known as the *Stefan–Boltzmann constant*.

1.1.2 Fluid Mechanics

Fluid mechanics is one of the oldest branches of physics, and is concerned with the motion of gases and liquids and their interaction with the surroundings. For example, the flight of birds in the air and the motion of fish in the water can be understood by the principles of fluid mechanics. Such understanding helps us design airplanes and ships. The formation of tornadoes, hurricanes, and thunderstorms can also be explained with the help of the equations of fluid mechanics.

A fluid state of matter, as opposed to the solid state of matter, is characterized by the relative mobility of its molecules. Very strong intermolecular attractive forces exist in solids which are responsible for the property of relative rigidity (or stiffness) in solids. The intermolecular forces are weaker in liquids and extremely small in gases. The stress in a solid body is proportional to the strain (i.e., deformation per unit length), while the stress in a fluid is proportional to the time rate of strain (i.e., rate of deformation). The proportionality parameter in the case of fluids is known as the *viscosity*. It is a measure of the intermolecular forces exerted as layers of fluid attempt to slide past one another. The viscosity of a fluid, in general, is a function of the thermodynamic state of the fluid and in some cases the strain rate.

Fluid mechanics is a very broad area and is traditionally divided into smaller topical areas based on characteristics of the fluid properties or the basic nature of the flow. An *inviscid fluid* is one where the viscosity is assumed to be zero. An *incompressible fluid* is one with constant density, and an *incompressible flow* is one in which density variations (compared to a reference density) are negligible. An inviscid and incompressible fluid is termed an *ideal* or a *perfect* fluid. A *real fluid* is one with finite viscosity, and it may or may not be incompressible. When the viscosity of a fluid depends only on thermodynamic properties, and the stress is linearly related to the strain rate, the fluid is said to be *Newtonian*. A *non-Newtonian fluid* is one which does not obey the Newtonian (i.e., linear) stress-strain rate relation. A non-Newtonian constitutive relation can be of algebraic (e.g., power-law), differential, or integral type.

The flow of viscous fluids can also be classified into two major types: a smooth, orderly motion is called *laminar flow*, and a random, fluctuating motion is called *turbulent flow*. The nature of a viscous flow can be characterized by a nondimensional parameter known as the *Reynolds number*, $Re = \rho UL/\mu$, which is defined as the ratio of inertial forces ρU^2 to viscous forces $\mu U/L$. Here ρ denotes the density of the fluid, U the characteristic flow velocity, μ is the fluid viscosity, and L is a characteristic dimension of the flow region. High viscosity fluids and/or small velocities produce relatively small Reynolds numbers and a laminar flow. The flow of less viscous fluids and/or higher velocities lead to higher Reynolds numbers and a turbulent flow. Transitional flows contain regions of both laminar and turbulent flows.

At the outset one must note that fluid mechanics is seldom concerned with fluids alone but rather with effects of fluids on the surroundings or systems with which the fluids come in contact. Therefore, interaction of fluids with their surroundings is of considerable interest in practice, especially when the surroundings are “reactive” (e.g., deform or change their configuration).

1.2 Present Study

The purpose of the present book is to describe the use of the finite element method for the solution of engineering problems in well-defined areas of heat transfer and fluid mechanics. In particular, we will concentrate on finite element computational procedures for the areas of conduction and convection heat transfer. Since convection relies heavily on fluid dynamics, the study of isothermal flows will play a prominent role in developing the overall subject. To focus the presentation, all fluid mechanics applications discussed here will be limited to the area of viscous, low-speed flows. The low-speed limit will be defined later in terms of the Mach number, $Ma = U/a$, where a is the speed of sound in the fluid and U is again, a characteristic fluid velocity. Radiation heat transfer will be discussed to the extent that it affects thermal boundary conditions and heat transfer in enclosures. In addition, nonisothermal flows in porous media will be considered as an extension of the viscous flow problem. Complexities such as change of phase, free surface boundaries, and non-Newtonian constitutive behavior will also be considered. Also, there is some description of the interaction between fluid and thermal problems and other areas of applied mechanics.

In the remainder of this chapter, we will present certain mathematical preliminaries (see Reddy [1]) that are useful in the sequel, and then provide a résumé of the equations governing heat transfer and fluid flows in a continuous medium. The boundary conditions for typical problems are also presented, and the special conditions associated with change-of-phase problems, free-surface flows, and enclosure-radiation are discussed.

1.3 Mathematical Preliminaries

1.3.1 Vectors and Tensors

The laws of nature are independent of the choice of a coordinate system and we may seek to represent the laws in a manner independent of a particular coordinate system. This can be done using *vectors* and *tensors* that obey certain rules of vector addition and multiplication by a scalar (see Chapter 2 of Reddy [1]). The use of vectors and tensors in formulating natural laws leaves them *invariant*, and one may express them in any chosen coordinate system.

In written or typed material, a vector is denoted by placing an arrow over the letter denoting the vector, such as \vec{A} . In printed material, the letter used for the vector is commonly denoted by a boldface letter, \mathbf{A} , such as used in this study. The magnitude of the vector \mathbf{A} is denoted by $|A|$, $\|\mathbf{A}\|$, or A . Magnitude of a vector is a scalar.

It is customary to denote the direction of a plane area by means of a unit vector drawn normal to that plane. The direction of the normal is taken by convention as that in which a right-handed screw advances as it is rotated according to the sense of travel along the boundary curve or contour. Let the unit normal vector be given by $\hat{\mathbf{n}}$. Then the area \mathbf{A} can be denoted by $\mathbf{A} = A \hat{\mathbf{n}}$.

Tensors are more general objects that are endowed with a magnitude and multiple direction(s) but satisfy the rules of “vector addition and scalar multiplication.” In fact, vectors are often termed the first-order tensors. For example, the stress vector \mathbf{t} , which is a measure of force per unit area, depends not only on the magnitude

and direction of the force but also on the orientation of the plane on which the force acts, as shown in Figure 1.3.1:

$$\mathbf{t}(\hat{\mathbf{n}}) = \lim_{\Delta a \rightarrow 0} \frac{\Delta \mathbf{f}(\hat{\mathbf{n}})}{\Delta a} \quad (1.3.1)$$

Thus, specification of the stress vector at a point requires *two* vectors, one perpendicular to the plane on which the force is acting and the other in the direction of the force. Hence, stress is a *second-order tensor*.

The analytical description of vectors is useful in expressing, for example, the laws of physics in analytical form. The analytical description of a vector is based on the notion of its components. In a three-dimensional space, a set of no more than three linearly independent vectors¹ can be found. Let us choose any set and denote it as follows:

$$\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3. \quad (1.3.2)$$

Each vector \mathbf{e}_i ($i = 1, 2, 3$) is called a basis vector and the set is called a *basis* (or a base system). A basis is called *orthonormal*, if \mathbf{e}_i are mutually orthogonal and have unit magnitudes. To distinguish the basis $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$ that is not orthonormal from one that is orthonormal, we denote the orthonormal basis by $(\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3)$, where

$$\begin{aligned} \hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_2 &= 0, & \hat{\mathbf{e}}_2 \cdot \hat{\mathbf{e}}_3 &= 0, & \hat{\mathbf{e}}_3 \cdot \hat{\mathbf{e}}_1 &= 0, \\ \hat{\mathbf{e}}_1 \cdot \hat{\mathbf{e}}_1 &= 1, & \hat{\mathbf{e}}_2 \cdot \hat{\mathbf{e}}_2 &= 1, & \hat{\mathbf{e}}_3 \cdot \hat{\mathbf{e}}_3 &= 1 \end{aligned} \quad (1.3.3)$$

Sometimes the notation $(\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}})$ or $(\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y, \hat{\mathbf{e}}_z)$ is used in place of $(\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3)$.

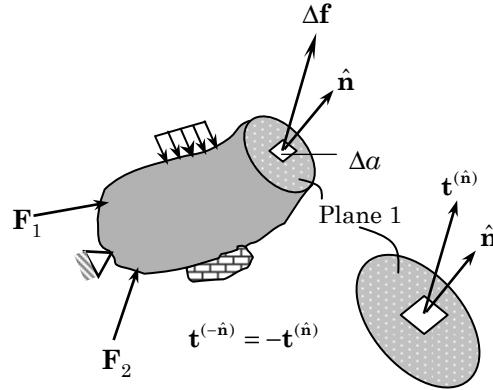


Figure 1.3.1: Definition of a stress vector acting on a plane with normal $\hat{\mathbf{n}}$.

¹ A set of n vectors, $\{A_1, A_2, \dots, A_n\}$, is said to be *linearly dependent* if a set of n numbers c_1, c_2, \dots, c_n can be found such that $c_1 A_1 + c_2 A_2 + \dots + c_n A_n = 0$, where c_1, c_2, \dots, c_n cannot all be zero. If this expression cannot be satisfied (that is, all c_i are zero), the set of vectors $\{A_1, A_2, \dots, A_n\}$ is said to be *linearly independent*.

Using the concept of linear independence of vectors, we can represent any vector in three-dimensional space as a linear combination of the basis vectors:

$$\mathbf{A} = A_x \hat{\mathbf{e}}_x + A_y \hat{\mathbf{e}}_y + A_z \hat{\mathbf{e}}_z = A_1 \hat{\mathbf{e}}_1 + A_2 \hat{\mathbf{e}}_2 + A_3 \hat{\mathbf{e}}_3. \quad (1.3.4)$$

The vectors $A_1 \hat{\mathbf{e}}_1$, $A_2 \hat{\mathbf{e}}_2$, and $A_3 \hat{\mathbf{e}}_3$ are called the *vector components* of \mathbf{A} , and A_1 , A_2 , and A_3 are called *scalar components* of \mathbf{A} associated with the basis $(\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3)$.

A second-order tensor, also called a *dyad*, requires two basis vectors to represent it. We can display all of the components of a second-order tensor Φ as

$$\begin{aligned} \Phi &= \phi_{11} \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_1 + \phi_{12} \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_2 + \phi_{13} \hat{\mathbf{e}}_1 \hat{\mathbf{e}}_3 \\ &\quad + \phi_{21} \hat{\mathbf{e}}_2 \hat{\mathbf{e}}_1 + \phi_{22} \hat{\mathbf{e}}_2 \hat{\mathbf{e}}_2 + \phi_{23} \hat{\mathbf{e}}_2 \hat{\mathbf{e}}_3 \\ &\quad + \phi_{31} \hat{\mathbf{e}}_3 \hat{\mathbf{e}}_1 + \phi_{32} \hat{\mathbf{e}}_3 \hat{\mathbf{e}}_2 + \phi_{33} \hat{\mathbf{e}}_3 \hat{\mathbf{e}}_3. \end{aligned} \quad (1.3.5)$$

Equation (1.3.5) illustrates that a second-order tensor in three-dimensional space has nine independent components in general, each component associated with a certain dyad pair. The components are thus said to be ordered. When the ordering is understood, such as suggested by the form (1.3.5), the explicit writing of the dyads can be suppressed and the dyad is written as an array (or matrix):

$$[\Phi] = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} \\ \phi_{21} & \phi_{22} & \phi_{23} \\ \phi_{31} & \phi_{32} & \phi_{33} \end{bmatrix} \text{ and } \Phi = \left\{ \begin{array}{c} \hat{\mathbf{e}}_1 \\ \hat{\mathbf{e}}_2 \\ \hat{\mathbf{e}}_3 \end{array} \right\}^T [\Phi] \left\{ \begin{array}{c} \hat{\mathbf{e}}_1 \\ \hat{\mathbf{e}}_2 \\ \hat{\mathbf{e}}_3 \end{array} \right\} \quad (1.3.6)$$

This representation is simpler than Eq. (1.3.5), but it is taken to mean the same.

1.3.2 Index Notation and Summation Convention

Use of index notation facilitates writing long expressions in succinct form. For example, consider the component form of vector \mathbf{A} in Eq. (1.3.4), which can be abbreviated as

$$\mathbf{A} = \sum_{i=1}^3 A_i \mathbf{e}_i \text{ or } \mathbf{A} = \sum_{j=1}^3 A_j \mathbf{e}_j.$$

If we had chosen the notation

$$\mathbf{A} = A_x \mathbf{e}_x + A_y \mathbf{e}_y + A_z \mathbf{e}_z$$

where (A_x, A_y, A_z) are the same components as (A_1, A_2, A_3) and the basis $(\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)$ is the same as $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$, it is not possible to write it with the summation convention. The summation index i or j is arbitrary as long as the same index is used for both A and \mathbf{e} . The expression can be further shortened by omitting the summation sign and having the understanding that a repeated index means summation over all values of that index. Thus, the three-term expression $A_1 \mathbf{e}_1 + A_2 \mathbf{e}_2 + A_3 \mathbf{e}_3$ can be simply written as

$$\mathbf{A} = A_i \mathbf{e}_i \quad (1.3.7)$$

This notation is called the *summation convention*. The summation convention allows us to write several expressions or equations in a single statement.

6 THE FINITE ELEMENT METHOD IN HEAT TRANSFER AND FLUID DYNAMICS

The repeated index in (1.3.7) is called a *dummy index* because it can be replaced by *any other symbol that has not already been used* in that expression. Thus, the expression in Eq. (1.3.7) can also be written as

$$\mathbf{A} = A_i \mathbf{e}_i = A_j \mathbf{e}_j = A_m \mathbf{e}_m \quad (1.3.8)$$

and so on. As a rule, no index must appear more than twice in an expression. A *free index* is one that does not repeat and appears in every expression of an equation, except for expressions that contain scalars only.

It is convenient to introduce the Kronecker delta δ_{ij} and alternating symbol e_{ijk} because they allow easy representation of the dot product (or scalar product) and cross product, respectively, of orthonormal vectors in a right-handed basis system. We define the dot product $\hat{\mathbf{e}}_i \cdot \hat{\mathbf{e}}_j$ as

$$\hat{\mathbf{e}}_i \cdot \hat{\mathbf{e}}_j = \delta_{ij} \quad (1.3.9)$$

where

$$\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} \quad (1.3.10)$$

The Kronecker delta δ_{ij} , due to its definition, modifies (or contracts) the subscripts in the coefficients of an expression in which it appears

$$A_i \delta_{ij} = A_j, \quad A_i B_j \delta_{ij} = A_i B_i = A_j B_j, \quad \delta_{ij} \delta_{ik} = \delta_{jk}$$

We define the cross product $\hat{\mathbf{e}}_i \times \hat{\mathbf{e}}_j$ as

$$\hat{\mathbf{e}}_i \times \hat{\mathbf{e}}_j \equiv e_{ijk} \hat{\mathbf{e}}_k \quad \text{or} \quad e_{ijk} = \hat{\mathbf{e}}_i \times \hat{\mathbf{e}}_j \cdot \hat{\mathbf{e}}_k = \hat{\mathbf{e}}_i \cdot \hat{\mathbf{e}}_j \times \hat{\mathbf{e}}_k, \quad (1.3.11)$$

where

$$e_{ijk} = \begin{cases} 1, & \text{if } i, j, k \text{ are in cyclic order} \\ & \text{and not repeated } (i \neq j \neq k), \\ -1, & \text{if } i, j, k \text{ are not in cyclic order} \\ & \text{and not repeated } (i \neq j \neq k), \\ 0, & \text{if any of } i, j, k \text{ are repeated.} \end{cases} \quad (1.3.12)$$

The symbol e_{ijk} is called the *alternating symbol* or *permutation symbol*. By definition, the subscripts of the permutation symbol can be permuted without changing its value; an interchange of any two subscripts will change the sign (hence, interchange of two subscripts twice keeps the value unchanged):

$$e_{ijk} = e_{kij} = e_{jki}, \quad e_{ijk} = -e_{jik} = e_{jki} = -e_{kji}.$$

In an orthonormal basis $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$, the scalar and vector products can be expressed in the index notation using the Kronecker delta and the alternating symbols:

$$\begin{aligned} \mathbf{A} \cdot \mathbf{B} &= (A_i \hat{\mathbf{e}}_i) \cdot (B_j \hat{\mathbf{e}}_j) = A_i B_j \delta_{ij} = A_i B_i, \\ \mathbf{A} \times \mathbf{B} &= (A_i \hat{\mathbf{e}}_i) \times (B_j \hat{\mathbf{e}}_j) = A_i B_j e_{ijk} \hat{\mathbf{e}}_k. \end{aligned} \quad (1.3.13)$$

Note that the components of a vector in an orthonormal coordinate system can be expressed as

$$A_i = \mathbf{A} \cdot \hat{\mathbf{e}}_i \quad (1.3.14)$$

and therefore we can express vector \mathbf{A} as

$$\mathbf{A} = A_i \hat{\mathbf{e}}_i = (\mathbf{A} \cdot \hat{\mathbf{e}}_i) \hat{\mathbf{e}}_i \quad (1.3.15)$$

Further, the Kronecker delta and the permutation symbol are related by the identity, known as the *e-δ identity*,

$$e_{ijk} e_{imn} = \delta_{jm} \delta_{kn} - \delta_{jn} \delta_{km}. \quad (1.3.16)$$

1.3.3 The Del Operator and Calculus of Vectors and Tensors

Let us denote a scalar field by $\phi = \phi(\mathbf{x})$, \mathbf{x} being the position vector in a rectangular Cartesian system (x_1, x_2, x_3) . Let us now denote a differential element with $d\mathbf{x}$ and its magnitude by $ds \equiv |d\mathbf{x}|$. Then $\hat{\mathbf{e}} = d\mathbf{x}/ds$ is a unit vector in the direction of $d\mathbf{x}$, and we may express it as (by the use of the chain rule of differentiation)

$$\left(\frac{d\phi}{ds} \right)_{\hat{\mathbf{e}}} = \frac{d\mathbf{x}}{ds} \cdot \frac{\partial \phi}{\partial \mathbf{x}} = \hat{\mathbf{e}} \cdot \left(\hat{\mathbf{e}}_1 \frac{\partial \phi}{\partial x_1} + \hat{\mathbf{e}}_2 \frac{\partial \phi}{\partial x_2} + \hat{\mathbf{e}}_3 \frac{\partial \phi}{\partial x_3} \right). \quad (1.3.17)$$

The derivative $(d\phi/ds)_{\hat{\mathbf{e}}}$ is called the *directional derivative* of ϕ . We see that it is the *rate of change* of ϕ with respect to distance and that it depends on the direction $\hat{\mathbf{e}}$ in which the distance is taken. The vector $\partial\phi/\partial\mathbf{x}$ is called the *gradient vector* and is denoted by $\text{grad } \phi$:

$$\text{grad } \phi \equiv \hat{\mathbf{e}}_1 \frac{\partial \phi}{\partial x_1} + \hat{\mathbf{e}}_2 \frac{\partial \phi}{\partial x_2} + \hat{\mathbf{e}}_3 \frac{\partial \phi}{\partial x_3} = \hat{\mathbf{e}}_x \frac{\partial \phi}{\partial x} + \hat{\mathbf{e}}_y \frac{\partial \phi}{\partial y} + \hat{\mathbf{e}}_z \frac{\partial \phi}{\partial z} \quad (1.3.18)$$

We interpret $\text{grad } \phi$ as some operator operating on function ϕ , that is, $\text{grad } \phi \equiv \nabla \phi$. This operator is expressed as

$$\nabla \equiv \hat{\mathbf{e}}_1 \frac{\partial}{\partial x_1} + \hat{\mathbf{e}}_2 \frac{\partial}{\partial x_2} + \hat{\mathbf{e}}_3 \frac{\partial}{\partial x_3} = \hat{\mathbf{e}}_x \frac{\partial}{\partial x} + \hat{\mathbf{e}}_y \frac{\partial}{\partial y} + \hat{\mathbf{e}}_z \frac{\partial}{\partial z}. \quad (1.3.19)$$

and is called the *del operator*. The del operator is a *vector differential* operator. It is important to note that whereas the del operator has some of the properties of a vector, it does not have them all, because it is an operator. For instance $\nabla \cdot \mathbf{A}$ is a scalar, called the *divergence of \mathbf{A}* ($\partial A_i / \partial x_i$), whereas $\mathbf{A} \cdot \nabla$ is a scalar differential operator [$A_i (\partial / \partial x_i)$]. Thus the del operator does not commute in this sense.

An important note is in order concerning the del operator. Two types of gradients are used in continuum mechanics: forward and backward gradients. The forward gradient is the usual gradient and backward gradient is the transpose of the forward gradient operator. To see the difference between the two types of gradients, consider a vector function $\mathbf{A} = A_i(\mathbf{x}) \hat{\mathbf{e}}_i$. The forward and backward gradients of \mathbf{A} are (both are second-order tensors)

$$\vec{\nabla} \mathbf{A} = \nabla \mathbf{A} = \hat{\mathbf{e}}_j \frac{\partial}{\partial x_j} (A_i \hat{\mathbf{e}}_i) = \frac{\partial A_i}{\partial x_j} \hat{\mathbf{e}}_j \hat{\mathbf{e}}_i = A_{i,j} \hat{\mathbf{e}}_j \hat{\mathbf{e}}_i, \quad (1.3.20)$$

$$\overleftarrow{\nabla} \mathbf{A} = (\nabla \mathbf{A})^T = \frac{\partial A_i}{\partial x_j} (\hat{\mathbf{e}}_j \hat{\mathbf{e}}_i)^T = A_{i,j} \hat{\mathbf{e}}_i \hat{\mathbf{e}}_j, \quad (1.3.21)$$

where $A_{i,j} = \partial A_i / \partial x_j$. The backward gradient is often used (without explanation) in defining the velocity gradient tensor.

The dot product of a del operator with a vector is called the *divergence of a vector* and denoted by

$$\nabla \cdot \mathbf{A} \equiv \operatorname{div} \mathbf{A}. \quad (1.3.22)$$

If we take the divergence of the gradient of a scalar function $\phi(\mathbf{x})$, we have

$$\operatorname{div}(\operatorname{grad} \phi) \equiv \nabla \cdot \nabla \phi = (\nabla \cdot \nabla) \phi = \nabla^2 \phi. \quad (1.3.23)$$

The notation $\nabla^2 = \nabla \cdot \nabla$ is called the *Laplacian operator*. In Cartesian systems this reduces to the simple form

$$\nabla^2 \phi = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + \frac{\partial^2 \phi}{\partial z^2} = \frac{\partial^2 \phi}{\partial x_i \partial x_i}. \quad (1.3.24)$$

The Laplacian of a scalar appears frequently in the partial differential equations governing physical phenomena.

The *curl of a vector* is defined as the del operator operating on a vector by means of the cross product:

$$\operatorname{curl} \mathbf{A} = \nabla \times \mathbf{A} = e_{ijk} \hat{\mathbf{e}}_i \frac{\partial A_k}{\partial x_j}. \quad (1.3.25)$$

Let $\hat{\mathbf{n}}$ denote the unit vector, taken positive outward, normal to the surface Γ of a continuous medium occupying the region Ω , as shown in Figure 1.3.2. In a Cartesian coordinate system, the unit normal vector can be expressed in terms of its components as

$$\hat{\mathbf{n}} = n_1 \hat{\mathbf{e}}_1 + n_2 \hat{\mathbf{e}}_2 + n_3 \hat{\mathbf{e}}_3 = n_x \hat{\mathbf{e}}_x + n_y \hat{\mathbf{e}}_y + n_z \hat{\mathbf{e}}_z \quad (1.3.26)$$

The components $(n_1, n_2, n_3) = (n_x, n_y, n_z)$ are called *direction cosines* because of the fact

$$n_i = \text{cosine of the angle between } \hat{\mathbf{n}} \text{ and the } x_i\text{-axis} \quad (1.3.27)$$

The quantity $\hat{\mathbf{n}} \cdot \operatorname{grad} \phi$ of a function ϕ is called the *normal derivative* of ϕ and it is denoted by

$$\frac{\partial \phi}{\partial n} \equiv \hat{\mathbf{n}} \cdot \operatorname{grad} \phi = \hat{\mathbf{n}} \cdot \nabla \phi. \quad (1.3.28)$$

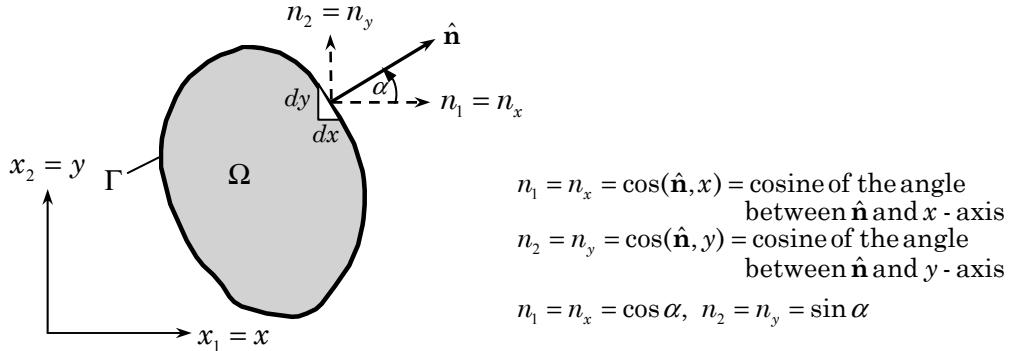


Figure 1.3.2: A unit vector normal to a surface.

Two commonly used orthogonal curvilinear coordinate systems are the *cylindrical* coordinate system and *spherical* coordinate system. Table 1.3.1 contains a summary of the basic information for the two coordinate systems (from Reddy [1]).

Table 1.3.1: Base vectors and del and Laplace operators in cylindrical and spherical coordinate systems.

Cylindrical coordinate system (r, θ, z)

$$x = r \cos \theta, y = r \sin \theta, z = z, \quad \mathbf{R} = r\hat{\mathbf{e}}_r + z\hat{\mathbf{e}}_z$$

$$\mathbf{A} = A_r\hat{\mathbf{e}}_r + A_\theta\hat{\mathbf{e}}_\theta + A_z\hat{\mathbf{e}}_z, \quad \hat{\mathbf{e}}_r = \cos \theta \hat{\mathbf{e}}_x + \sin \theta \hat{\mathbf{e}}_y, \\ \hat{\mathbf{e}}_\theta = -\sin \theta \hat{\mathbf{e}}_x + \cos \theta \hat{\mathbf{e}}_y, \quad \hat{\mathbf{e}}_z = \hat{\mathbf{e}}_z$$

$$\frac{\partial \hat{\mathbf{e}}_r}{\partial \theta} = -\sin \theta \hat{\mathbf{e}}_x + \cos \theta \hat{\mathbf{e}}_y = \hat{\mathbf{e}}_\theta,$$

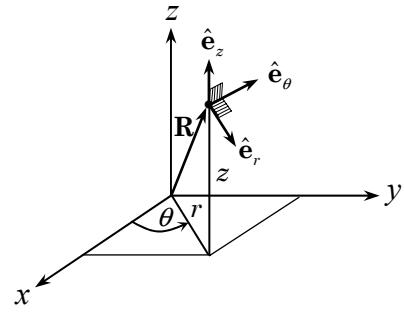
$$\frac{\partial \hat{\mathbf{e}}_\theta}{\partial \theta} = -\cos \theta \hat{\mathbf{e}}_x - \sin \theta \hat{\mathbf{e}}_y = -\hat{\mathbf{e}}_r$$

All other derivatives of the base vectors are zero.

$$\nabla = \hat{\mathbf{e}}_r \frac{\partial}{\partial r} + \frac{1}{r} \hat{\mathbf{e}}_\theta \frac{\partial}{\partial \theta} + \hat{\mathbf{e}}_z \frac{\partial}{\partial z}, \quad \nabla^2 = \frac{1}{r} \left[\frac{\partial}{\partial r} \left(r \frac{\partial}{\partial r} \right) + \frac{1}{r} \frac{\partial^2}{\partial \theta^2} + r \frac{\partial^2}{\partial z^2} \right]$$

$$\nabla \cdot \mathbf{A} = \frac{1}{r} \left[\frac{\partial(rA_r)}{\partial r} + \frac{\partial A_\theta}{\partial \theta} + r \frac{\partial A_z}{\partial z} \right]$$

$$\nabla \times \mathbf{A} = \left(\frac{1}{r} \frac{\partial A_z}{\partial \theta} - \frac{\partial A_\theta}{\partial z} \right) \hat{\mathbf{e}}_r + \left(\frac{\partial A_r}{\partial z} - \frac{\partial A_z}{\partial r} \right) \hat{\mathbf{e}}_\theta + \frac{1}{r} \left[\frac{\partial(rA_\theta)}{\partial r} - \frac{\partial A_r}{\partial \theta} \right] \hat{\mathbf{e}}_z$$



Spherical coordinate system (R, ϕ, θ)

$$x = R \sin \phi \cos \theta, \quad y = R \sin \phi \sin \theta, \quad z = R \cos \phi, \quad \mathbf{R} = R\hat{\mathbf{e}}_R$$

$$\mathbf{A} = A_R\hat{\mathbf{e}}_R + A_\phi\hat{\mathbf{e}}_\phi + A_\theta\hat{\mathbf{e}}_\theta \quad (\text{typical vector})$$

$$\hat{\mathbf{e}}_R = \sin \phi \cos \theta \hat{\mathbf{e}}_x + \sin \phi \sin \theta \hat{\mathbf{e}}_y + \cos \phi \hat{\mathbf{e}}_z$$

$$\hat{\mathbf{e}}_\phi = \cos \phi \cos \theta \hat{\mathbf{e}}_x + \cos \phi \sin \theta \hat{\mathbf{e}}_y - \sin \phi \hat{\mathbf{e}}_z$$

$$\hat{\mathbf{e}}_\theta = -\sin \theta \hat{\mathbf{e}}_x + \cos \theta \hat{\mathbf{e}}_y$$

$$\hat{\mathbf{e}}_x = \sin \phi \cos \theta \hat{\mathbf{e}}_R + \cos \phi \cos \theta \hat{\mathbf{e}}_\phi - \sin \theta \hat{\mathbf{e}}_\theta$$

$$\hat{\mathbf{e}}_y = \sin \phi \sin \theta \hat{\mathbf{e}}_R + \cos \phi \sin \theta \hat{\mathbf{e}}_\phi + \cos \theta \hat{\mathbf{e}}_\theta$$

$$\hat{\mathbf{e}}_z = \cos \phi \hat{\mathbf{e}}_R - \sin \phi \hat{\mathbf{e}}_\phi$$

$$\frac{\partial \hat{\mathbf{e}}_R}{\partial \phi} = \hat{\mathbf{e}}_\phi, \quad \frac{\partial \hat{\mathbf{e}}_R}{\partial \theta} = \sin \phi \hat{\mathbf{e}}_\theta, \quad \frac{\partial \hat{\mathbf{e}}_\phi}{\partial \phi} = -\hat{\mathbf{e}}_R, \quad \frac{\partial \hat{\mathbf{e}}_\phi}{\partial \theta} = \cos \phi \hat{\mathbf{e}}_\theta,$$

$$\frac{\partial \hat{\mathbf{e}}_\theta}{\partial \theta} = -\sin \phi \hat{\mathbf{e}}_R - \cos \phi \hat{\mathbf{e}}_\phi$$

All other derivatives of the base vectors are zero.

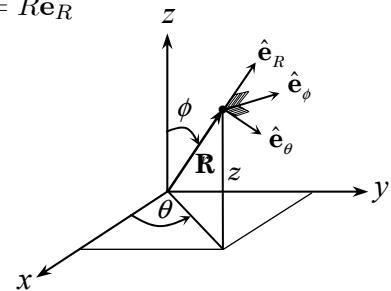
$$\nabla = \hat{\mathbf{e}}_R \frac{\partial}{\partial R} + \frac{1}{R} \hat{\mathbf{e}}_\phi \frac{\partial}{\partial \phi} + \frac{1}{R \sin \phi} \hat{\mathbf{e}}_\theta \frac{\partial}{\partial \theta},$$

$$\nabla^2 = \frac{1}{R^2} \frac{\partial}{\partial R} \left(R^2 \frac{\partial}{\partial R} \right) + \frac{1}{R^2 \sin \phi} \frac{\partial}{\partial \phi} \left(\sin \phi \frac{\partial}{\partial \phi} \right) + \frac{1}{R^2 \sin^2 \phi} \frac{\partial^2}{\partial \theta^2}$$

$$\nabla \cdot \mathbf{A} = 2 \frac{A_R}{R} + \frac{\partial A_R}{\partial R} + \frac{1}{R \sin \phi} \frac{\partial(A_\phi \sin \phi)}{\partial \phi} + \frac{1}{R \sin \phi} \frac{\partial A_\theta}{\partial \theta}$$

$$\nabla \times \mathbf{A} = \frac{1}{R \sin \phi} \left[\frac{\partial(\sin \phi A_\theta)}{\partial \phi} - \frac{\partial A_\phi}{\partial \theta} \right] \hat{\mathbf{e}}_R + \left[\frac{1}{R \sin \phi} \frac{\partial A_R}{\partial \theta} - \frac{1}{R} \frac{\partial(R A_\theta)}{\partial R} \right] \hat{\mathbf{e}}_\phi$$

$$+ \frac{1}{R} \left[\frac{\partial(R A_\phi)}{\partial R} - \frac{\partial A_R}{\partial \phi} \right] \hat{\mathbf{e}}_\theta$$



1.4 Governing Equations of a Continuum

1.4.1 Introduction

In the present study we are interested in the combined convective and conductive transport of thermal energy in a material region, Ω . In the general case, the region Ω is made up of subregions containing a moving fluid in Ω_f and a solid body in Ω_s , where $\Omega = \Omega_f \cup \Omega_s$. The motion or equilibrium of a continuous medium is governed by global conservation principles.

There are two alternative descriptions used to express the conservation laws in analytical form. In the first, one considers the motion of all matter passing through a *fixed spatial location*. Here one is interested in various properties (e.g., velocity, pressure, temperature, density, and so on) of the matter that instantly occupies the fixed spatial location. This description is called the *Eulerian description* or *spatial description*. In the second, one focuses attention on a *set of fixed material particles*, irrespective of their spatial locations. The relative displacements of these particles and the stress caused by external forces and temperature are of interest in this case. This description is known as the *Lagrangian description* or *material description*. The Eulerian description is most commonly used to study fluid flows and convective heat transfer, while the Lagrangian description is generally used to study solid body heat conduction and the stress and deformation of solid bodies. Here we present the governing equations of a continuous medium based primarily on the Eulerian description. For a derivation of the basic conservation equations, the reader may consult the books on continuum mechanics (e.g., Reddy [1], Bird, Stewart, Lightfoot [2], Malvern [3], and Reddy and Rasmussen [4]), heat transfer (e.g., see Bejan [5], and Özisik [6,7]), and fluid mechanics (e.g., Batchelor [8] and Schlichting [9]).

In the next sections, the general conservation laws will be stated and specialized to the primary case of interest, nonisothermal, incompressible flows. Later sections will address the equations for low-speed, variable density flows, flows in porous media and other types of transport. Note that in the following, standard vector and tensor notation and operations are used without detailed explanation (see Reddy [1] or Reddy and Rasmussen [4]). Both Gibbs and Einstein summation notations are used as appropriate and convenient.

1.4.2 Conservation of Mass; the Continuity Equation

The principle of conservation of mass can be stated as *the time rate of change of mass in a fixed volume is equal to the net rate of flow of mass across the surface*. The mathematical statement of the principle results in the following equation, known as the *continuity (of mass) equation*

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \quad (1.4.1)$$

where ρ is the density (kg/m^3) of the medium, \mathbf{v} the velocity vector (m/s), and ∇ is the nabla or del operator. The continuity equation in (1.4.1) is in conservation (or divergence) form since it can be derived directly from an integral statement of mass conservation. By introducing the *material derivative* or *Eulerian derivative* operator D/Dt

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{v} \cdot \nabla, \quad (1.4.2)$$

the continuity equation (1.4.1) can be expressed in the alternate, non-conservation (or advective) form

$$\frac{\partial \rho}{\partial t} + \mathbf{v} \cdot \nabla \rho + \rho \nabla \cdot \mathbf{v} = \frac{D\rho}{Dt} + \rho \nabla \cdot \mathbf{v} = 0 \quad (1.4.3)$$

For steady-state conditions, the continuity equation becomes

$$\nabla \cdot (\rho \mathbf{v}) = 0 \quad (1.4.4)$$

When the density changes following a fluid particle are negligible, the continuum is termed *incompressible* and we have $D\rho/Dt = 0$. The continuity equation (1.4.3) then becomes

$$\nabla \cdot \mathbf{v} = 0 \quad (1.4.5)$$

which is often referred to as the incompressibility condition or incompressibility constraint.

1.4.3 Conservation of Momenta

The principle of conservation of linear momentum (or Newton's Second Law of motion) states that *the time rate of change of linear momentum of a given set of particles is equal to the vector sum of all the external forces acting on the particles of the set, provided Newton's Third Law of action and reaction governs the internal forces*. Newton's Second Law can be written as

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v}) = \nabla \cdot \sigma + \rho \mathbf{f} \quad (1.4.6)$$

where \otimes is the tensor (or dyadic) product of two vectors, σ is the Cauchy stress tensor (N/m^2) and \mathbf{f} is the body force vector, measured per unit mass and normally taken to be the gravity vector. Equation (1.4.6) describes the motion of a continuous medium, and in fluid mechanics they are also known as the *Navier equations*.

The form of the momentum equation shown in (1.4.6) is the conservation (divergence) form that is most often utilized for compressible flows. This equation may be simplified to a form more commonly used with incompressible flows. Expanding the first two derivatives and collecting terms

$$\rho \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \nabla \cdot \mathbf{v} \right) + \mathbf{v} \left(\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{v} \right) = \nabla \cdot \sigma + \rho \mathbf{f} \quad (1.4.7)$$

The second term in parentheses is the continuity equation (1.4.1) and neglecting this term allows (1.4.7) to reduce to the non-conservation (advective) form

$$\rho \frac{D \mathbf{v}}{Dt} = \nabla \cdot \sigma + \rho \mathbf{f} \quad (1.4.8)$$

where the material derivative (1.4.2) has been employed.

The principle of conservation of angular momentum can be stated as *the time rate of change of the total moment of momentum of a given set of particles is equal to the vector sum of the moments of the external forces acting on the system*. In the absence of distributed couples, the principle leads to the symmetry of the stress tensor:

$$\sigma = (\sigma)^T \quad (1.4.9)$$

where the superscript T denotes the transpose of the enclosed quantity.

1.4.4 Conservation of Energy

The law of conservation of energy (or the First Law of Thermodynamics) states that *the time rate of change of the total energy is equal to the sum of the rate of work done by applied forces and the change of heat content per unit time*. In the general case, the First Law of Thermodynamics can be expressed in conservation form as

$$\frac{\partial \rho e^t}{\partial t} + \nabla \cdot \rho \mathbf{v} e^t = -\nabla \cdot \mathbf{q} + \nabla \cdot (\sigma \cdot \mathbf{v}) + Q + \rho \mathbf{f} \cdot \mathbf{v} \quad (1.4.10)$$

where $e^t = e + 1/2\mathbf{v} \cdot \mathbf{v}$ is the total energy (J/m^3), e is the internal energy, \mathbf{q} is the heat flux vector (W/m^2) and Q is the internal heat generation (W/m^3). The total energy equation (1.4.10) is useful for high speed compressible flows where the kinetic energy is significant. For incompressible flows, an internal energy equation is more appropriate and can be derived from (1.4.10) with use of the momentum equation (1.4.6). Taking the dot product of the velocity vector with the momentum equation produces an equation for the kinetic energy; this equation is subtracted from the total energy equation (1.4.10) to produce the conservation (divergence) form of the internal energy equation

$$\frac{\partial \rho e}{\partial t} + \nabla \cdot \rho \mathbf{v} e = -\nabla \cdot \mathbf{q} + Q + \Phi \quad (1.4.11)$$

where Φ is a dissipation function that is defined by

$$\Phi = \sigma : \nabla \mathbf{v} \quad (1.4.12)$$

In Eq.(1.4.12) $\nabla \mathbf{v}$ is the velocity gradient tensor which will be defined more completely in the following sections.

The thermal energy equation in (1.4.11) can be simplified further by expanding the derivatives on the left-hand side of the equation and using the continuity equation. The resulting equation is the non-conservative (advective) form of the energy equation

$$\rho \frac{De}{Dt} = -\nabla \cdot \mathbf{q} + Q + \Phi \quad (1.4.13)$$

which is the standard form used for incompressible flows. Constitutive relations for e and \mathbf{q} will be defined in the next sections and allow (1.4.13) to be expressed in terms of the temperature T .

1.4.5 Equation of State

In addition to the conservation laws described above, two thermodynamic relations or equations of state are required. A caloric equation of state relates the internal energy of the material to two (intensive) thermodynamic variables. Of the many possible forms for this equation (see, for example, Bejan [10]) one of the most common for fluid dynamics is the functional relation

$$e = e(T, \nu) \quad (1.4.14)$$

where $\nu = 1/\rho$ is the specific volume. The change in internal energy is then

$$de = \frac{\partial e}{\partial T} \Big|_{\nu} dT + \frac{\partial e}{\partial \nu} \Big|_T d\nu \quad (1.4.15a)$$

or

$$de = C_v dT + \left(T \frac{\partial P}{\partial T} \Big|_{\nu} - P \right) d\nu \quad (1.4.15b)$$

where standard Maxwell relations from thermodynamics have been used. When converted to a material derivative the change in internal energy becomes

$$\rho \frac{De}{Dt} = \rho C_v \frac{DT}{Dt} + \left[T \frac{\partial P}{\partial T} \Big|_{\rho} - P \right] \nabla \cdot \mathbf{v} \quad (1.4.16)$$

For an incompressible fluid or flow, the second term on the right-hand side of (1.4.16) is zero by the continuity equation and

$$\rho \frac{De}{Dt} = \rho C_v \frac{DT}{Dt} = \rho C_p \frac{DT}{Dt} = \rho C \frac{DT}{Dt} \quad (1.4.17)$$

where C is the specific heat [$\text{J}/(\text{kg}\cdot^{\circ}\text{C})$]; the specific heat at constant volume C_v is the same as the specific heat at constant pressure C_p for constant density processes.

The equation of state provides a relation among three (intensive) thermodynamic variables. The perfect gas law

$$P = \rho RT \quad (1.4.18)$$

is one common equation of state with R being the universal gas constant. For most of the applications considered here, the incompressible flow assumption leads to the equation of state

$$\rho = \rho_0 \quad (1.4.19)$$

which implies that the pressure is no longer a thermodynamic variable; the temperature remains as a variable through the caloric equation of state. Under the incompressible assumption the pressure is defined through the mechanical constitutive relation (see next section) and is treated as a mechanical variable. For nonisothermal, incompressible flows, the relation in (1.4.19) decouples the conservation of energy equation from the momentum and continuity equations. However, for flows involving buoyancy forces the equation of state is altered to an extended form of the Boussinesq approximation (see Gray and Giorgini [11] and Gartling and Hickox [12]). The extended Boussinesq approximation allows

the fluid properties to be expressed as functions of the thermodynamic state (e.g., temperature) and the density ρ to vary with temperature T according to the relation

$$\rho = \rho_0[1 - \beta(T - T_0)] \quad (1.4.20)$$

where β is the coefficient of thermal expansion ($1/\text{ }^{\circ}\text{C}$) and the subscript zero indicates a reference condition. The variation of density as given in Eq. (1.4.20) is permitted only in the description of the body force; the density in all other situations is assumed to be that of the reference state, ρ_0 .

1.4.6 Constitutive Equations

The fluids of interest, for the moment, are assumed to be Newtonian (i.e., the constitutive relations are linear). Non-Newtonian fluids will be considered in a later chapter. Further, the flow is laminar. Then, for viscous incompressible fluids the total stress σ can be decomposed into hydrostatic and viscous parts:

$$\sigma = -P\mathbf{I} + \tau \quad (1.4.21)$$

where P is the hydrostatic (mechanical) pressure, \mathbf{I} is the unit tensor and τ is the viscous stress tensor. For Newtonian fluids, the viscous stress tensor is related to the strain rate tensor \mathbf{D} by

$$\tau = \mathbf{C} : \mathbf{D} \quad (1.4.22)$$

where \mathbf{C} is the fourth-order tensor of fluid properties and \mathbf{D} is the strain rate tensor (or rate of deformation tensor)

$$\mathbf{D} = \frac{1}{2}[(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T] \quad (1.4.23)$$

and $\nabla \mathbf{v}$ is the velocity gradient tensor.

For an isotropic fluid (i.e., whose material properties are independent of direction), the fourth-order tensor \mathbf{C} can be expressed in terms of two constants λ and μ , called Lamé constants, and Eq. (1.4.22) takes the form

$$\tau = \lambda(\text{tr } \mathbf{D})\mathbf{I} + 2\mu\mathbf{D} \quad (1.4.24a)$$

where $(\text{tr } \mathbf{D})$ denotes the *trace* (or sum of the diagonal elements) of the matrix \mathbf{D} . For an incompressible fluid, we have $\text{tr } \mathbf{D} = 0$ and Eq. (1.4.24a) becomes

$$\tau = 2\mu\mathbf{D} \quad (1.4.24b)$$

and the total stress tensor is then

$$\sigma = -P\mathbf{I} + 2\mu\mathbf{D} \quad (1.4.25)$$

With the stress tensor defined, the dissipation function in the energy equation may be rewritten as

$$\Phi = (-P\mathbf{I} + \tau) : \nabla \mathbf{v} \quad (1.4.26)$$

Note that the velocity gradient tensor can be expressed as the sum of the symmetric strain rate tensor and the skew-symmetric spin tensor

$$\nabla \mathbf{v} = \frac{1}{2} [(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T] + \frac{1}{2} [(\nabla \mathbf{v}) - (\nabla \mathbf{v})^T] \equiv \mathbf{D} - \boldsymbol{\Omega} \quad (1.4.27)$$

When substituted into (1.4.26) and contracted with the symmetric stress tensor, the dissipation function becomes

$$\Phi = -P \nabla \cdot \mathbf{v} + \tau : \mathbf{D} \quad (1.4.28)$$

For an incompressible fluid, the reversible pressure work term is zero, and using the definition of τ from (1.4.24b) the most common form of the dissipation function becomes

$$\Phi = 2\mu \mathbf{D} : \mathbf{D} \quad (1.4.29)$$

The Fourier heat conduction law states that

$$\mathbf{q} = -\mathbf{k} \cdot \nabla T \quad (1.4.30a)$$

where \mathbf{k} denotes the conductivity tensor of order two. For an isotropic medium, \mathbf{k} is of the form

$$\mathbf{k} = k \mathbf{I} \quad (1.4.30b)$$

where k denotes the thermal conductivity [$\text{W}/(\text{m} \cdot ^\circ\text{C})$] of the medium, and \mathbf{I} is again the unit tensor.

The material coefficients, μ, C, β , and \mathbf{k} are generally functions of the fluid temperature; property variations as a function of a second thermodynamic variable (e.g., pressure) are possible but usually are unimportant for an incompressible fluid. Solid materials may also have specific heats and conductivities that vary with temperature. In addition, the solid conductivity may vary with spatial position and, in general, remains a symmetric second-order tensor (i.e., $\mathbf{k}^T = \mathbf{k}$) for anisotropic materials. The volumetric heat source for the fluid and/or solid may be a function of temperature, time, and spatial location. In developing the finite element models in the coming chapters, the dependence of the material properties on the spatial location is assumed. The dependence of the viscosity and conductivity on the strain rates and/or temperature are also discussed in later sections and chapters.

1.4.7 Divergence and Advection Forms

In the previous descriptions of the conservation principles, both the divergence (conservation) and advection (non-conservation) forms of the transport operators were considered. The differences between the two forms were related to products (moments) of the continuity equation. In the exact continuum description these differences are unimportant. However, in a discretized computational form, the divergence and advective operators are not the same, as the continuity equation may only be satisfied approximately. Here the general transport operator is written for the scalar or vector function f as

$$(1-\alpha) \left(\rho \frac{\partial f}{\partial t} + \rho \mathbf{v} \cdot \nabla f \right) + \alpha \left(\frac{\partial \rho f}{\partial t} + \nabla \cdot \rho \mathbf{v} f \right) = \rho \frac{\partial f}{\partial t} + \rho \mathbf{v} \cdot \nabla f + \alpha f \left(\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{v} \right) \quad (1.4.31)$$

which is the weighted combination of the divergence and advective forms. If $\alpha = 0$, an advective derivative is obtained while $\alpha = 1$ produces the divergence form. An $\alpha = 1/2$ value corresponds to the “absolute” conserving form, first proposed by Piacsek and Williams [13]. The alpha form of the transport operator (1.4.31) has been used with the conservation equations to study and demonstrate conservation of various basic and derived quantities in both the continuum and discrete finite element settings. The conservation of certain quantities, such as temperature squared, can be shown to promote stability of some forms of finite element models. Using the definition in (1.4.31) allows any of the various forms to be easily constructed for both the momentum and energy equations. Further comments on the advection versus divergence forms will be delayed until the discretized forms of the flow and transport equations are developed.

1.5 Governing Equations in Terms of Primitive Variables

1.5.1 Vector Form

The advective form of the conservation equations (1.4.3), (1.4.8), and (1.4.13) can be expressed in terms of the primitive variables (\mathbf{v}, P, T) by means of equations (1.4.17), (1.4.20), (1.4.21), (1.4.25) and (1.4.30). The results are summarized below for isotropic, Newtonian, viscous, incompressible fluids in the presence of buoyancy forces:

$$\nabla \cdot \mathbf{v} = 0 \quad (1.5.1)$$

$$\begin{aligned} \rho_0 \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) &= -\nabla P + \nabla \cdot \left\{ \mu \left[(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T \right] \right\} \\ &\quad + \rho_0 \mathbf{g} - \rho_0 \mathbf{g} \beta (T - T_0) \end{aligned} \quad (1.5.2)$$

$$\rho_0 C \left(\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T \right) = \nabla \cdot (k \nabla T) + Q + \Phi \quad (1.5.3)$$

where \mathbf{v} represents the velocity vector, ρ_0 the density, \mathbf{g} the gravity force vector per unit mass, T the temperature, C the specific heat of the fluid, and Q the rate of heat generation. The viscous dissipation is related to the velocity through (1.4.29) and (1.4.23).

The above equations are valid for the fluid region Ω_f . In the solid region Ω_s , the fluid velocity is zero, $\mathbf{v} = \mathbf{0}$, and the only relevant equation is (1.5.3). The energy equation (1.5.3) for the solid region is given by

$$\rho_s C_s \frac{\partial T}{\partial t} = \nabla \cdot (k_s \nabla T) + Q_s \quad (1.5.4)$$

In writing Eq. (1.5.4) it is assumed that the solid material is stationary with respect to the Eulerian coordinate frame such that the advective transport of energy [i.e., the velocity-dependent part of Eq. (1.5.3)] need not be considered. Equation (1.5.4) is also valid for a solid material moving in a Lagrangian reference frame.

1.5.2 Cartesian Component Form

The vector form of the equations in (1.5.1)–(1.5.3) allows us to express them in any coordinate system. In the Cartesian coordinate system (x_1, x_2, x_3) , the kinematic and constitutive relations become

$$D_{ij} = \frac{1}{2} \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \quad (1.5.5)$$

$$\sigma_{ij} = -P\delta_{ij} + \tau_{ij}; \quad \tau_{ij} = 2\mu D_{ij} \quad (1.5.6)$$

The conservation equations can be expressed as

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (1.5.7)$$

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left[-P\delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 g_i - \rho_0 g_i \beta(T - T_0) \quad (1.5.8)$$

$$\rho_0 C \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + Q + 2\mu D_{ij} D_{ij} \quad (1.5.9)$$

for the fluid region Ω_f and

$$\rho_s C_s \frac{\partial T}{\partial t} = \frac{\partial}{\partial x_i} \left(k_s \frac{\partial T}{\partial x_i} \right) + Q_s \quad (1.5.10)$$

for the solid region Ω_s . Equations (1.5.5)–(1.5.10) are written for a Cartesian geometry in an Eulerian reference frame, with the indices $i, j = 1, 2, 3$ (or $i, j = 1, 2$ for two-dimensional problems); the summation convention of Section 1.3.2 is used (also see Reddy [1], pp. 18–21).

1.5.3 Cylindrical Component Form

Equations (1.5.1)–(1.5.3) can also be expressed in a cylindrical coordinate system, (r, θ, z) , by writing all vectors and tensors, including the del operator, in terms of components in a cylindrical coordinate system (see Table 1.3.1). For example, the del operator and the material time derivative operators in the cylindrical coordinate system are given by (see Reddy [4], pp. 39–41)

$$\nabla = \hat{\mathbf{e}}_r \frac{\partial}{\partial r} + \hat{\mathbf{e}}_\theta \frac{1}{r} \frac{\partial}{\partial \theta} + \hat{\mathbf{e}}_z \frac{\partial}{\partial z} \quad (1.5.11)$$

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + v_r \frac{\partial}{\partial r} + \frac{v_\theta}{r} \frac{\partial}{\partial \theta} + v_z \frac{\partial}{\partial z} \quad (1.5.12)$$

where $(\hat{\mathbf{e}}_r, \hat{\mathbf{e}}_\theta, \hat{\mathbf{e}}_z)$ are the unit basis vectors and (v_r, v_θ, v_z) are the velocity components in the r , θ , and z directions, respectively. Note that the basis vector $\hat{\mathbf{e}}_z$ is constant while the vectors $\hat{\mathbf{e}}_r$ and $\hat{\mathbf{e}}_\theta$ depend on the angular coordinate θ . Thus

the derivatives of the basis vectors $\hat{\mathbf{e}}_r$ and $\hat{\mathbf{e}}_\theta$ with respect to the coordinates r and z are zero, and the derivatives with respect to θ are given by

$$\frac{\partial \hat{\mathbf{e}}_r}{\partial \theta} = \hat{\mathbf{e}}_\theta; \quad \frac{\partial \hat{\mathbf{e}}_\theta}{\partial \theta} = -\hat{\mathbf{e}}_r \quad (1.5.13)$$

The kinematic and constitutive relations are

$$\begin{aligned} D_{rr} &= \frac{\partial v_r}{\partial r}; & D_{\theta\theta} &= \frac{1}{r} \frac{\partial v_\theta}{\partial \theta} + \frac{v_r}{r} \\ D_{zz} &= \frac{\partial v_z}{\partial z}; & 2D_{r\theta} &= \frac{\partial v_\theta}{\partial r} - \frac{v_\theta}{r} + \frac{1}{r} \frac{\partial v_r}{\partial \theta} \\ 2D_{\theta z} &= \frac{1}{r} \frac{\partial v_z}{\partial \theta} + \frac{\partial v_\theta}{\partial z}; & 2D_{zr} &= \frac{\partial v_r}{\partial z} + \frac{\partial v_z}{\partial r} \end{aligned} \quad (1.5.14)$$

$$\begin{aligned} \sigma_{rr} &= -P + 2\mu D_{rr}; & \sigma_{\theta\theta} &= -P + 2\mu D_{\theta\theta}; & \sigma_{zz} &= -P + 2\mu D_{zz} \\ \sigma_{r\theta} &= 2\mu D_{r\theta}; & \sigma_{\theta z} &= 2\mu D_{\theta z}; & \sigma_{zr} &= 2\mu D_{zr} \end{aligned} \quad (1.5.15)$$

The governing equations are summarized below:

$$\frac{1}{r} \frac{\partial}{\partial r} (rv_r) + \frac{1}{r} \frac{\partial v_\theta}{\partial \theta} + \frac{\partial v_z}{\partial z} = 0 \quad (1.5.16)$$

$$\begin{aligned} \rho_0 \left(\frac{Dv_r}{Dt} - \frac{v_\theta^2}{r} \right) &= \rho g_r + \frac{1}{r} \left[\frac{\partial(r\sigma_{rr})}{\partial r} + \frac{\partial\sigma_{r\theta}}{\partial\theta} + \frac{\partial(r\sigma_{zr})}{\partial z} \right] - \frac{\sigma_{\theta\theta}}{r} \\ \rho_0 \left(\frac{Dv_\theta}{Dt} + \frac{v_r v_\theta}{r} \right) &= \rho g_\theta + \frac{1}{r} \left[\frac{\partial(r\sigma_{r\theta})}{\partial r} + \frac{\partial\sigma_{\theta\theta}}{\partial\theta} + \frac{\partial(r\sigma_{\theta z})}{\partial z} \right] + \frac{\sigma_{r\theta}}{r} \\ \rho_0 \left(\frac{Dv_z}{Dt} \right) &= \rho g_z + \frac{1}{r} \left[\frac{\partial(r\sigma_{zr})}{\partial r} + \frac{\partial\sigma_{z\theta}}{\partial\theta} + \frac{\partial(r\sigma_{zz})}{\partial z} \right] \end{aligned} \quad (1.5.17)$$

$$\begin{aligned} \rho_0 C \left(\frac{\partial T}{\partial t} + v_r \frac{\partial T}{\partial r} + \frac{v_\theta}{r} \frac{\partial T}{\partial \theta} + v_z \frac{\partial T}{\partial z} \right) &= \frac{1}{r} \frac{\partial}{\partial r} \left(r k_{rr} \frac{\partial T}{\partial r} \right) + \frac{1}{r^2} \frac{\partial}{\partial \theta} \left(k_{\theta\theta} \frac{\partial T}{\partial \theta} \right) \\ &\quad + \frac{\partial}{\partial z} \left(k_{zz} \frac{\partial T}{\partial z} \right) + \Phi + Q \end{aligned} \quad (1.5.18)$$

where the stress components are known in terms of the velocity components *via* equations (1.5.14) and (1.5.15), and the viscous dissipation Φ is given by

$$\begin{aligned} \Phi &= 2\mu \left[\left(\frac{\partial v_r}{\partial r} \right)^2 + \left(\frac{1}{r} \frac{\partial v_\theta}{\partial \theta} + \frac{v_r}{r} \right)^2 + \left(\frac{\partial v_z}{\partial z} \right)^2 \right] + \mu \left(\frac{\partial v_\theta}{\partial r} - \frac{v_\theta}{r} + \frac{1}{r} \frac{\partial v_r}{\partial \theta} \right)^2 \\ &\quad + \mu \left(\frac{1}{r} \frac{\partial v_z}{\partial \theta} + \frac{\partial v_\theta}{\partial z} \right)^2 + \mu \left(\frac{\partial v_r}{\partial z} + \frac{\partial v_z}{\partial r} \right)^2 \end{aligned} \quad (1.5.19)$$

Note that the time derivative of a vector (or tensor) in a rotating reference frame is given by (see Reddy and Rasmussen [4], pp. 69–74)

$$\left[\frac{D(\cdot)}{Dt} \right]_{\text{nonrot}} = \left[\frac{D(\cdot)}{Dt} \right]_{\text{rot}} + \omega \times (\cdot) \quad (1.5.20)$$

where $\omega = \frac{d\theta}{dt} \hat{\mathbf{e}}_z = \frac{v_\theta}{r} \hat{\mathbf{e}}_z$ is the angular velocity of the rotating frame of reference.

Therefore, we have

$$\begin{aligned} \left(\frac{D\mathbf{v}}{Dt} \right)_{\text{nonrot}} &= \left(\frac{D\mathbf{v}}{Dt} \right)_{\text{rot}} + \left(\frac{v_\theta}{r} \hat{\mathbf{e}}_z \right) \times \mathbf{v} \\ &= \hat{\mathbf{e}}_r \left(\frac{Dv_r}{Dt} - v_\theta \frac{v_\theta}{r} \right) + \hat{\mathbf{e}}_\theta \left(\frac{Dv_\theta}{Dt} + v_r \frac{v_\theta}{r} \right) - \hat{\mathbf{e}}_z \frac{Dv_z}{Dt} \end{aligned} \quad (1.5.21)$$

Equation (1.5.21) could be used in conjunction with Eqs. (1.5.17) to provide a description of fluid motion in a rotating cylindrical coordinate system.

1.5.4 Closure

Equations (1.5.1)–(1.5.4) describe the convective and conductive heat transfer problems of primary interest. Although the velocities, pressure, and temperature were selected as the dependent variables, other choices for the description of the fluid motion are possible. Indeed, equations employing the stream function and vorticity, or the stream function alone, can be derived and used effectively with the energy equation to analyze convection problems. The choice of velocity, pressure, and temperature variables, the so-called “primitive variables,” is made here for a number of reasons. The velocity and pressure variables are a natural description for fluid mechanics and permit boundary conditions to be described in a straightforward and intuitive manner. The description of free surface flows, problems in multiply connected domains, and phase change problems is direct and uncomplicated with the primitive variables. Also, the use in two or three dimensions and various coordinate systems is relatively straightforward, as is the inclusion of non-Newtonian and/or porous (Darcy) flow effects. Although none of the reasons cited above is by itself a compelling argument for exclusive use of the primitive variable approach, it is the most commonly used approach. All of the subsequent developments in this book will be carried out using the primitive variable description.

1.6 Porous Flow Equations

Suppose that the fluid domain, Ω_f , is composed of two subregions denoted by Ω_{vf} and Ω_{pf} such that $\Omega_f = \Omega_{vf} \cup \Omega_{pf}$. In the subregion Ω_{vf} , the viscous flow equations (1.5.7)–(1.5.9) are assumed to hold. The subregion Ω_{pf} is assumed to contain a rigid porous medium that is saturated with a viscous, incompressible fluid. The saturating fluid in Ω_{pf} is the same fluid as in Ω_{vf} if the two regions share a common permeable interface; otherwise the fluids in the two regions may be different. If the porous medium is assumed to be homogeneous and isotropic and the fluid and solid are in thermal equilibrium, then the equations describing the fluid motion and energy transport in the region Ω_{pf} can be written for a Cartesian rectangular coordinate system as

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (1.6.1)$$

$$\begin{aligned} \rho_0 \alpha_{ij} \frac{\partial v_j}{\partial t} + \left(\frac{\rho_0 \hat{c}}{\sqrt{\kappa}} \|\mathbf{v}\| + \frac{\mu_e}{\kappa} \right) v_i &= \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu_e \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] \\ &+ \rho_0 g_i - \rho_0 g_i \beta(T - T_0) \end{aligned} \quad (1.6.2)$$

$$(\rho C)_e \left(\frac{\partial T}{\partial t} \right) + (\rho_0 C) \left(v_j \frac{\partial T}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left(k_e \frac{\partial T}{\partial x_i} \right) + Q \quad (1.6.3)$$

Analogous equations are available for other coordinate systems. In Eqs. (1.6.1)–(1.6.3), κ is the permeability, \hat{c} is the inertia coefficient, $\|\mathbf{v}\|$ is the magnitude of the velocity vector, and α_{ij} are the components of the acceleration tensor. The porosity, φ , of the medium is defined as the fraction of the total volume of material occupied by interconnected pores (voids). The velocity components in (1.6.1)–(1.6.3) represent the volume averaged Darcy velocity (also termed the seepage or filtration velocity) which is related to the interstitial pore level velocity, v_i^p , by the inverse of the porosity of the medium, i.e., $v_i = v_i^p/\varphi$. The subscript e indicates an effective property; all other symbols are as defined earlier. The effective capacitance and conductivity are generally functions of the fluid and solid matrix properties and the porosity. For example, the capacitance is usually a porosity weighted function such that $(\rho C)_e = \varphi(\rho_0 C) + (1 - \varphi)(\rho_s C_s)$. The effective conductivity can also be a function of the fluid velocity when thermal dispersion effects are important. The acceleration tensor is dependent on the geometry of the porous matrix, though in most cases it is proportional to $1/\varphi$. Flow transients in a porous material usually decay very rapidly and the time-dependent term in (1.6.2) is often neglected. The effective viscosity is often taken to be $\mu_e = \mu$, although other approximations are possible. Suitable empirical models are available for all these quantities, and they can be found in the books and articles by Kaviany [14], Nield and Joseph [15], Nield [16], and Nield and Bejan [17].

Equations (1.6.1)–(1.6.3) represent a generalization of the standard Darcy equations for nonisothermal flow in a saturated porous medium. This system is sometimes referred to as the *Forchheimer–Brinkman* or *Darcy–Forchheimer–Brinkman* model for porous flow. No attempt will be made here to delineate all the conditions under which such a model is appropriate. However, it should be noted that by selectively including or omitting certain terms, a number of other standard porous flow models can be derived. For example, if $\hat{c} = 0$, a Brinkman model is obtained, while a prescription of $\hat{c} = 0$ and $\mu_e = 0$ produces the standard Darcy's model. For further discussion of these models and their regions of applicability, the reader is referred to Nield [16], Nield and Bejan [17], Beavers and Joseph [18], Bear [19], and Gartling, Hickox, and Givler [20].

The general form of the porous flow equations is very similar to the equations for the viscous flow of a bulk fluid. By simply redefining certain terms and coefficients in equations (1.6.1)–(1.6.3), they can in fact be obtained from (1.5.7)–(1.5.10). As a result of this simple transformation process, it is quite straightforward to include a porous flow model in the general computational framework for nonisothermal viscous flow problems. The region Ω_{pf} may exist in conjunction with Ω_{vf} or in place of Ω_{vf} . The solid region, Ω_s , may or may not be included as the problem dictates. In the following sections specific reference will be made to the porous flow equations only when their treatment differs significantly from the equations for a viscous fluid.

1.7 Low-Speed, Compressible Flow Equations

Though the nonisothermal, viscous, incompressible equations of previous sections are a standard model for many heat transfer problems, they are limited in their

applicability. The Boussinesq approximation and equations of Section 1.4 are restricted to flows with “small” variations in density and pressure, relative to a static state. These limitations also imply a restriction to small temperature differences. For gases, the Boussinesq model is generally limited to temperature differences of less than $30^{\circ}K$ and for liquids the limit is $3^{\circ}K$. The Boussinesq model approximates density as a constant in all terms of the conservation equations except for the body force (buoyancy) term in the momentum equation. The allowance of density variations with temperature in the body force provides the usual association of the Boussinesq equations with free or natural convection heat transfer. Note that the restriction of small density variations is still required for incompressible forced convection problems.

To allow for large temperature differences in convective heat transfer, the restrictions on small density variations must be eliminated. This implies the use of some form of the compressible flow equations that are appropriate for relatively low-speed flows. The fully compressible flow equations are not well suited to this flow regime because of the disparity in time scales that is supported by the equations. The relatively slow mean flow, that is of primary interest, also contains faster acoustic waves that are of minimal importance to the heat transfer process. An appropriate compressible flow formulation that is useful for low-speed, heat transfer applications may be derived by eliminating or filtering the acoustic waves from the compressible equations. The acoustic modes are removed by decoupling the density and temperature variations in the momentum equation from the variations in the equation of state.

The formal method for deriving the “acoustically filtered” equations for low-speed, compressible flows relies on an asymptotic analysis for the limit of small Mach number. The early work in this area was by Rehm and Baum [21] for inviscid flows and Paolucci [22] for viscous flows; a recent in-depth analysis is due to Principe and Codina [23]. In this derivation the Mach number is defined by $Ma = V/a$ where a is the speed of sound in the fluid and V is the characteristic flow speed. Expanding each flow variable in a power series with the small parameter Ma^2 and substituting into the fully compressible flow equations (1.4.1), (1.4.6), and (1.4.10) plus appropriate state and constitutive relations produces a hierarchy of equations. Considering the sets of equations of equal order in Ma^2 leads to the basic outcome of the analysis, the acoustically filtered equations

$$\frac{D\rho}{Dt} + \rho \nabla \cdot \mathbf{v} = 0 \quad (1.7.1)$$

$$\rho \frac{D\mathbf{v}}{Dt} = -\nabla P' + \nabla \cdot \tau_c + \rho \mathbf{f} \quad (1.7.2)$$

$$\rho C_v \frac{DT}{Dt} = -\nabla \cdot \mathbf{q} + \beta T \frac{dP_0}{dt} + Q \quad (1.7.3)$$

with

$$\rho = \rho(T, P_0) \quad (1.7.4)$$

$$\tau_c = 2\mu \mathbf{D} - \frac{2}{3}\mu(\nabla \cdot \mathbf{v})\mathbf{I} \quad (1.7.5)$$

where τ_c is the compressible version of the viscous stress tensor and C_v is the specific heat at constant volume.

In Eqs. (1.7.2)–(1.7.4), the pressure is composed of two parts, a spatially uniform, time-dependent background (thermodynamic) pressure $P_0(t)$ and an asymptotically smaller dynamical (mechanical) pressure, $P'(\mathbf{x}, t)$. The acoustic waves are suppressed because dynamic pressure fluctuations do not enter the equation of state. Note that the background pressure is determined from the initial condition for an open flow system and from an energy balance for a closed fluid system. The acoustically filtered system admits large variations in density (with minimal pressure dependence) and temperature, and are therefore useful for a wide range of heat transfer applications. Further details on the theoretical aspects of the non-Boussinesq equations can be found in [22] and [23].

The acoustically filtered equations in (1.7.1)–(1.7.3) are written in advective or non-conservation form. Another method for deriving a system that is uniformly valid in the low Mach number limit was provided by Hauke and Hughes [24]. In this case, the fully compressible equations in conservation form are written in terms of the pressure, primitive variables (rather than density as a variable), which are valid at all Mach numbers. In the low-speed limit, if the isothermal compressibility coefficient is set to zero, the remaining equations are essentially equivalent to the above acoustically filtered equations but in conservation form.

When considering a computational scheme for the acoustically filtered (or non-Boussinesq) equations, it is convenient to rewrite (1.7.1)–(1.7.3) in the mostly advective form

$$\nabla \cdot \rho \mathbf{v} = -\frac{\partial \rho}{\partial t} \quad (1.7.6)$$

$$\rho \frac{\partial \mathbf{v}}{\partial t} + \rho \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla P' + \nabla \cdot \left\{ \mu \left[(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T \right] \right\} - \frac{2}{3} \nabla \cdot (\mu \nabla \cdot \mathbf{v}) \mathbf{I} + \rho \mathbf{f} \quad (1.7.7)$$

$$\rho C_v \frac{\partial T}{\partial t} + \rho C_v \mathbf{v} \cdot \nabla T = \nabla \cdot k \nabla T + Q + \beta T \frac{dP_0}{dt} \quad (1.7.8)$$

and the equation of state

$$\rho = \rho(T, P_0) \quad (1.7.9)$$

Comparing these equations with the standard incompressible equations in (1.5.1)–(1.5.3) shows a great similarity in form. If the density is treated as a variable parameter [evaluated from (1.7.9)] then (1.7.6)–(1.7.8) are the same as the incompressible equations with additional source terms in the continuity and energy equations. Computational methods developed for the incompressible equations should then be applicable to the acoustically filtered equations and allow an expanded range of heat transfer problems to be considered. A finite element computational method for the acoustically filtered equations will be described and demonstrated in a later chapter.

1.8 Auxiliary Transport Equations

The various equation formulations governing nonisothermal, viscous flow problems outlined in the previous sections are very general. However, there are still a significant number of flow problems that cannot be described by the stated equations. For flows in which transport processes other than those of a thermal

nature are important, additional equations are required. The added equations are of the advection-diffusion type and are generic in the sense that they are not specifically associated with a particular physical process. Possible uses for these additional equations include the description of mass transport in a multicomponent system, the simulation of certain types of chemical reactions, and the prediction of volume fraction or particle orientation for flows containing suspended particles or fibers. Here we give the additional equations describing such flows.

Consider the material region Ω . The transport of a scalar quantity ϕ_i within this region is governed, for fixed i (i.e., no sum on i) by

$$C_i \left(\frac{\partial \phi_i}{\partial t} + v_j \frac{\partial \phi_i}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(D_i \frac{\partial \phi_i}{\partial x_j} \right) + Q_i \quad (1.8.1)$$

In Eq. (1.8.1), C_i is a “capacitance” coefficient [analogous to ρC in Eq. (1.5.9)], D_i is a diffusion coefficient, and Q_i is a volumetric source term; ϕ_i is not to be confused with the porosity φ . The subscript i runs between 1 and n , and it indicates that up to n transport equations could be added to the general flow problem. Note that Eq.(1.8.1) is valid in the solid regions of Ω if v_j is set to zero; the transport equation can also be used in a porous region if C_i and D_i are interpreted as “effective” properties of the fluid/matrix system.

In order to fully couple the above transport equation to the general nonisothermal flow problem, certain extensions must be made to the previously stated functional form for the fluid density variation. The possibility of buoyancy forces due to variations in the auxiliary variables ϕ_i requires that the equation of state (1.4.20) be modified to

$$\rho = \rho_0 [1 - \beta(T - T_0) - \gamma_1(\phi_1 - \phi_{1_0}) \dots - \gamma_n(\phi_n - \phi_{n_0})] \quad (1.8.2)$$

where γ_i is an expansion coefficient and the subscript zero refers to a reference condition. As before, this variation in density is only permitted in the body force term and represents an extension to the Boussinesq approximation for incompressible flows. The body force term in Eq. (1.5.8) must therefore be altered to include the last n terms in Eq. (1.8.2) whenever auxiliary transport equations of the appropriate type are included in the problem formulation. For the non-Boussinesq case, the thermal equation of state could be modified to include a functional dependence on ϕ_i .

1.9 Chemically Reacting Systems

The inclusion of chemical reactions in a fluid or thermal problem adds considerably to the difficulty of the numerical simulation. However, such effects are central to many important technology areas. The majority of these applications involve combustion or other gas phase processes that are often described by compressible or acoustically filtered equations. Chemical reactions in incompressible flows are often associated with gelation or polymerization reactions; some low-speed gas flows, such as chemical vapor deposition processing, are frequently modeled as incompressible. Porous flows and thermal diffusion problems coupled with chemical kinetics are

important for a variety of geomechanics applications, thermal ignition studies in energetic materials (e.g., pyrotechnics and explosives) and in the simulation of material decomposition or ablation. In this section we will outline the general form of the continuum equations that describe a chemically reactive system and make these equations specific for a few particular applications.

For a chemically reactive fluid, the boundary value problems described in Sections 1.4 and 1.5 are altered by the fact that the incompressible fluid must now be considered a mixture of various chemical species. Conservation principles must be invoked for the individual constituents of the fluid as well as for the fluid mixture as a whole. The overall mass, momentum, and energy balances for the mixture remain as given in Section 1.5 and are rewritten here only for reference.

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (1.9.1)$$

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = - \frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} \left[\mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho g_i \quad (1.9.2)$$

$$\rho C \frac{\partial T}{\partial t} + \rho C v_j \frac{\partial T}{\partial x_j} = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + Q_R \quad (1.9.3)$$

In equations (1.9.1)–(1.9.3) the thermophysical properties will normally be strong functions of the chemical composition of the fluid. Assume that the fluid is composed of I individual species and that these species may react according to J reactions. The primary effect of the reaction process on the flow is to alter the internal energy of the system; this change in energy provides the volume source term Q_R in equation (1.9.3).

A mass conservation equation for each of the I species is required and these have the form

$$\frac{\partial [N_i]}{\partial t} + v_j \frac{\partial [N_i]}{\partial x_j} = \frac{\partial}{\partial x_j} \left(D_{jk} \frac{\partial [N_i]}{\partial x_k} \right) + R_i \quad \text{for } i = 1, 2, \dots, I \quad (1.9.4)$$

where $[N_i]$ is the concentration variable (mole fraction) for species i , D_{jk} is the diffusion tensor and R_i is the chemical reaction rate. Note that there is no summation on i in Eq. (1.9.4). In writing Eq. (1.9.4), an incompressible flow was assumed and the diffusion process was assumed to follow Fick's law. More general species conservation equations can be derived that account for other types of diffusion processes (see Herschfelder, et al. [25] and Oran and Boris [26]), but these need not be considered here.

To describe the details of the chemical reaction, the stoichiometry, reaction kinetics, and material property behavior must be specified. For a material with I species and J reactions, the stoichiometry is provided by

$$\sum_{i=1}^I \nu'_{ij} \mathcal{M}_i \rightarrow \sum_{i=1}^I \nu''_{ij} \mathcal{M}_i \quad \text{for } j = 1, 2, \dots, J \quad (1.9.5)$$

where ν'_{ij} and ν''_{ij} are stoichiometric coefficients (usually integer values) and \mathcal{M}_i is the chemical symbol for the i th species. Generally, these expressions are given as

reversible reactions; however, they are treated here as irreversible, and the reversed reactions are specified as additional reaction steps. To accommodate expressions for global reactions, the stoichiometric coefficients are allowed to be non-integer.

For each step of the reaction, a reaction rate, r_j , is defined in the form:

$$r_j = k_j(T) \prod_{i=1}^I [N_i]^{\mu_{ij}} \text{ for } j = 1, 2, \dots, J \quad (1.9.6)$$

where μ_{ij} are the concentration exponents (usually $\mu_{ij} = \nu'_{ij}$ in kinetic theory, but here they are treated independently) and the symbol Π denotes the product operation. Typically, the expressions for the kinetic coefficients, $k_j(T)$, are given in an Arrhenius form

$$k_j(T) = T^{\beta_j} A_j \exp(-E_j/RT) \quad (1.9.7)$$

where β_j is the coefficient for a steric factor, A_j is the pre-exponential factor, E_j is the activation energy, and the universal gas constant is R . It is convenient to define $\nu_{ij} = (\nu''_{ij} - \nu'_{ij})$ and thus the rate of change of the species (neglecting diffusion) is given as

$$R_i = \sum_{j=1}^J \nu_{ij} r_j \text{ for } i = 1, 2, \dots, I \quad (1.9.8)$$

The chemical reaction process is coupled directly to the energy equation (1.9.3) by the volumetric source term

$$Q_r = \sum_{j=1}^J q_j r_j \quad (1.9.9)$$

where q_j represents the known endothermic or exothermic energy release for reaction step j .

The material properties for the mixture are usually represented as mole fraction weighted averages of the I constituents. That is

$$\begin{aligned} (\rho C) &= (\rho C)_{mix} = \sum_{i=1}^I [N_i](\rho C)_i \\ \mu &= (\mu)_{mix} = \sum_{i=1}^I [N_i](\mu)_i \\ k &= (k)_{mix} = \sum_{i=1}^I [N_i](k)_i \end{aligned} \quad (1.9.10)$$

where the constituent properties could still be functions of the temperature or other variables.

The equations described above include a wide spectrum of multistep chemical reactions. For many flow or heat transfer problems, such generality is not required and the chemical process can be simplified. For a polymerization or curing reaction,

or in fact any binary mixture problem, the set of mass conservation equations for the species reduces to a single equation

$$\frac{\partial \alpha}{\partial t} + v_j \frac{\partial \alpha}{\partial x_j} = \frac{\partial}{\partial x_j} \left(D \frac{\partial \alpha}{\partial x_j} \right) + R \quad (1.9.11)$$

where α is now integrated as a mass fraction or extent of reaction variable that varies from 0 to 1. Equation (1.9.11) describes the simple process of a material being transformed from species A to species B with an accompanying change in energy. The reaction ratio R in (1.9.11) would typically be of the Arrhenius type. In a fluid system the heat release from the reaction could cause a buoyancy-induced flow and thus couple the flow problem to (1.9.11). Such a problem will be described in Chapter 5 in the context of an epoxy solidification example.

When the flow system is described by the porous flow equations, additional effects and coupled processes may need to be considered. If the solid matrix is reactive and produces a fluid phase as part of the kinetic reaction, a mass source may be necessary in the continuity equation. In this case, both fluid and thermal transport are directly coupled to the reactions. The mass addition would typically be evaluated with an equation analogous to (1.9.9) with solid density in place of the heat release and the reaction sum limited to solid-to-fluid reactions. More complexities of this type are encountered when multi-phase fluid systems are considered (see, e.g., [26]).

In the case of a thermal conduction problem, the elimination of the fluid velocity allows additional simplification of the species equations by elimination of the advection term in (1.9.4). Also, for most solids, the species diffusion is negligibly small and the conservation equations in (1.9.4) can be reduced to a set of ordinary differential equations

$$\frac{d[N_i]}{dt} = R_i = \sum_{j=1}^J \nu_{ij} r_j \quad (1.9.12)$$

The equations in (1.9.12) must be solved in conjunction with the standard heat conduction equation and the appropriate definitions for the heat source and kinetic parameters. Algorithms for, and numerical studies of, reactive systems of this type will be described in Chapter 3.

1.10 Boundary Conditions

To complete the description of the general initial and/or boundary value problem posed in the previous sections, suitable boundary and initial conditions are required. Boundary conditions are most easily understood and described by considering the fluid mechanics separate from other transport processes. In the following sections, boundary conditions associated with viscous flow, porous flow, and thermal transport are discussed, followed by a discussion of the initial conditions.

1.10.1 Viscous Flow Boundary Conditions

For the fluid dynamic part of the problem, either the velocity components or the total surface stress or traction must be specified on the boundary of the fluid region.

In general the boundary conditions can be classified into two types (see Reddy [27], pp. 28–32), as given below.

Dirichlet or essential boundary conditions:

$$v_i = f_i^v(s_k, t) \quad \text{on } \Gamma_v \quad (1.10.1a)$$

Neumann or natural boundary conditions:

$$\mathcal{T}_i = \sigma_{ij}(s_k, t)n_j(s_k) = f_i^T(s_k, t) \quad \text{on } \Gamma_T \quad (1.10.1b)$$

where s_k are the coordinates along the boundary, t is the time, n_i is the outward unit normal to the boundary, and Γ_f is the total boundary enclosing the fluid domain, Ω_f , with $\Gamma_f = \Gamma_v \cup \Gamma_T$ (see Figure 1.10.1). Note that the conditions written in (1.10.1a,b) are in component form, i.e., there must be a condition on each component of the velocity. The specified functions f_i^v and f_i^T are generally simple expressions for standard situations where the fluid is contained by fixed boundaries ($f_i^v = 0$) or planes/lines of symmetry ($f_i^T = 0$), or enters/leaves the domain Ω_f ($f_i^T = \text{constant}$). The boundary conditions listed in (1.10.1a,b) are adequate for most situations involving a single fluid in an uncomplicated domain.

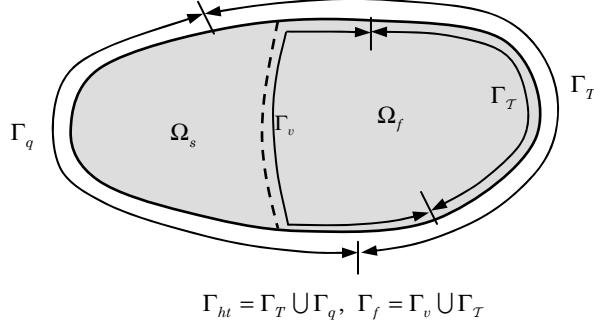


Figure 1.10.1: Schematic for boundary condition definitions.

A slight generalization of the previous Dirichlet and Neumann condition leads to the possibility of representing periodic conditions in a flow. In this case the velocity field would be specified by

$$v_i(s_k, t) = v_i(s_k + \Delta s_k, t) \quad (1.10.2a)$$

and the traction condition by

$$\mathcal{T}_i(s_k, t) = \mathcal{T}_i(s_k + \Delta s_k, t) + g_i^T(s_k + \Delta s_k, t) \quad (1.10.2b)$$

Here the quantity Δs_k indicates a spatial offset from one boundary to another; this offset is usually a simple translation or rotation. The velocity components on the periodic surfaces are equal, but unknown. The tractions on the two surfaces may differ by a function g_i^T . The most common situation is for the normal tractions

(essentially the pressure) to have an offset equal to the pressure change over the length Δs_k , while the tangential tractions are equal.

In some specialized applications it may be required to specify that the fluid slips along a solid surface. In this case the tangential Dirichlet conditions on the surface would be replaced by traction conditions of the form

$$\alpha \sigma_{ij} n_j t_i = (v_i - v_i^s) t_i \quad (1.10.3)$$

where α is the slip coefficient, t_i are the components of the tangent vector to the surface and v_i^s is the velocity component at the surface.

The general boundary or interface conditions between two immiscible fluids are more complex and are described here for several important types of problems. It is assumed that there is no mass transfer across the interface which allows all components of the velocity field to be continuous across the boundary. Note that this condition is often retained even for change of phase problems in order to simplify the formulation. In concert with this velocity specification, a kinematic condition is also imposed that requires the interface to remain an interface, i.e., the interface is a stream surface. This requirement can be represented by the equation

$$\frac{DF}{Dt} = \frac{\partial F}{\partial t} + v_j \frac{\partial F}{\partial x_j} = 0 \quad (1.10.4)$$

where $F(x_i, t)$ is the equation for the fluid interface. Since the location and shape of the interface between two or more fluids is usually not known a priori, this type of problem is termed a free boundary problem. For time-dependent free surface flows, Eq. (1.10.4) provides an evolution equation for the free surface that must be solved in conjunction with the viscous flow equations; the fluid domain Ω_f is now a function of time, $\Omega_f(t)$. In the case of a time-independent flow the kinematic condition simplifies to

$$v_i n_i = 0 \quad (1.10.5)$$

which simply states that the free surface is a stream surface. For the steady case there is no explicit equation for the free surface; equation (1.10.5) only provides a constraint condition that must be applied along the unknown interface. The numerical treatment of free surface flows will be considered in Chapter 4.

The final condition at the free surface specifies that the change in total stress (traction) between the two fluids is related to surface tension forces. The interfacial stress condition is expressed as

$$\sigma_{ij}^2 n_j - \sigma_{ij}^1 n_j = \gamma \left(\frac{1}{R_1} + \frac{1}{R_2} \right) n_i - (\delta_{ij} - n_i n_j) \frac{\partial \gamma}{\partial x_j} \quad (1.10.6)$$

where γ is the surface tension, R_1 and R_2 are the principal radii of curvature of the interface, and the superscripts “2” and “1” on σ_{ij} refer to the two fluids. The last term in (1.10.6) allows variations in the surface tension (e.g., due to temperature) to produce a shear stress in the fluid (Marangoni flows).

The complex conditions expressed in (1.10.6) are necessary when the motion in both fluids is of equal importance (e.g., two immiscible liquids). However, in many

cases, one of the fluids is a gas and its motion relative to a liquid may be neglected. For this simplified case the continuity condition on the velocity is removed since the gas motion is ignored, and shear stresses (drag forces) on the interface are neglected. If the gas is assumed to be at a uniform pressure, P_{amb} , then (1.10.6) becomes

$$\sigma_{ij}n_j = \gamma \left(\frac{1}{R_1} + \frac{1}{R_2} \right) n_i - P_{amb} n_i \quad (1.10.7)$$

which relates the stress normal to the free surface to the surface tension, surface curvature and ambient external pressure.

Other types of computational boundary conditions, such as multipoint constraints, are possible and may be encountered in various modeling or simulation situations. These conditions have their theoretical basis in the conditions outlined here but are best described in detail as part of the numerical algorithm.

1.10.2 Porous Flow Boundary Conditions

The flow conditions that can be applied to the boundary of a fluid-saturated porous medium depend on the specific model used to describe the problem. For the general Forchheimer–Brinkman equation listed in (1.6.2), the admissible boundary conditions are the same as those given for the Navier–Stokes equations, namely, Eq. (1.10.1a,b). However, in cases where the Brinkman terms are excluded from the momentum equation ($\mu_e = 0$), the order of the equation is reduced and fewer boundary conditions are needed. A Forchheimer or simple Darcy model allows the following types of boundary conditions

$$v_i n_i = f^v(s_k, t) \quad \text{on } \Gamma_v \quad (1.10.8a)$$

$$\mathcal{T}_i n_i = -P(s_k, t) \delta_{ij} n_j(s_k) n_i = -P(s_k, t) = f^T(s_k, t) \quad \text{on } \Gamma_T \quad (1.10.8b)$$

In essence these conditions state that the fluid velocity normal to the boundary may be specified (outflow/inflow) or the normal force (pressure) on the boundary may be imposed. These simplified models do not allow a tangential velocity to be imposed (e.g., no-slip walls are not admissible) nor the specification of viscous forces. Also, note that only one type of boundary condition can be specified at any point on the boundary.

For many applications a saturated porous layer will adjoin a clear fluid region and certain continuity conditions will be required at the open interface. The applicable conditions vary with the type of porous media model and in many cases have not been rigorously verified. The major difficulty stems from the heterogeneous nature of the porous medium and the fact that a continuum description such as (1.6.1)–(1.6.3) is derived by an averaging process. Averaging within a distance $\approx \sqrt{\kappa}$, where κ is the permeability, of a boundary or interface is not valid and some loss of information will occur; detailed microscopic analysis or observations must be used to generate valid macroscopic boundary conditions for the continuum models. In both the Darcy and Brinkman formulations (seepage) the velocity normal to the interface is assumed continuous; the Brinkman model assumes all tangential velocity components are also continuous between the porous layer and the clear

fluid. Many investigators have assumed that all components of the viscous stress and the pressure are continuous at an interface with a Brinkman formulation, a situation that is very convenient from a computational point of view. However, Nield [16] and Nield and Bejan [17] believe that the stresses and pressure are not continuous, and they advocate the use of a Navier (slip) condition for the tangential stress at the interface. No proposal has been offered for estimating the changes in normal stress or pressure at the interface. The situation for the Darcy model is just as confused since no stress terms are present in this formulation. The standard assumptions in this case are that the pressure is continuous and the shear stress in the bulk fluid is related to the tangential velocity in the porous layer by the Navier type condition proposed by Beavers and Joseph [18]

$$\frac{\partial v_f}{\partial n} = \frac{\alpha}{\sqrt{\kappa}} (v_f - v_p) \quad (1.10.9)$$

Here v_f is the clear fluid velocity, v_p is the velocity in the porous media, and α is a dimensionless (porous) material parameter. Based on the above discussion it is clear that some caution must be used when considering interface and boundary conditions for a saturated porous material. Further theoretical discussions of interface conditions are considered in [28,29] and numerical demonstrations of various combinations of formulations and interface conditions are provided, for example, in [20,30,31].

1.10.3 Thermal and Transport Boundary Conditions

The thermal part of the boundary value problem for the fluid or solid requires the temperature (Dirichlet or essential condition) or the heat flux (Neumann or natural condition) to be specified on all parts of the boundary enclosing the heat transfer region:

$$T = f^T(s_k, t) \quad \text{on } \Gamma_T \quad (1.10.10a)$$

$$-\left(k_{ij}\frac{\partial T}{\partial x_j}\right)n_i \equiv q_i n_i = q_c + q_r + q_a = f^q(s_k, t) \quad \text{on } \Gamma_q \quad (1.10.10b)$$

where Γ_{ht} is the total boundary enclosing the heat transfer region, and $\Gamma_{ht} = \Gamma_T \cup \Gamma_q$. Note that the condition on Γ_q for a fluid or an isotropic solid can be simplified, since k_{ij} reduces to a scalar k . Also, q_a refers to a specified flux and q_c and q_r refer to the convective and radiative flux components given by

$$q_c = h_c(s_k, T, t)(T - T_c) \quad (1.10.11a)$$

$$q_r = h_r(s_k, T, t)(T - T_r) \quad (1.10.11b)$$

In Eq. (1.10.11a), h_c is the convective heat transfer coefficient which, in general, depends on the location on the boundary, temperature, and time, and T_c is a reference (or sink) temperature for convective transfer. Also, the effective radiation heat transfer coefficient is $h_r = \mathcal{F}\sigma(T + T_r)(T^2 + T_r^2)$ where σ is the Stefan–Boltzmann constant, \mathcal{F} is a form factor, and T_r the reference temperature for radiative transfer. The form factor \mathcal{F} is related to the boundary emissivity ϵ and

the position of the boundary relative to surrounding surfaces (see Arpaci [32]). Note that this type of radiation boundary condition is appropriate when a body or surface radiates to a black body environment that can be characterized by a single temperature. A more complex condition of surface to surface or enclosure radiation is described in Section 1.12. As in the case of the fluid boundary conditions, the specified temperature f^T and total flux f^q are simple functions for the cases normally encountered in practical problems. One of the more complex conditions occurs in change of phase problems and this will be treated separately in the next section.

Another condition that is of concern when considering a material interface between two or more solid regions is the problem of gap or contact resistance. In some situations, the thermal resistance at the interface is attributed to a fictitious material that has a (generally) negligible specific heat and a nonlinear conductivity. The boundary or interface conditions in this situation are the usual continuity conditions on temperature and heat flux since the gap resistance is dictated by property variations. A more mathematical representation of contact resistance provides that the heat flux across the interface be described by an internal boundary condition of the general form given in Eqs. (1.10.11a,b):

$$q_{gap} = h_{gap}(s_k, T_{gap}, t)(T_M - T_S) \quad (1.10.12)$$

where h_{gap} is an effective heat transfer coefficient for the contact surface, and T_{gap} is an average temperature between T_M and T_S . The subscripts M and S designate the “master” and “slave” sides of the contact surface, a distinction that is important in the numerical implementation of this condition. Note that in the limit as h_{gap} becomes large, the temperatures of the two surfaces will become equal, a situation that is useful in some applications.

As a final point on the boundary condition specification note that the boundaries Γ_f and Γ_{ht} need not coincide (see Figure 1.10.1). Indeed, for problems involving regions of both convection and solid body conduction, the boundary regions are not the same. When both fluid and solid regions exist in a problem, continuity conditions on the temperature and heat flux must prevail along the interface of the two materials. Examples of this type of conjugate problem will be given in a later section.

Although not specifically written here, boundary and initial conditions for the auxiliary transport variables are required if these equations are included in the formulation. The form of these boundary conditions generally follows the mathematical form of the boundary conditions associated with the energy equation for a fluid region. This is due to the close similarity between the transport equation (1.8.1) and the energy equation (1.5.9). Thus the boundary conditions on ϕ_i are the same as those given in Eq. (1.10.10) with T replaced by ϕ_i and the radiation condition neglected.

1.10.4 Initial Conditions

For time-dependent problems, a set of initial conditions is necessary for the dependent variables. Very often these conditions consist of a solid body at a uniform temperature and a quiescent fluid at a uniform temperature and hydrostatic pressure. A second common possibility is for the transient motion to be initiated

from an established steady-state flow and temperature field. In all cases, the dependent variables must be known for all \mathbf{x} at $t = 0$ and must satisfy the basic conservation equations (e.g., the initial fluid velocity field must be divergence free and have a compatible pressure field).

1.11 Change of Phase

Many problems of importance involving conductive and convective heat transfer have the added complication of a material change of phase. Such problems are part of a general class of boundary value problems termed *free* or *moving boundary problems* (see Ockendon and Hodgkins [33]). The major difficulty in analyzing such problems is the occurrence of a boundary (or interface) surface, across which certain jump conditions must be satisfied and whose location is unknown a priori. Thus, the location of the moving boundary becomes part of the solution. In this section the equations for a limited class of such problems will be described.

The present discussion will be limited to melting/solidification phase transitions where the solid phase is stationary. Also, it is assumed that density changes upon change of phase may be neglected and that the effects of latent heat release may be accounted for through an appropriate modification of the specific heat. The present development also ignores the explicit modeling of the porous layer or mushy zone that may occur between the solidus and liquidus temperatures in some materials. Methods for treating the mushy zone are easily incorporated into a numerical simulation following the works of Voller and Prakash [34] and Prescott, Incropera, and Bennion [35], and primarily involve the use of the porous media equations described previously. With these assumptions, the phase change problem is described by the equations of the previous sections plus a special set of conditions on the melt/solid interface. At the phase change boundary the following conditions are required:

$$v_i|_X = 0 \quad (1.11.1)$$

$$T_f|_X = T_s|_X \quad (1.11.2)$$

$$k_f \frac{\partial T}{\partial n} \Big|_X - k_s \frac{\partial T}{\partial n} \Big|_X = \rho L \frac{\partial X}{\partial t} = \rho L v^* \quad (1.11.3)$$

where L is the latent heat, $X(t)$ is the spatial position of the phase boundary, v^* is the velocity of the surface, and the subscripts f and s refer to fluid and solid, respectively. The function $X(t)$ is unknown and must be determined as part of the solution. Basically, the melt/solid interface is taken to be a no-slip surface [Eq. (1.11.1)], with continuous temperature [Eqs. (1.11.2)] and a jump in the heat flux [Eq. (1.11.3)]. Note that the solid is assumed stationary; a moving solid would require an additional velocity on the right-hand side of Eq. (1.11.3). Following the work of Bonacini, et al. [36] and Comini, et al. [37], the jump condition in Eq. (1.11.3) may be rewritten using the so-called “enthalpy method.”

By observing that the latent heat, L , corresponds to the isothermal change in the enthalpy, H , for a material at the transition temperature, T_t , the following relation can be introduced

$$H(T) = \int_{T_{ref}}^T C(\xi) d\xi + L \eta(T - T_t) \quad (1.11.4)$$

with

$$\eta(T) = \begin{cases} 1 & \text{if } T \geq 0 \\ 0 & \text{if } T < 0 \end{cases} \quad (1.11.5)$$

where η is the Heaviside step function with argument T . The equivalent specific heat, C^* , is then introduced by

$$C^*(T) = \frac{dH}{dT} = C(T) + L \delta(T - T_t) \quad (1.11.6)$$

where δ is the Dirac delta function. Through the use of Eq. (1.11.6) latent heat effects may be included via the specific heat function, and the jump in the heat flux [see Eq. (1.11.3)] is eliminated from the problem formulation. This particular approach to the problem has a theoretically sound basis, as outlined by Bonacini, et al. [36]. Moreover, it is a computationally effective modification since a two-region problem with a jump condition has been converted to a single-region problem with rapidly varying properties. For use in a finite element model, Eq. (1.11.6) requires additional modification, the details of which will be explained in a later section.

The enthalpy method described above is the most heavily used formulation for the phase change problem but certainly not the only method available. Another approach for treating the latent heat is the heat source method. The development follows the enthalpy method up to Eq. (1.11.6) but proceeds one step further. When Eq.(1.11.6) is substituted into the energy equation, the latent heat term may be viewed as a temperature dependent heat source and moved to the right-hand side of the energy balance. An effective heat source is then

$$Q_{lh} = \rho L \frac{\partial T}{\partial t} \delta(T - T_t) \quad (1.11.7)$$

which is active only when the local temperature is at the transition temperature. For computational purposes, (1.11.7) requires some modification because of the difficulties inherent in the delta function $\delta(\cdot)$.

A final method for resolution of the phase change problem draws on the similarities between a chemical reaction with energy release and the phase transition process. This method requires that the phase change be viewed as a reaction with material A going to material B with an appropriate “kinetic” description. The energy release from this reaction appears as a continuous source term in the energy balance with its magnitude dependent on the reaction rates. This type of formulation avoids many of the computational difficulties associated with the enthalpy and heat source methods, though it does introduce the complexity of treating a chemically reactive material. Details of this method are presented in Chapter 3.

The previous phase change discussion has been presented under the rather strong assumption that any density changes involved in the phase transition were negligible. The relaxation of this assumption leads to a significant increase in problem complexity. Though the no-slip velocity condition in Eq. (1.11.1) continues to apply for the velocity components tangent to the interface, a normal mass balance augments Eq. (1.11.1) for the velocity normal to the interface,

$$\rho_f n_j (v_j^f - v_j^*) = \rho_s n_j (v_j^s - v_j^*) \quad (1.11.8)$$

with the special case of a stationary solid being

$$\rho_f n_j (v_j^f - v_j^*) = -\rho_s n_j v_j^* \quad (1.11.9)$$

The subscripts and superscripts s and f refer to the fluid and solid. A nonzero essential velocity condition, such as the one in Eq. (1.11.8) or (1.11.9), is difficult to implement in many computational schemes. This velocity condition implies that some specific algorithm is required to explicitly locate, resolve, and follow the phase boundary. Though a number of computational methods exist for this type of problem, this boundary condition is sufficiently difficult to keep the constant density assumption as a standard procedure.

1.12 Enclosure Radiation

Radiant energy exchange between neighboring surfaces of a region or between a region and its surroundings can produce large effects in the overall heat transfer problem. Though the radiation effects generally enter the heat transfer problem only through the boundary conditions, the coupling is especially strong due to the nonlinear dependence of the radiation on the surface temperature. The present description considers a restricted class of radiation problems that ignore participation of the fluid phase; the radiative exchange only influences the flow and/or solid body conduction problem through modification of the surface temperature distribution.

Enclosure or surface-to-surface radiation is limited to diffuse, gray, opaque surfaces. This assumption implies that all energy emitted or reflected from a surface is diffuse. Further, surface emissivity ϵ , absorbtivity α , and reflectivity ρ are independent of wavelength and direction so that $\epsilon(T) = \alpha(T) = 1 - \rho(T)$. Each individual area or surface that is considered in the radiation process must be at a uniform temperature; emitted and reflected energy are uniform over each such surface. Note that the definition of a surface is arbitrary and can be based on geometry alone or be defined to specifically satisfy the uniform temperature criterion.

With the above assumptions, the radiation problem can be approached using the net-radiation method (see Siegel and Howell [38]). For purposes of discussion, consider the two-dimensional enclosure made up of N distinct surfaces as shown in Figure 1.12.1. Associated with each surface is a uniform temperature \bar{T}_j , an area A_j , and a surface emissivity ϵ_j . An energy balance for each surface in the enclosure leads to the following system of equations

$$\sum_{j=1}^N \left[\frac{\delta_{kj}}{\epsilon_j} - F_{k-j} \left(\frac{1 - \epsilon_j}{\epsilon_j} \right) \right] \frac{Q_j}{A_j} = \sum_{j=1}^N (\delta_{kj} - F_{k-j}) \sigma \bar{T}_j^4 \quad (1.12.1)$$

Equation (1.12.1) relates the net energy loss, Q_j , from each surface to the temperature of each surface, where δ_{kj} is the unit tensor, σ is the Stefan–Boltzmann constant, and F_{k-j} are radiation view (configuration) factors. The view factor is

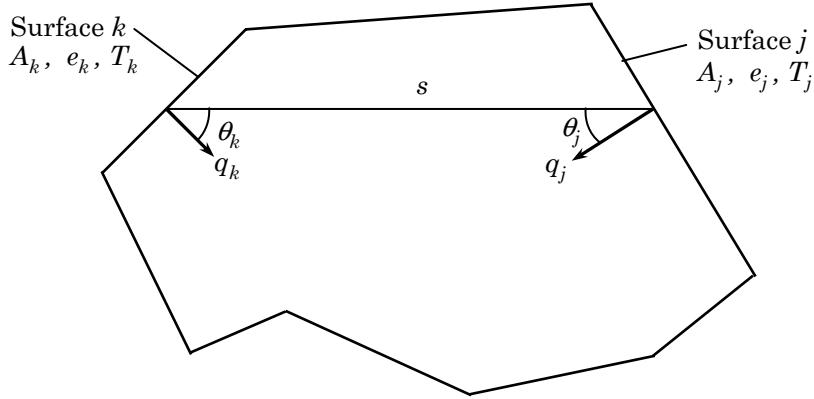


Figure 1.12.1: Nomenclature for enclosure radiation.

defined as the fraction of energy leaving a surface that arrives at a second surface. For surfaces with finite areas, the view factors are defined by

$$F_{k-j} = \frac{1}{A_k} \int_{A_k} \int_{A_j} \frac{\cos \theta_k \cos \theta_j}{\pi S^2} dA_j dA_k \quad (1.12.2)$$

where S is the distance from a point on surface A_j to a point on surface A_k . The angles θ_j and θ_k are measured between the line S and the normals to the surface as shown in Figure 1.12.1. It is clear from Eq. (1.12.2) that the view factors are purely geometric quantities that can in principle be evaluated for any given distribution of surfaces. Although not explicitly shown in the figure, note that view factors may be modified by the presence of intervening bodies or surfaces that shadow surface A_j from surface A_k . Methods for evaluating F_{k-j} will be outlined later in the book.

For purposes of computation it is convenient to rearrange Eq. (1.12.1) into the following equations:

$$\sum_{j=1}^N [\delta_{kj} - (1 - \epsilon_k) F_{k-j}] q_j^o = \epsilon_k \sigma \bar{T}_k^4 \quad (1.12.3)$$

$$q_k = q_k^o - \sum_{j=1}^N F_{k-j} q_j^o \quad (1.12.4)$$

Equations (1.12.3) and (1.12.4) are expressed in terms of the outgoing radiative flux for each surface, q_j^o , and the net flux from each surface $q_k = Q_k/A_k$. For known surface temperatures \bar{T}_k in the enclosure, Eq. (1.12.3) can be solved for the outgoing radiative flux at each surface. Equation (1.12.4) then allows the net flux at each surface to be evaluated. The actual method of solution using this formulation in a finite element context will be discussed in the later chapters.

1.13 Summary of Equations

All of the governing differential equations, constitutive relations, and boundary conditions of heat transfer and fluid mechanics relevant to the present discussion were presented in the previous sections. It is useful to summarize the complete boundary value problem in one section. Therefore, the complete set of equations is presented here in Cartesian component form for each physical process. A sum on repeated indices is assumed, and the free indices take on the values of 1 to 3.

Flows of Viscous Incompressible Fluids

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (1.13.1)$$

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 g_i - \rho_0 \hat{g}_i \quad (1.13.2)$$

where

$$\hat{g}_i = g_i [\beta(T - T_0) + \gamma_1(c_1 - c_{10}) + \gamma_2(c_2 - c_{20}) + \dots] \quad (1.13.3)$$

Porous Flow

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (1.13.4)$$

$$\rho_0 \alpha_{ij} \frac{\partial v_j}{\partial t} + \left(\frac{\rho_0 \hat{c}}{\sqrt{\kappa}} \|\mathbf{v}\| + \frac{\mu}{\kappa} \right) v_i = \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu_e \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 g_i - \rho_0 \hat{g}_i \quad (1.13.5)$$

where

$$\hat{g}_i = g_i [\beta(T - T_0) + \gamma_1(c_1 - c_{10}) + \gamma_2(c_2 - c_{20}) + \dots] \quad (1.13.6)$$

Heat Transfer (Convection)

$$\rho_0 C \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + Q + \Phi \quad (1.13.7)$$

Heat Transfer (Conduction)

$$\rho_s C_s \left(\frac{\partial T}{\partial t} \right) = \frac{\partial}{\partial x_i} \left(k_{s_{ij}} \frac{\partial T}{\partial x_j} \right) + Q_s \quad (1.13.8)$$

Auxiliary Transport

$$C_i \left(\frac{\partial \phi_i}{\partial t} + v_j \frac{\partial \phi_i}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(D_i \frac{\partial \phi_i}{\partial x_j} \right) + Q_i \quad (1.13.9)$$

Flows of Viscous Compressible (Acoustically Filtered) Fluids

$$\frac{\partial \rho v_i}{\partial x_i} = -\frac{\partial \rho}{\partial t} \quad (1.13.10)$$

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(-P' \delta_{ij} + \frac{\partial \tau_{ij}}{\partial x_j} \right) + \rho g_i \quad (1.13.11)$$

$$\rho C_v \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) + \beta T \frac{dP_0}{dt} + Q + \Phi \quad (1.13.12)$$

where

$$\tau_{ij} = \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) - \frac{2}{3} \mu \left(\frac{\partial v_k}{\partial x_k} \right) \delta_{ij} \quad (1.13.13)$$

$$\rho = \rho(T, P_0) \quad (1.13.14)$$

$$P(x_i, t) = P_0(t) + P'_0(x_i, t) \quad (1.13.15)$$

Equations (1.13.1)–(1.13.15) with the noted boundary and initial conditions form a complete set for the determination of the velocity, pressure, temperature, and auxiliary variables in a fluid, porous, or solid region. Embedded in this description are numerous classes of flow problems that can be recovered through the omission or inclusion of specific equations, the neglect of certain terms within an equation, and the variation of material properties.

Problems

- 1.1** Show that $\nabla \mathbf{v}$ in the cylindrical coordinate system is given by

$$\begin{aligned} \nabla \mathbf{v} = & \hat{\mathbf{e}}_r \hat{\mathbf{e}}_r \frac{\partial v_r}{\partial r} + \hat{\mathbf{e}}_r \hat{\mathbf{e}}_\theta \frac{\partial v_\theta}{\partial r} + \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_r \frac{1}{r} \left(\frac{\partial v_r}{\partial \theta} - v_\theta \right) \\ & + \hat{\mathbf{e}}_r \hat{\mathbf{e}}_z \frac{\partial v_z}{\partial r} + \hat{\mathbf{e}}_z \hat{\mathbf{e}}_r \frac{\partial v_r}{\partial z} + \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_\theta \frac{1}{r} \left(v_r + \frac{\partial v_\theta}{\partial \theta} \right) \\ & + \frac{1}{r} \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_z \frac{\partial v_z}{\partial \theta} + \hat{\mathbf{e}}_z \hat{\mathbf{e}}_\theta \frac{\partial v_\theta}{\partial z} + \hat{\mathbf{e}}_z \hat{\mathbf{e}}_z \frac{\partial v_z}{\partial z} \end{aligned}$$

- 1.2** Show that $\nabla \mathbf{v}$ in the spherical coordinate system is given by

$$\begin{aligned} \nabla \mathbf{v} = & \frac{\partial v_R}{\partial R} \hat{\mathbf{e}}_R \hat{\mathbf{e}}_R + \frac{\partial v_\phi}{\partial R} \hat{\mathbf{e}}_R \hat{\mathbf{e}}_\phi + \frac{\partial v_\theta}{\partial R} \hat{\mathbf{e}}_R \hat{\mathbf{e}}_\theta \\ & + \frac{1}{R} \left(\frac{\partial v_R}{\partial \phi} - v_\phi \right) \hat{\mathbf{e}}_\phi \hat{\mathbf{e}}_R + \frac{1}{R} \left(\frac{\partial v_\phi}{\partial \phi} + v_R \right) \hat{\mathbf{e}}_\phi \hat{\mathbf{e}}_\phi + \frac{1}{R} \frac{\partial v_\theta}{\partial \phi} \hat{\mathbf{e}}_\phi \hat{\mathbf{e}}_\theta \\ & + \frac{1}{R \sin \phi} \left[\left(\frac{\partial v_R}{\partial \theta} - v_\theta \sin \phi \right) \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_R + \left(\frac{\partial v_\phi}{\partial \theta} - v_\theta \cos \phi \right) \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_\phi \right. \\ & \left. + \left(\frac{\partial v_\theta}{\partial \theta} + v_R \sin \phi + v_\phi \cos \phi \right) \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_\theta \right]. \end{aligned}$$

- 1.3** For an arbitrary second-order tensor \mathbf{S} show that $\nabla \cdot \mathbf{S}$ in the cylindrical coordinate system is given by

$$\begin{aligned} \nabla \cdot \mathbf{S} = & \left[\frac{\partial S_{rr}}{\partial r} + \frac{1}{r} \frac{\partial S_{\theta r}}{\partial \theta} + \frac{\partial S_{zr}}{\partial z} + \frac{1}{r} (S_{rr} - S_{\theta \theta}) \right] \hat{\mathbf{e}}_r \\ & + \left[\frac{\partial S_{r\theta}}{\partial r} + \frac{1}{r} \frac{\partial S_{\theta\theta}}{\partial \theta} + \frac{\partial S_{z\theta}}{\partial z} + \frac{1}{r} (S_{r\theta} + S_{\theta r}) \right] \hat{\mathbf{e}}_\theta \\ & + \left[\frac{\partial S_{rz}}{\partial r} + \frac{1}{r} \frac{\partial S_{\theta z}}{\partial \theta} + \frac{\partial S_{zz}}{\partial z} + \frac{1}{r} S_{rz} \right] \hat{\mathbf{e}}_z. \end{aligned}$$

- 1.4** For an arbitrary second-order tensor \mathbf{S} show that $\nabla \times \mathbf{S}$ in the cylindrical coordinate system is given by

$$\begin{aligned}\nabla \times \mathbf{S} = & \hat{\mathbf{e}}_r \hat{\mathbf{e}}_r \left(\frac{1}{r} \frac{\partial S_{zr}}{\partial \theta} - \frac{\partial S_{\theta r}}{\partial z} - \frac{1}{r} S_{z\theta} \right) + \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_\theta \left(\frac{\partial S_{r\theta}}{\partial z} - \frac{\partial S_{z\theta}}{\partial r} \right) + \\ & \hat{\mathbf{e}}_z \hat{\mathbf{e}}_z \left(\frac{1}{r} S_{\theta z} - \frac{1}{r} \frac{\partial S_{rz}}{\partial \theta} + \frac{\partial S_{\theta z}}{\partial r} \right) + \hat{\mathbf{e}}_r \hat{\mathbf{e}}_\theta \left(\frac{1}{r} \frac{\partial S_{z\theta}}{\partial \theta} - \frac{\partial S_{\theta\theta}}{\partial z} + \frac{1}{r} S_{zr} \right) + \\ & \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_r \left(\frac{\partial S_{rr}}{\partial z} - \frac{\partial S_{zr}}{\partial r} \right) + \hat{\mathbf{e}}_r \hat{\mathbf{e}}_z \left(\frac{1}{r} \frac{\partial S_{zz}}{\partial \theta} - \frac{\partial S_{\theta z}}{\partial z} \right) + \\ & \hat{\mathbf{e}}_z \hat{\mathbf{e}}_r \left[\frac{\partial S_{\theta r}}{\partial r} - \frac{1}{r} \frac{\partial S_{rr}}{\partial \theta} + \frac{1}{r} (S_{r\theta} + S_{\theta r}) \right] + \hat{\mathbf{e}}_\theta \hat{\mathbf{e}}_z \left(\frac{\partial S_{rz}}{\partial z} - \frac{\partial S_{zz}}{\partial r} \right) + \\ & \hat{\mathbf{e}}_z \hat{\mathbf{e}}_\theta \left[\frac{\partial S_{\theta\theta}}{\partial r} + \frac{1}{r} (S_{\theta\theta} - S_{rr}) - \frac{1}{r} \frac{\partial S_{r\theta}}{\partial \theta} \right].\end{aligned}$$

- 1.5** For an arbitrary second-order tensor \mathbf{S} show that $\nabla \cdot \mathbf{S}$ in the spherical coordinate system is given by

$$\begin{aligned}\nabla \cdot \mathbf{S} = & \left\{ \frac{\partial S_{RR}}{\partial R} + \frac{1}{R} \frac{\partial S_{\phi R}}{\partial \phi} + \frac{1}{R \sin \phi} \frac{\partial S_{\theta R}}{\partial \theta} \right. \\ & \left. + \frac{1}{R} [2S_{RR} - S_{\phi\phi} - S_{\theta\theta} + S_{\phi R} \cot \phi] \right\} \hat{\mathbf{e}}_R \\ & + \left\{ \frac{\partial S_{R\phi}}{\partial R} + \frac{1}{R} \frac{\partial S_{\phi\phi}}{\partial \phi} + \frac{1}{R \sin \phi} \frac{\partial S_{\theta\phi}}{\partial \theta} \right. \\ & \left. + \frac{1}{R} [(S_{\phi\phi} - S_{\theta\theta}) \cot \phi + S_{\phi R} + 2S_{R\phi}] \right\} \hat{\mathbf{e}}_\phi \\ & + \left\{ \frac{\partial S_{R\theta}}{\partial R} + \frac{1}{R} \frac{\partial S_{\phi\theta}}{\partial \phi} + \frac{1}{R \sin \phi} \frac{\partial S_{\theta\theta}}{\partial \theta} \right. \\ & \left. + \frac{1}{R} [(S_{\phi\theta} + S_{\theta\phi}) \cot \phi + 2S_{R\theta} + S_{\theta R}] \right\} \hat{\mathbf{e}}_\theta.\end{aligned}$$

- 1.6** Let $\rho(\mathbf{x}, t)$ denote the mass density of a continuous region, Ω . Then conservation of mass for a material region requires that

$$\frac{D}{Dt} \int_{\Omega} \rho \, d\Omega = 0 \quad (1)$$

Show that for a fixed region, conservation of mass can also be expressed as

$$\int_{\Omega} \frac{D\rho}{Dt} \, d\Omega + \int_{\Gamma} \rho \hat{\mathbf{n}} \cdot \mathbf{v} \, d\Gamma = 0 \quad (2)$$

Interpret the equation in physical terms.

- 1.7** Newton's second law of motion (or the principle of conservation of linear momentum) applied to a continuum states that the rate of change of momentum following a material region Ω of fixed mass is equal to the sum of all the forces acting on the region. This can be expressed as

$$\frac{D}{Dt} \int_{\Omega} \rho \mathbf{v} \, d\Omega = \int_{\Gamma} \hat{\mathbf{n}} \cdot \boldsymbol{\sigma} \, d\Gamma + \int_{\Omega} \rho \mathbf{f} \, d\Omega \quad (1)$$

where $\boldsymbol{\sigma}$ is the total stress tensor, \mathbf{f} is the body force per unit mass, ρ is the mass density, and \mathbf{v} is the material velocity. Since the material particle mass $\rho d\Omega$ is constant with respect to the material time derivative D/Dt , make use of the divergence theorem and obtain the differential form of Newton's second law of motion:

$$\rho \frac{D\mathbf{v}}{Dt} = \nabla \cdot \boldsymbol{\sigma} + \rho \mathbf{f} \quad (2)$$

- 1.8** Let ϵ denote the thermodynamic internal energy per unit mass of a material. Then the first law of thermodynamics applied to a material region Ω can be written as

$$\frac{D}{Dt} \int_{\Omega} \rho(e + \frac{1}{2}\mathbf{v} \cdot \mathbf{v}) d\Omega = \int_{\Gamma} \hat{\mathbf{n}} \cdot \boldsymbol{\sigma} \cdot \mathbf{v} d\Gamma + \int_{\Omega} \rho \mathbf{f} \cdot \mathbf{v} d\Omega - \int_{\Gamma} \mathbf{q} \cdot \hat{\mathbf{n}} d\Gamma + \int_{\Omega} Q d\Omega \quad (1)$$

The first two terms on the right-hand side describe the rate of work done on the material region by the surface stresses and the body forces. The third integral describes the net outflow of heat from the region, causing a decrease of energy inside the region, where \mathbf{q} denotes the heat flux vector. The fourth integral describes the internal heat generation in the region, where Q denotes the internal heat generation per unit volume. Show that the differential form of the first law of thermodynamics is given by

$$\rho \frac{De}{Dt} = -\nabla \cdot \mathbf{q} + Q + \boldsymbol{\sigma} : \mathbf{D} \quad (2)$$

Make use of the momentum equation and the identity

$$\operatorname{div}(\boldsymbol{\sigma} \cdot \mathbf{v}) - \mathbf{v} \cdot \operatorname{div} \boldsymbol{\sigma} = \boldsymbol{\sigma} : \mathbf{D} \quad (3)$$

For a viscous incompressible fluid, we take $\rho(De/Dt) = \rho C_v(DT/Dt)$.

- 1.9** Use equations (1.5.11)–(1.5.13) in equation (1.5.4) and derive the energy equation in the cylindrical coordinate system (r, θ, z) .
- 1.10** For axisymmetric flows of viscous, incompressible fluids (i.e., flow field is independent of θ -coordinate), show that the Stokes flow equations (i.e., when the nonlinear terms in the convective parts are neglected) are given as follows:

Conservation of Linear Momentum

$$\rho \frac{\partial v_r}{\partial t} = \frac{1}{r} \frac{\partial}{\partial r}(r\sigma_{rr}) - \frac{\sigma_{\theta\theta}}{r} + \frac{\partial \sigma_{rz}}{\partial z} + f_r \quad (1)$$

$$\rho \frac{\partial v_z}{\partial t} = \frac{1}{r} \frac{\partial}{\partial r}(r\sigma_{rz}) + \frac{\partial \sigma_{zz}}{\partial z} + f_z \quad (2)$$

Conservation of Mass

$$\frac{1}{r} \frac{\partial}{\partial r}(rv_r) + \frac{\partial v_z}{\partial z} = 0 \quad (3)$$

Constitutive Relations

$$\begin{aligned} \sigma_{rr} &= -P + 2\mu \frac{\partial v_r}{\partial r}, & \sigma_{\theta\theta} &= -P + 2\mu \frac{v_r}{r} \\ \sigma_{zz} &= -P + 2\mu \frac{\partial v_z}{\partial z}, & \sigma_{rz} &= \mu \left(\frac{\partial v_r}{\partial z} + \frac{\partial v_z}{\partial r} \right) \end{aligned} \quad (4)$$

- 1.11** Using equations (1.5.11)–(1.5.13) in equations (1.5.1)–(1.5.3), derive equations (1.5.16)–(1.5.18).
- 1.12** Use the definition (1.4.23) of the symmetric part of the velocity gradient tensor, \mathbf{D} , in terms of the velocity vector \mathbf{v} and equations (1.5.11) and (1.5.13) to verify the relations in equation (1.5.14).

References for Additional Reading

1. J. N. Reddy *An Introduction to Continuum Mechanics with Applications*, Cambridge University Press, New York (2008).
2. R. B. Bird, W. E. Stewart, and E. N. Lightfoot, *Transport Phenomena*, John Wiley & Sons, New York (1960).
3. L. E. Malvern, *Introduction to the Mechanics of a Continuous Medium*, Prentice Hall, Englewood Cliffs, New Jersey (1969).
4. J. N. Reddy and M. L. Rasmussen, *Advanced Engineering Analysis*, John Wiley & Sons, New York (1982); reprinted by Krieger Publishing, Melbourne, Florida (1991).
5. A. Bejan, *Convection Heat Transfer*, Second Edition, John Wiley & Sons, New York (1995).
6. M. N. Özisik, *Heat Transfer: A Basic Approach*, McGraw-Hill, New York (1985).
7. M. N. Özisik, *Heat Conduction*, Second Edition, John Wiley & Sons, New York (1993).
8. G. K. Batchelor, *An Introduction to Fluid Dynamics*, Cambridge University Press, Cambridge (1967).
9. H. Schlichting, *Boundary-Layer Theory*, (translated by J. Kestin), Seventh Edition, McGraw-Hill, New York (1979).
10. A. Bejan, *Advanced Engineering Thermodynamics*, Second Edition, John Wiley & Sons, New York (1997).
11. D. D. Gray and A. Giorgini, "The Validity of the Boussinesq Approximation for Liquids and Gases," *International Journal of Heat and Mass Transfer*, **19**, 545–551 (1976).
12. D. K. Gartling and C. E. Hickox, "A Numerical Study of the Applicability of the Boussinesq Approximation for a Fluid-Saturated Porous Medium," *International Journal for Numerical Methods in Fluids*, **5**, 995–1013 (1985).
13. S. A. Piacsek and G. Williams, "Conservation Properties of Convection Difference Schemes," *Journal of Computational Physics*, **6**, 393–405 (1970).
14. M. Kaviany, *Principles of Heat Transfer in Porous Media*, Springer-Verlag, New York (1991).
15. D. A. Nield and D. D. Joseph, "Effects of Quadratic Drag on Convection in a Saturated Porous Medium," *Physics of Fluids*, **28**, 995–997 (1985).
16. D. A. Nield, "The Limitations of the Brinkman-Forchheimer Equation in Modeling Flow in a Saturated Porous Medium and at an Interface," *International Journal of Heat and Fluid Flow*, **12**, 269–272 (1991).
17. D. A. Nield and A. Bejan, *Convection in Porous Media*, Springer-Verlag, Berlin (1992).
18. G. S. Beavers and D. D. Joseph, "Boundary Conditions at a Naturally Permeable Wall," *Journal of Fluid Mechanics*, **30**, 197–207 (1967).
19. J. Bear, *Dynamics of Fluids in Porous Media*, American Elsevier, New York (1972).
20. D. K. Gartling, C. E. Hickox and R. C. Givler, "Simulations of Coupled Viscous Porous Flow Problems," *Computational Fluid Dynamics*, **7**, 23–48 (1996).
21. R. G. Rehm and H. R. Baum, "The Equations of Motion for Thermally Driven Buoyant Flows," *Journal of Research of the National Bureau of Standards*, **3**, 297–308 (1978).
22. S. Paolucci, "On the Filtering of Sound from the Navier-Stokes Equations," Sandia National Laboratories Technical Report, SAND 82-8257, Albuquerque, NM (1982).
23. J. Principe and R. Codina, "Mathematical Models for Thermally Coupled Low Speed Flows," *Advances in Theoretical and Applied Mechanics*, **2**, 93–112(2009).
24. G. Hauke and T. J. R. Hughes, "A Comparative Study of Different Sets of Variables for Solving Compressible and Incompressible Flows," *Computer Methods in Applied Mechanics and Engineering*, **153**, 1–44 (1998).

25. J. O. Herschfelder, C. F. Curtiss, and R. B. Bird, *Molecular Theory of Gases and Liquids*, John Wiley & Sons, New York (1954).
26. E. S. Oran and J. P. Boris, *Numerical Simulation of Reactive Flows*, Elsevier, New York (1987).
27. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).
28. J. A. Ochoa-Tapia and S. Whitaker, "Momentum Transfer at the Boundary Between a Porous Medium and a Homogeneous Fluid - I. Theoretical Development," *International Journal of Heat and Mass Transfer*, **38**, 2635–2646 (1995).
29. J. A. Ochoa-Tapia and S. Whitaker, "Momentum Transfer at the Boundary Between a Porous Medium and a Homogeneous Fluid - II. Comparison with Experiment," *International Journal of Heat and Mass Transfer*, **38**, 2647–2655 (1995).
30. A. G. Salinger, R. Aris and J. J. Derby, "Finite Element Formulation for Large-Scale Coupled Flows in Adjacent Porous and Open Fluid Domains," *International Journal for Numerical Methods in Fluids*, **18**, 1185–1209 (1994).
31. P. Yu, T. S. Lee, Y. Zeng and H. T. Low, "A Numerical Method for Flows in Porous and Homogeneous Fluid Domains Coupled at the Interface by Stress Jump," *International Journal for Numerical Methods in Fluids*, **53**, 1755–1775 (2007).
32. V. S. Arpaci, *Conduction Heat Transfer*, Addison-Wesley Publishing, Reading, Massachusetts (1966).
33. J. R. Ockendon and W. R. Hodgkins (Eds.), *Moving Boundary Problems in Heat Flow and Diffusion*, Clarendon Press, Oxford (1975).
34. V. R. Voller and C. Prakash, "A Fixed Grid Numerical Modeling Methodology for Convection Diffusion Mushy Region Phase Change Problems," *International Journal for Numerical Methods in Engineering*, **30**, 1709–1719 (1987).
35. P. J. Prescott, F. P. Incropera, and W. D. Bennion, "Modeling of Dendrite Solidification Systems: Reassessment of the Continuum Momentum Equation," *International Journal of Heat and Mass Transfer*, **34**, 2351–2359 (1991).
36. C. Bonacini, G. Comini, A. Fasano and M. Primicerio, "Numerical Solution of Phase-Change Problems," *International Journal of Heat and Mass Transfer*, **16**, 1825–1832 (1973).
37. G. Comini, S. DelGuidice, R. Lewis, and O. C. Zienkiewicz, "Finite Element Solution of Non-Linear Heat Conduction Problems with Special Reference to Phase Change," *International Journal for Numerical Methods in Engineering*, **8**, 613–624 (1974).
38. R. Siegel and J. R. Howell, *Thermal Radiation Heat Transfer*, McGraw-Hill, New York (1972).

The Finite Element Method

2.1 Introduction

The finite element method is a powerful computational technique for the solution of differential and integral equations that arise in various fields of engineering and applied science. The method is a generalization of the classical variational (i.e., the Ritz) and weighted-residual (e.g., weak-form Galerkin, least-squares, collocation, etc.) methods, which are based on the idea that the solution u of a differential equation can be represented as a linear combination of unknown parameters c_j and appropriately selected functions ϕ_j in the *entire domain* of the problem. The parameters c_j are then determined such that the differential equation is satisfied, often, in a weighted-integral sense. The functions ϕ_j , called the *approximation functions*, are selected such that they satisfy the boundary conditions of the problem. For additional details on the variational and weighted-residual methods, the reader may consult Mikhlin [1,2], Reddy [3-6], and Reddy and Rasmussen [7].

The traditional variational and weighted-residual methods suffer from one major shortcoming: construction of the approximation functions that satisfy the boundary conditions of the problem to be solved. Most real-world problems are defined on regions that are geometrically complex, and therefore it is difficult to generate approximation functions that satisfy different types of boundary conditions on different portions of the boundary of the complex domain. However, if the domain can be represented as a collection of “simple subdomains” that permit construction of the approximation functions for any arbitrary but physically meaningful boundary conditions, then the traditional variational methods can be used to solve practical engineering problems. The basic idea of the finite element method is to view a given domain as an assemblage of simple geometric shapes, called *finite elements*, for which it is possible to systematically generate the approximation functions needed in the solution of differential equations by any of the variational and weighted-residual methods. The ability to represent domains with irregular geometries by a collection of finite elements makes the method a valuable practical tool for the solution of boundary, initial, and eigenvalue problems arising in various fields of engineering. The approximation functions are often constructed using ideas from interpolation theory, and hence they are also called *interpolation functions*. Thus, the finite element method is an element-wise application of the classical variational and weighted-residual methods. For a given differential equation, it is possible to develop different *finite element models*, depending on the choice of a particular variational or weighted-residual method. The finite element model is a set of algebraic relations among the unknown parameters of the approximation.

The primary objective of this chapter is to present a general introduction to the finite element method. For detailed introductions to the finite element method and nonlinear finite element analysis, the reader is advised to consult the textbooks by Reddy [4,5]. The major steps in the finite element formulation and analysis of a typical problem are:

1. Discretization of the domain into a set of selected finite elements. A finite element is not just a geometric shape but it is endowed with certain geometric and physical features, as will be clear in the sequel.
- 2a. Construction of a statement, often a weighted-integral or weak-form statement, that is equivalent (in some sense) to the differential equation to be analyzed over a typical element.
- 2b. Development of the finite element model (i.e., set of algebraic relations among the unknowns) using the weighted-integral statement or weak form over an element. The same differential equation can have different finite element models depending on the choice of the method of approximation (e.g., Galerkin, weak-form Galerkin, least-squares, subdomain, collocation, and so on).
3. Assembly of finite elements to obtain the global system of algebraic equations.
4. Imposition of boundary conditions.
5. Solution of equations.
6. Postcomputation of the solution and quantities of interest.

Steps 2a and 2b are formulative steps while the remaining steps are computational.

In the following sections, we will discuss the first four steps with the help of a model differential equation, namely, the equation governing heat transfer in a two-dimensional solid medium with constant material properties (i.e., linear problem).

2.2 Model Differential Equation

Consider the problem of finding the steady-state temperature T distribution in a two-dimensional orthotropic medium Ω , with boundary Γ . We presume that the material axes coincide with the problem coordinates and that there is no transfer of heat in the perpendicular direction to the plane of the solid. A rectangular Cartesian system is used to describe the problem physics. The equation governing the temperature distribution is given by setting the time derivative term to zero and $x_1 = x, x_2 = y$ in Eq. (1.13.8):

$$-\left[\frac{\partial}{\partial x}\left(k_{xx}\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k_{yy}\frac{\partial T}{\partial y}\right)\right] = Q \quad \text{in } \Omega \quad (2.2.1)$$

where k_{xx} and k_{yy} are conductivities in the x and y directions, respectively, and $Q(x, y)$ is the known internal heat generation per unit area. For a nonhomogeneous conducting medium, the conductivities k_{xx} and k_{yy} are functions of position (x, y) . For an isotropic medium, we set $k_{xx} = k_{yy} = k$ in Eq. (2.2.1) and obtain the Poisson equation

$$-\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) - \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) = Q \quad \text{in } \Omega \quad (2.2.2)$$

or in vector form

$$-\nabla \cdot (k \nabla T) = Q \quad \text{in } \Omega \quad (2.2.3)$$

where ∇ is the gradient operator

$$\nabla = \hat{\mathbf{e}}_x \frac{\partial}{\partial x} + \hat{\mathbf{e}}_y \frac{\partial}{\partial y} \quad (2.2.4)$$

and $\hat{\mathbf{e}}_x$ and $\hat{\mathbf{e}}_y$ denote the unit vectors directed along the x and y axes, respectively.

Equation (2.2.1) must be solved in conjunction with specified boundary conditions of the problem. The following two types of boundary conditions are assumed in the following development [see Eqs. (1.10.10a,b)]:

$$T = f^T(s) \quad \text{on } \Gamma_T \quad (2.2.5a)$$

$$q_n = \left(k_{xx} \frac{\partial T}{\partial x} n_x + k_{yy} \frac{\partial T}{\partial y} n_y \right) + q_c = f^q(s) \quad \text{on } \Gamma_q \quad (2.2.5b)$$

where Γ_T and Γ_q are disjoint portions of the boundary Γ such that $\Gamma = \Gamma_T \cup \Gamma_q$, q_c refers to the convective component of heat flux

$$q_c = h_c(s, T)(T - T_c) \quad (2.2.6)$$

and (n_x, n_y) denote the direction cosines of the unit normal vector on the boundary Γ (see Figure 1.3.2). In Eq. (2.2.6), h_c denotes the convective heat transfer coefficient and T_c the ambient temperature (see Section 1.10.3). The radiative heat transfer boundary condition is not considered in the present development.

2.3 Finite Element Approximation

As stated in the introduction to this chapter, the main feature of the finite element method is to construct approximation functions required in the solution of differential equations by a variational or weighted-integral method. In the finite element method, this is accomplished by subdividing the given domain $\bar{\Omega} = \Omega \cup \Gamma$ into a set of subdomains, called finite elements (see Figure 2.3.1). The phrase *finite element* often refers to both the geometry of the element and degree (or order) of approximation used for the dependent unknown over the element, both of which would be known in the context of the discussion. Any geometric shape for which the approximation functions can be derived uniquely qualifies as an element. We shall discuss various geometric shapes and orders of approximation shortly. To keep the formulative steps very general (i.e., not confine the formulation to a specific geometric shape or order of approximation), we denote the domain of a typical element by the symbol Ω^e and its boundary by Γ^e . The element Ω^e can be a triangle or quadrilateral in shape, and the degree of interpolation over it can be linear, quadratic, and so on. The non-overlapping sum (or *assembly*) of all elements used to represent the actual domain will be denoted by Ω^h , and it is called the *finite element mesh* of the domain Ω . In general, Ω^h may not equal the actual domain Ω because of the geometric complexity of the actual domain. Of course, for simple geometries (e.g., polygonal domains) the finite element mesh exactly duplicates the actual domain.

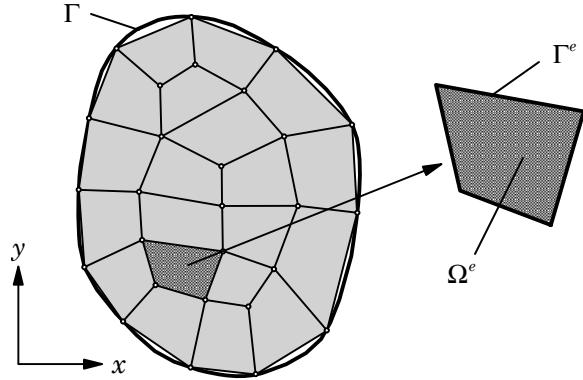


Figure 2.3.1: Finite element discretization of a domain.

For the moment let us presume that it is possible to systematically generate the approximation functions for the element Ω^e . Suppose that the dependent unknown T is approximated over a typical finite element Ω^e by the expression

$$T(x, y) \approx T^e(x, y) = \sum_{j=1}^n T_j^e \psi_j^e(x, y) \quad (2.3.1)$$

where $T^e(x, y)$ represents an approximation of $T(x, y)$ over the element Ω^e , T_j^e denote the values of the function $T^e(x, y)$ at a selected number of points, called *element nodes*, in the element Ω^e , and ψ_j^e are the approximation functions associated with the element. As we shall see shortly, the interpolation functions depend not only on the number of nodes in the element, but also on the shape of the element. The shape of the element must be such that its geometry is uniquely defined by a set of points in the element, which serve as the element nodes in the development of the approximation functions. It is possible to add a set of nodeless variables c_j to the expression in Eq. (2.3.1) in the interest of enriching the solution

$$T(x, y) \approx T^e(x, y) = \sum_{j=1}^n T_j^e \psi_j^e(x, y) + \sum_{j=1}^m c_j \phi_j(x, y) \quad (2.3.2)$$

Such considerations are beyond the scope of this chapter.

Approximation (2.3.1) is of the same form as that used in the traditional variational and weighted-residual methods except that the approximation functions used in the finite element method are functions defined over the element and the undetermined parameters T_j^e denote the values of the function being approximated at selective points (i.e., nodes) of the element. In a majority of cases, the approximation functions are polynomials, and the choice is dictated by two factors: (1) interpolation theory can be readily used to derive them, and (2) integral expressions of polynomials can be evaluated exactly using numerical integration methods. The use of function values as unknown parameters has the primary objective of imposing continuity of the dependent variable across the boundary between elements. This will be clear when assembly of elements is discussed.

2.4 Weighted-Integral Statements and Weak Forms

2.4.1 Preliminary Comments

The n nodal values T_j^e in Eq. (2.3.1) must be determined such that the approximate solution $T^e(x, y)$ satisfies the governing Eq. (2.2.1) and specified boundary conditions of the problem in *some acceptable sense*. As in the case of a variational and weighted-residual method, we seek to satisfy the boundary conditions exactly by the choice of the approximation functions and then determine the unknowns of the approximation by satisfying the governing differential equation in a weighted-integral sense. Since we are working with a typical element, we satisfy the differential equation in a weighted-integral sense over the element Ω^e . This process leads to n algebraic equations among the nodal values $(T_1^e, T_2^e, \dots, T_n^e)$. The set of algebraic equations is termed a *finite element model* of the original differential equation over the element. The type of finite element model depends on the weighted-integral form used to generate the algebraic equations. Thus, if one uses a *weak form*, whose meaning will be made clear shortly, the resulting model will be different from those obtained with a weighted-residual statement in which the weight function can be any one of several choices (see Reddy [2-6] and Reddy and Rasmussen [7] for additional details). Throughout the present study, we will be primarily concerned with the weak-form finite element models, in which the weight functions are selected to be the same as the approximation functions (the so-called Ritz or weak-form Galerkin models).

2.4.2 Weak Form Development

The weak form of a differential equation is a weighted-integral statement that is equivalent to both the governing differential equation as well as certain types of boundary conditions. We shall develop the weak form of Eqs. (2.2.1) and (2.2.5b) over the typical element Ω^e and introduce phrases like *primary and secondary variables* and *essential and natural boundary conditions*.

There are three steps in the development of a weak form of a second-order differential equation. The first step is to take all nonzero expressions in Eq. (2.2.1) to one side of the equality, multiply the resulting equation with a weight function w , and integrate the expression over the element domain Ω^e :

$$0 = \int_{\Omega^e} w \left[-\frac{\partial}{\partial x} \left(k_{xx} \frac{\partial T^e}{\partial x} \right) - \frac{\partial}{\partial y} \left(k_{yy} \frac{\partial T^e}{\partial y} \right) - Q(x, y) \right] dx dy \quad (2.4.1)$$

The expression in the square brackets of the above equation represents a residual of (or error in) the differential equation (2.2.1), because $T^e(x, y)$ is only an approximation of $T(x, y)$. Therefore, Eq. (2.4.1) is called the *weighted-residual statement* of Eq. (2.2.1). For every choice of the weight function $w(x, y)$, we obtain an algebraic equation from Eq. (2.4.1) among the nodal values T_j^e . For n independent choices of w , we obtain a set of n linearly independent algebraic equations. This set is called a *weighted-residual finite element model*. The drawback of this model for second- and higher-order differential equations is that the approximation functions ψ_j^e should be differentiable as many times as the actual solution T in the differential Eq. (2.2.1) while the weight function w is not subject to differentiation. If the equation being modeled were a first-order differential equation, we would substitute the approximation (2.2.1) and obtain the algebraic relations among the T_i^e . This would be the

true Galerkin finite element model. Since we have a second-order differential equation, the approximation functions ψ_j^e used in (2.2.1) should be twice-differentiable with respect to both x and y . This in turn requires that ψ_j^e be a quadratic or higher-order polynomial in the x and y coordinates. In the weak form, as the name suggests, this continuity requirement is reduced (or weakened) by moving part of the differentiation from T^e to the weight function w .

In the second step, we distribute the differentiation so that both T (superscript “ e ” on T is omitted in the interest of brevity) and w are required to be differentiable only once with respect to x and y . To achieve this we use integration-by-parts (or the Green–Gauss theorem) on the first two terms in Eq. (2.4.1). First we note the following identities for any differentiable functions $w(x, y)$, $F_1(x, y)$, and $F_2(x, y)$:

$$\frac{\partial}{\partial x}(wF_1) = \frac{\partial w}{\partial x}F_1 + w\frac{\partial F_1}{\partial x} \quad \text{or} \quad -w\frac{\partial F_1}{\partial x} = \frac{\partial w}{\partial x}F_1 - \frac{\partial}{\partial x}(wF_1) \quad (2.4.2a)$$

$$\frac{\partial}{\partial y}(wF_2) = \frac{\partial w}{\partial y}F_2 + w\frac{\partial F_2}{\partial y} \quad \text{or} \quad -w\frac{\partial F_2}{\partial y} = \frac{\partial w}{\partial y}F_2 - \frac{\partial}{\partial y}(wF_2) \quad (2.4.2b)$$

Next, we use the component form of the gradient (or divergence) theorem,

$$\int_{\Omega^e} \frac{\partial}{\partial x}(wF_1) dx dy = \oint_{\Gamma^e} (wF_1)n_x ds \quad (2.4.3a)$$

$$\int_{\Omega^e} \frac{\partial}{\partial y}(wF_2) dx dy = \oint_{\Gamma^e} (wF_2)n_y ds \quad (2.4.3b)$$

where n_x and n_y are the components (i.e., the direction cosines) of the unit normal vector

$$\hat{\mathbf{n}} = n_x \hat{\mathbf{e}}_x + n_y \hat{\mathbf{e}}_y = \cos \alpha \hat{\mathbf{e}}_x + \sin \alpha \hat{\mathbf{e}}_y \quad (2.4.4)$$

on the boundary Γ^e , and ds is the arc length of an infinitesimal line element along the boundary (see Figure 2.4.1). Using Eqs. (2.4.2a,b) and (2.4.3a,b), with

$$F_1 = k_{xx} \frac{\partial T}{\partial x}, \quad F_2 = k_{yy} \frac{\partial T}{\partial y}$$

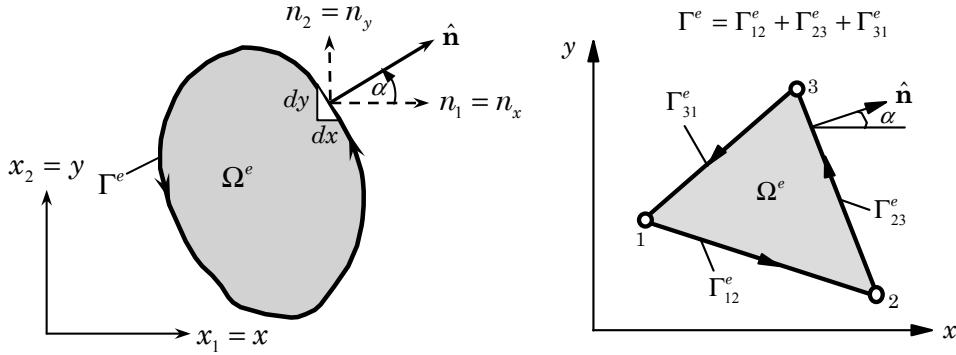


Figure 2.4.1: The unit normal vector on the boundary of a finite element.

we obtain

$$0 = \int_{\Omega^e} \left(k_{xx} \frac{\partial w}{\partial x} \frac{\partial T}{\partial x} + k_{yy} \frac{\partial w}{\partial y} \frac{\partial T}{\partial y} - wQ \right) dx dy - \oint_{\Gamma^e} w \left(k_{xx} \frac{\partial T}{\partial x} n_x + k_{yy} \frac{\partial T}{\partial y} n_y \right) ds \quad (2.4.5)$$

Since T is the dependent unknown in the differential equation, we call it the *primary variable*. Then the coefficient of the weight function w in the boundary term that appeared because of the use of integration-by-parts [see Eq. (2.4.5)]

$$q_n \equiv k_{xx} \frac{\partial T}{\partial x} n_x + k_{yy} \frac{\partial T}{\partial y} n_y \quad (2.4.6)$$

is called the *secondary variable* of the formulation. The specification of the primary variable T on a portion of the boundary is called *essential boundary condition* and the specification of a secondary variable q_n on the boundary is called the *natural boundary condition* [see Eqs. (2.2.5a,b)].

The function $q_n = q_n(s)$ denotes the projection of the conductive heat flux vector

$$\mathbf{k} \cdot \nabla T = k_{xx} \frac{\partial T}{\partial x} \hat{\mathbf{e}}_x + k_{yy} \frac{\partial T}{\partial y} \hat{\mathbf{e}}_y \quad (2.4.7)$$

along the unit normal $\hat{\mathbf{n}} = n_x \hat{\mathbf{e}}_x + n_y \hat{\mathbf{e}}_y$. By definition, q_n is positive outward from the surface as we move counterclockwise along the boundary Γ^e . The variable q_n denotes the conductive heat flux normal to the boundary of the element (into the element because of the negative sign in the Fourier heat conduction equation). In a general case, the secondary variable q_n is replaced by the sum of the conductive and convective heat fluxes normal to the boundary of the element, as discussed next.

The third and last step of the formulation is to use Eqs. (2.2.5b) and (2.2.6) to replace q_n in Eq. (2.4.5) and arrive at

$$0 = \int_{\Omega^e} \left(k_{xx} \frac{\partial w}{\partial x} \frac{\partial T}{\partial x} + k_{yy} \frac{\partial w}{\partial y} \frac{\partial T}{\partial y} - wQ \right) dx dy - \oint_{\Gamma^e} w [q_n - h_c(T - T_c)] ds \quad (2.4.8)$$

where q_n now denotes the *total* heat flux normal to the boundary of the element and h_c and T_c are convective heat transfer coefficient and ambient temperature, respectively. Note that in the absence of convective heat transfer at the boundary we set $h_c = 0$, and q_n becomes the conductive heat flux given in Eq. (2.4.6). This completes the weak form development.

It is always possible to construct the weighted-integral statement of *any* differential equation. The weak form exists for any second- and higher-order equations, because for such equations it is possible to trade differentiation from the dependent unknown to the weight function and hence include the natural boundary condition (or boundary conditions on the secondary variable) into the weighted-integral statement. These observations hold for linear as well as for nonlinear problems. The weak form in Eq. (2.4.8) is used to develop the finite element model of Eq. (2.2.1). By substituting for T from Eq. (2.3.1) and setting $w = \psi_1, w = \psi_2, \dots, w = \psi_n$ (for

the weak-form Galerkin model), we obtain n algebraic equations from Eq. (2.4.8), as shown in the next section.

It is useful to cast the weak form in Eq. (2.4.8) in terms of the bilinear and linear forms as

$$B(w, T) = \ell(w) \quad (2.4.9)$$

where the *bilinear form* $B(\cdot, \cdot)$ and *linear form* $\ell(\cdot)$ are defined by

$$B(w, T) = \int_{\Omega^e} \left(k_{xx} \frac{\partial w}{\partial x} \frac{\partial T}{\partial x} + k_{yy} \frac{\partial w}{\partial y} \frac{\partial T}{\partial y} \right) dx dy + \oint_{\Gamma^e} h_c w T ds \quad (2.4.10a)$$

$$\ell(w) = \int_{\Omega^e} w Q dx dy + \oint_{\Gamma^e} w q_n ds - \oint_{\Gamma^e} h_c T_c w ds \quad (2.4.10b)$$

The word “bilinear” indicates that $B(w, T)$ is linear in *both* w and T . The statement in Eq. (2.4.8) is called the *variational problem* associated with Eqs. (2.2.1) and (2.2.5b). As we shall see shortly, $B(\cdot, \cdot)$ yields the coefficient matrix \mathbf{K} , and $\ell(\cdot)$ becomes the right-hand side \mathbf{F} of the finite element equations, $\mathbf{KT} = \mathbf{F}$.

2.5 Finite Element Model

The weak form in Eq. (2.4.8) requires that the approximation chosen for T should be at least linear in both x and y so that there are no terms in (2.4.8) that become identically zero. In addition, it requires that the primary variable T be made continuous across the elements. Suppose that T is approximated over a typical finite element Ω^e by the expression [repetition of Eq. (2.3.1)]

$$T(x, y) \approx T^e(x, y) = \sum_{j=1}^n T_j^e \psi_j^e(x, y) \quad (2.5.1)$$

where T_j^e is the value of $T^e(x, y)$ at the j th node with coordinates (x_j, y_j) of the element, and ψ_j^e are the approximation functions, with the *interpolation property* (see Reddy [4])

$$\psi_i^e(x_j, y_j) = \delta_{ij} \quad (2.5.2)$$

Note that the interpolation property implies that $T^e(x_i, y_i) = T_i^e$. Functions that satisfy the interpolation property (2.5.2) are known as the *Lagrange interpolation functions*. The specific form of ψ_i^e will be presented for triangular and rectangular elements in Section 2.6. Additional discussion of the finite element interpolation functions is presented in Section 2.10 and Chapter 3.

Substituting the finite element approximation (2.5.1) for T into the weak form (2.4.8) or (2.4.9), we obtain

$$\begin{aligned} 0 &= \int_{\Omega^e} \left[\frac{\partial w}{\partial x} \left(k_{xx} \sum_{j=1}^n T_j^e \frac{\partial \psi_j^e}{\partial x} \right) + \frac{\partial w}{\partial y} \left(k_{yy} \sum_{j=1}^n T_j^e \frac{\partial \psi_j^e}{\partial y} \right) - w Q \right] dx dy \\ &\quad + \oint_{\Gamma^e} h_c w \left(\sum_{j=1}^n T_j^e \psi_j^e \right) ds - \oint_{\Gamma^e} w q_n ds - \oint_{\Gamma^e} h_c T_c w ds \\ &= \sum_{j=1}^n \left\{ \int_{\Omega^e} \left[\frac{\partial w}{\partial x} \left(k_{xx} \frac{\partial \psi_j^e}{\partial x} \right) + \frac{\partial w}{\partial y} \left(k_{yy} \frac{\partial \psi_j^e}{\partial y} \right) \right] dx dy + \oint_{\Gamma^e} h_c w \psi_j^e ds \right\} T_j^e \\ &\quad - \int_{\Omega^e} w Q dx dy - \oint_{\Gamma^e} w q_n ds - \oint_{\Gamma^e} h_c T_c w ds \end{aligned} \quad (2.5.3)$$

This equation must hold for any weight function w . Since we need n independent algebraic equations to solve for the n unknowns, $T_1^e, T_2^e, \dots, T_n^e$, we choose n independent functions for w , one at a time. One choice is to use the approximation functions ψ_i^e ($i = 1, 2, \dots, n$) for the weight function, $w = \psi_1^e, w = \psi_2^e, \dots, w = \psi_n^e$, as in the Galerkin method. This particular choice of weight functions is a natural one when the weight function is viewed as a virtual variation of the dependent unknown (i.e., $w = \delta T^e = \sum_{i=1}^n \delta T_i^e \psi_i^e$ as in the Ritz method [3]). Note that the method proposed by Galerkin did not involve integration by parts (i.e., weak form was not used) but was based on Eq. (2.4.1). Thus, the phrase “weak-form Galerkin” formulation is different from “Galerkin” formulation.

In the weak-form Galerkin model, for each choice of w we obtain an algebraic relation among all $(T_1^e, T_2^e, \dots, T_n^e)$. We label the algebraic equation resulting from substitution of ψ_1^e for w into Eq. (2.5.3) as the first algebraic equation. Thus, the i th algebraic equation is obtained by substituting $w = \psi_i^e$ into Eq. (2.5.3):

$$0 = \sum_{j=1}^n \left\{ \int_{\Omega^e} \left[\frac{\partial \psi_i^e}{\partial x} \left(k_{xx} \frac{\partial \psi_j^e}{\partial x} \right) + \frac{\partial \psi_i^e}{\partial y} \left(k_{yy} \frac{\partial \psi_j^e}{\partial y} \right) \right] dx dy + \oint_{\Gamma^e} h_c \psi_i^e \psi_j^e ds \right\} T_j^e \\ - \int_{\Omega^e} \psi_i^e Q dx dy - \oint_{\Gamma^e} \psi_i^e q_n ds - \oint_{\Gamma^e} h_c T_c \psi_i^e ds$$

or

$$\sum_{j=1}^n K_{ij}^e T_j^e = Q_i^e + q_i^e \quad (2.5.4)$$

where the coefficients K_{ij}^e, Q_i^e , and q_i^e are defined by

$$K_{ij}^e = \int_{\Omega^e} \left(k_{xx} \frac{\partial \psi_i^e}{\partial x} \frac{\partial \psi_j^e}{\partial x} + k_{yy} \frac{\partial \psi_i^e}{\partial y} \frac{\partial \psi_j^e}{\partial y} \right) dx dy + \oint_{\Gamma^e} h_c \psi_i^e \psi_j^e ds \quad (2.5.5a)$$

$$Q_i^e = \int_{\Omega^e} Q \psi_i^e dx dy \quad (2.5.5b)$$

$$q_i^e = \oint_{\Gamma^e} q_n \psi_i^e ds + \oint_{\Gamma^e} h_c T_c \psi_i^e ds \quad (2.5.5c)$$

In matrix notation, Eq. (2.5.4) takes the form

$$[K^e]\{T^e\} = \{Q^e\} + \{q^e\} \quad \text{or} \quad \mathbf{K}^e \mathbf{T}^e = \mathbf{Q}^e + \mathbf{q}^e \quad (2.5.6)$$

The matrix \mathbf{K}^e is called the *coefficient matrix*, or conductivity matrix in the present context. We note that $K_{ij}^e = K_{ji}^e$ (i.e., \mathbf{K}^e is symmetric). The symmetry of the coefficient matrix is due to the symmetry of the bilinear form in Eq. (2.4.10a), which in turn is due to the weak form development. Equation (2.5.6) is called the *finite element model* of Eq. (2.2.1) or its weak form in Eq. (2.4.8). *This completes the finite element model development.*

Before we discuss assembly of element equations, it is informative to determine the interpolation functions ψ_i^e for linear two-dimensional elements. In the next section, we discuss the derivation of linear interpolation functions for triangles and rectangles, and numerically evaluate the coefficient matrices in Eq. (2.5.5a,b) for constant material properties.

2.6 Interpolation Functions

2.6.1 Properties of Approximation Functions

The finite element approximation $T^e(x, y)$ of $T(x, y)$ over an element Ω^e must satisfy the following conditions in order for the approximate solution to converge to the true solution:

1. $T^e(x, y)$ must be continuous as required in the weak form of the problem (i.e., all terms in the weak form are represented as nonzero values) and $T^e(x, y)$ must be single-valued across the elements.
2. The polynomials used to represent $T^e(x, y)$ must be complete (i.e., all terms, beginning with a constant term up to the highest-order used in the polynomial should be included in the expression of $T^e(x, y)$), and linearly independent.
3. All terms appearing in the polynomials should contain equal powers of both x and y (equipresence).

The number of linearly independent terms in the representation of T^e dictates the shape and number of degrees of freedom of the element. Because of the requirements (1) and (2), not all geometric shapes qualify as finite elements. It turns out that only triangles and quadrilaterals with appropriate number of nodes qualify as elements in two dimensions. In this section we review the interpolation functions of linear triangular and rectangular elements.

2.6.2 Linear Triangular Element

An examination of the weak form (2.4.8) and the finite element matrices in Eq. (2.5.5a) shows that the ψ_i^e should be at least linear functions of both x and y . The polynomial

$$T^e(x, y) = c_1^e + c_2^e x + c_3^e y \quad (2.6.1)$$

is the lowest-order polynomial that meets the requirements. It contains three linearly independent terms, and it is linear in both x and y . The polynomial is complete because the lower-order term, namely, the constant term, is included. To write the three constants (c_1^e, c_2^e, c_3^e) in terms of the nodal values of T^e , we must identify three points or nodes in the element Ω^e . The three nodes must be such that they uniquely define the geometry of the element and allow the imposition of inter-element continuity of the variable $T^e(x, y)$. Obviously, the geometric shape defined by three points in a two-dimensional domain is a triangle. Thus the polynomial in Eq. (2.6.1) is associated with a triangular element and the three nodes are identified as the vertices of the triangle.

The linear interpolation functions of a three-node triangle [see Figure 2.6.1(a)] are (see Reddy [4, pp. 417–421])

$$\psi_i^e(x, y) = \frac{1}{2A_e}(\alpha_i^e + \beta_i^e x + \gamma_i^e y), \quad (i = 1, 2, 3) \quad (2.6.2)$$

where A_e is the area of the triangle, and α_i^e , β_i^e , and γ_i^e are geometric constants known in terms of the nodal coordinates (x_i, y_i) :

$$\alpha_i^e = x_j y_k - x_k y_j; \quad \beta_i^e = y_j - y_k; \quad \gamma_i^e = -(x_j - x_k) \quad (2.6.3)$$

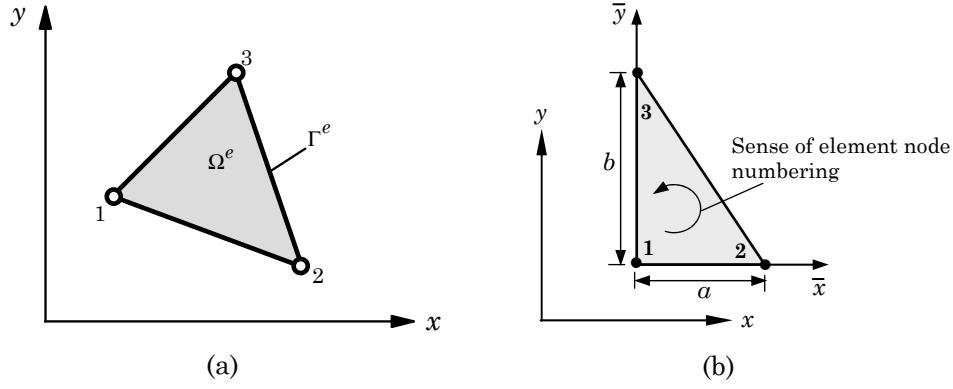


Figure 2.6.1: Linear triangular finite elements.

Here the subscripts are such that $i \neq j \neq k$, and i, j , and k permute in a natural order. Note that (x, y) are the coordinates used in the governing equation (2.2.1) over the domain Ω , and are called *global coordinates*. Since the element interpolation functions are restricted to an element, it is possible to express them in a *local coordinate system*. Such local coordinate systems are useful in the numerical evaluation of the integral expressions in Eqs. (2.5.5a,b). We will return to these issues in Section 2.11.

The interpolation functions $\psi_i^e (i = 1, 2, \dots, n)$ constructed such that Eq. (2.6.1) interpolates only the function at the nodes (and not its derivatives) are called the *Lagrange interpolation functions*. They satisfy the following properties:

$$(1) \quad \psi_i^e(x_j, y_j) = \delta_{ij}, \quad (i, j = 1, 2, 3); \quad (2) \quad \sum_{i=1}^3 \psi_i^e(x, y) = 1 \quad (2.6.4)$$

While the first one is called the interpolation property, the second one is known as the *partition of unity*. Note that use of the linear interpolation functions ψ_i^e of a triangle will result in the approximation of the curved surface $T(x, y)$ by a planar function $T^e = \sum_{i=1}^3 T_i^e \psi_i^e$.

The integrals in the definition of K_{ij}^e and Q_i^e can be evaluated for given data: k_{xx}, k_{yy}, Q and $h_c = 0$. For example, for element-wise constant values of the data, i.e., $k_{xx} = k_{xx}^e$, $k_{yy} = k_{yy}^e$, and $Q = Q^e$, we have (see Reddy [4, pp. 426–429]) the following results:

$$K_{ij}^e = \frac{1}{4A_e} (k_{xx}^e \beta_i^e \beta_j^e + k_{yy}^e \gamma_i^e \gamma_j^e); \quad Q_i^e = \frac{Q_e A_e}{3} \quad (2.6.5)$$

where A_e is the area of the triangular element, and β_i^e and γ_i^e are known in terms of the global nodal coordinates of the element nodes, as given in Eq. (2.6.3). For a right-angled triangular element with base a and height b [see Figure 2.6.1(b)], and node 1 at the right angle (nodes are numbered counterclockwise), $[K^e]$ takes the form (see Reddy [4, p. 428])

$$[K^e] = \frac{k_{xx}^e}{2} \begin{bmatrix} \alpha & -\alpha & 0 \\ -\alpha & \alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} + \frac{k_{yy}^e}{2} \begin{bmatrix} \beta & 0 & -\beta \\ 0 & 0 & 0 \\ -\beta & 0 & \beta \end{bmatrix} \quad (2.6.6)$$

where $\alpha = b/a$ and $\beta = a/b$. Of course, for cases in which the conductivities are functions of (x, y) , numerical integration can be used to evaluate the coefficients (see Section 2.12).

2.6.3 Linear Rectangular Element

The next complete polynomial to that in Eq. (2.6.1) is

$$T^e(x, y) = c_1^e + c_2^e x + c_3^e y + c_4^e xy \quad (2.6.7)$$

which contains four linearly independent terms and it is linear in x and y , with a bilinear term in x and y . This polynomial requires an element with four nodes. It is a rectangle with nodes at the four corners of the rectangle or a triangle with three nodes at vertices and the fourth node at the centroid of the triangle. The four-node triangular element does not meet the requirement that the function be single-valued between elements, and hence does not qualify as a finite element.

For a linear rectangular element (see Figure 2.6.2), we have

$$T(\bar{x}, \bar{y}) = \sum_{i=1}^4 T_i^e \psi_i^e(\bar{x}, \bar{y}) \quad (2.6.8)$$

where (see Reddy [4, pp. 421–425])

$$\begin{aligned} \psi_1^e &= (1 - \frac{\bar{x}}{a})(1 - \frac{\bar{y}}{b}), & \psi_2^e &= \frac{\bar{x}}{a}(1 - \frac{\bar{y}}{b}) \\ \psi_3^e &= \frac{\bar{x}}{a} \frac{\bar{y}}{b}, & \psi_4^e &= (1 - \frac{\bar{x}}{a}) \frac{\bar{y}}{b} \end{aligned} \quad (2.6.9)$$

and (\bar{x}, \bar{y}) denote the local coordinates with the origin located at node 1 of the element, and (a, b) denote the horizontal and vertical dimensions of the rectangle (see Figure 2.6.2).

The integrals in the definition of K_{ij}^e and Q_i^e can be easily evaluated over a rectangular element of sides a and b . For example, for element-wise constant values of the data, i.e., $k_{xx} = k_{xx}^e$, $k_{yy} = k_{yy}^e$, $Q = Q^e$, and $h_c = 0$, we have (see Reddy [4, pp. 429–431]) the following results:

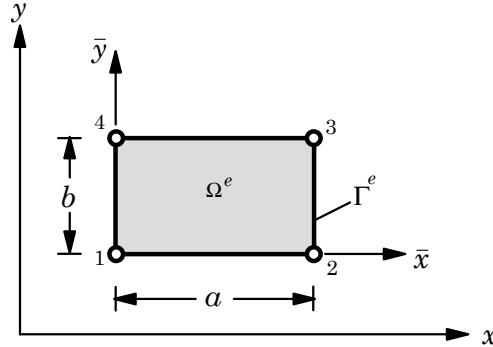


Figure 2.6.2: The linear rectangular finite element.

$$[K^e] = k_{xx}^e [S^{11}] + k_{yy}^e [S^{22}], \quad Q_i^e = \frac{Q_e ab}{4} \quad (2.6.10a)$$

where

$$[S^{11}] = \frac{1}{6} \begin{bmatrix} 2\alpha & -2\alpha & -\alpha & \alpha \\ -2\alpha & 2\alpha & \alpha & -\alpha \\ -\alpha & \alpha & 2\alpha & -2\alpha \\ \alpha & -\alpha & -2\alpha & 2\alpha \end{bmatrix} \quad (2.6.10b)$$

$$[S^{22}] = \frac{1}{6} \begin{bmatrix} 2\beta & \beta & -\beta & -2\beta \\ \beta & 2\beta & -2\beta & -\beta \\ -\beta & -2\beta & 2\beta & \beta \\ -2\beta & -\beta & \beta & 2\beta \end{bmatrix} \quad (2.6.10c)$$

and $\alpha = b/a$ and $\beta = a/b$. Again, for cases in which the conductivities are functions of (x, y) , numerical integration is used to evaluate the coefficients, as discussed in Section 2.11. When the element is nonrectangular, i.e., a quadrilateral, we use coordinate transformations to represent the integrals over a square geometry (see Sections 2.12 and 3.4) and then use numerical integration to evaluate them.

2.6.4 Evaluation of Boundary Integrals

The evaluation of boundary integrals of the type

$$q_i^e = \oint_{\Gamma^e} q_n^e \psi_i^e(s) \, ds \quad (2.6.11)$$

where q_n^e is a known function of the distance s along the boundary Γ^e , involves evaluation of line integrals. This is because the boundary of a two-dimensional element is a collection of lines and the interpolation functions along these lines become one-dimensional functions. It is necessary to compute such integrals only when Γ^e , or a portion of it, coincides with the boundary Γ_q of the problem on which the flux is specified. On portions of Γ^e that are in the interior of the domain Ω , q_n^e on side (i, j) of element Ω^e cancels with q_n^f on side (p, q) of element Ω^f when sides (i, j) of element Ω^e and (p, q) of element Ω^f are the same (i.e., at the interface of elements Ω^e and Ω^f). This can be viewed as the balance of the internal flux. When Γ^e falls on the boundary Γ_T of the domain Ω , q_n^e is not known there and can be determined in the post-computation. Note that the primary variable T is specified on Γ_T . For additional details, see Reddy [4, pp. 431–436].

2.7 Assembly of Elements

The assembly of finite elements to obtain the equations of the entire domain is based on the following two rules:

1. Continuity of the primary variable (i.e., temperature)
2. Balance of secondary variables (i.e., heat flux)

We illustrate the assembly procedure by considering a finite element mesh consisting of a triangular element and a quadrilateral element (see Figure 2.7.1).

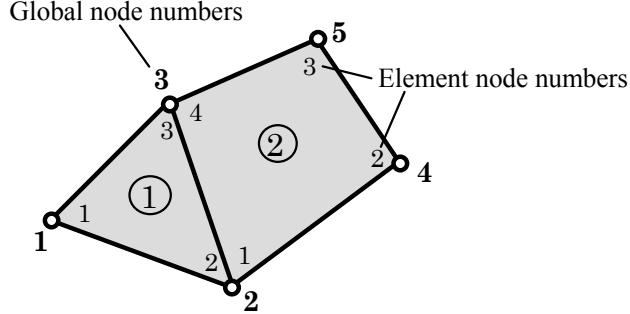


Figure 2.7.1: Global-local correspondence of nodes for assembly of elements.

Let K_{ij}^1 ($i, j = 1, 2, 3$) denote the coefficient matrix corresponding to the triangular element, and let K_{ij}^2 ($i, j = 1, 2, 3, 4$) denote the coefficient matrix corresponding to the quadrilateral element. The nodes of the finite element mesh are called *global nodes*. From the mesh shown in Figure 2.7.1, it is clear that the following correspondence between the global and element nodes exists: nodes 1, 2, and 3 of element 1 correspond to global nodes 1, 2, and 3, respectively. Nodes 1, 2, 3, and 4 of element 2 correspond to global nodes 2, 4, 5, and 3, respectively. Hence, the correspondence between the local and global nodal values of temperature is

$$T_1^1 = T_1, \quad T_2^1 = T_1^2 = T_2, \quad T_3^1 = T_4^2 = T_3, \quad T_2^2 = T_4, \quad T_3^2 = T_5 \quad (2.7.1)$$

which amounts to imposing the continuity of the primary variables at the nodes common to elements 1 and 2. Note that the continuity of the primary variables at the inter-element nodes guarantees the continuity of the primary variable along the entire inter-element boundary.

Next, we consider the balance of secondary variables. At the interface between the two elements, the flux from the two elements should be equal in magnitude and opposite in sign. For the two elements in Figure 2.7.1, the interface is along the side connecting global nodes 2 and 3. Hence, the internal flux q_n^1 on side 2–3 of element 1 should balance the flux q_n^2 on side 4–1 of element 2 (recall the sign convention on q_n^e):

$$(q_n^1)_{2-3} = (q_n^2)_{4-1} \text{ or } (q_n^1)_{2-3} = (-q_n^2)_{1-4} \quad (2.7.2)$$

In the finite element method, the above relation is imposed in a weighted-integral sense:

$$\int_{h_{23}^1} q_n^1 \psi_2^1 \, ds = - \int_{h_{14}^2} q_n^2 \psi_1^2 \, ds \quad (2.7.3a)$$

$$\int_{h_{23}^1} q_n^1 \psi_3^1 \, ds = - \int_{h_{14}^2} q_n^2 \psi_4^2 \, ds \quad (2.7.3b)$$

where h_{pq}^e denotes length of the side connecting node p to node q of element Ω^e .

Now we are ready to assemble the element equations to obtain the equations for the two-element mesh. The element equations of the two elements are written separately first. For the triangular element, the element equations are of the form

$$\begin{aligned} K_{11}^1 T_1^1 + K_{12}^1 T_2^1 + K_{13}^1 T_3^1 &= Q_1^1 + q_1^1 \\ K_{21}^1 T_1^1 + K_{22}^1 T_2^1 + K_{23}^1 T_3^1 &= Q_2^1 + q_2^1 \\ K_{31}^1 T_1^1 + K_{32}^1 T_2^1 + K_{33}^1 T_3^1 &= Q_3^1 + q_3^1 \end{aligned} \quad (2.7.4a)$$

For the rectangular element the element equations are given by

$$\begin{aligned} K_{11}^2 T_1^2 + K_{12}^2 T_2^2 + K_{13}^2 T_3^2 + K_{14}^2 T_4^2 &= Q_1^2 + q_1^2 \\ K_{21}^2 T_1^2 + K_{22}^2 T_2^2 + K_{23}^2 T_3^2 + K_{24}^2 T_4^2 &= Q_2^2 + q_2^2 \\ K_{31}^2 T_1^2 + K_{32}^2 T_2^2 + K_{33}^2 T_3^2 + K_{34}^2 T_4^2 &= Q_3^2 + q_3^2 \\ K_{41}^2 T_1^2 + K_{42}^2 T_2^2 + K_{43}^2 T_3^2 + K_{44}^2 T_4^2 &= Q_4^2 + q_4^2 \end{aligned} \quad (2.7.4b)$$

In order to impose the balance of secondary variables in Eq. (2.7.3), it is required that we add the second equation of element 1 to the first equation of element 2, and also add the third equation of element 1 to the fourth equation of element 2:

$$(K_{21}^1 T_1^1 + K_{22}^1 T_2^1 + K_{23}^1 T_3^1) + (K_{11}^2 T_1^2 + K_{12}^2 T_2^2 + K_{13}^2 T_3^2 + K_{14}^2 T_4^2) = (Q_2^1 + q_2^1) + (Q_1^2 + q_1^2) \quad (2.7.5a)$$

$$(K_{31}^1 T_1^1 + K_{32}^1 T_2^1 + K_{33}^1 T_3^1) + (K_{41}^2 T_1^2 + K_{42}^2 T_2^2 + K_{43}^2 T_3^2 + K_{44}^2 T_4^2) = (Q_3^1 + q_3^1) + (Q_4^2 + q_4^2) \quad (2.7.5b)$$

Using the local-global nodal variable correspondence in Eq. (2.7.1), we can rewrite the above equations as

$$\begin{aligned} K_{21}^1 T_1 + (K_{22}^1 + K_{11}^2) T_2 + (K_{23}^1 + K_{14}^2) T_3 + K_{12}^2 T_4 + K_{13}^2 T_5 \\ = Q_2^1 + Q_1^2 + (q_2^1 + q_1^2) \end{aligned} \quad (2.7.6a)$$

$$\begin{aligned} K_{31}^1 T_1 + (K_{32}^1 + K_{41}^2) T_2 + (K_{33}^1 + K_{44}^2) T_3 + K_{42}^2 T_4 + K_{43}^2 T_5 \\ = Q_3^1 + Q_4^2 + (q_3^1 + q_4^2) \end{aligned} \quad (2.7.6b)$$

Now we can impose the conditions in Eq. (2.7.3) by setting appropriate portions of the expressions in parenthesis on the right-hand side of the above equations to zero (or a specified nonzero value). In general, when several elements are connected, the assembly of the elements is carried out by putting element coefficients K_{ij}^e , Q_i^e , and q_i^e into proper locations of the global coefficient matrix and right-hand column vectors. This is done by means of the connectivity relations, i.e., correspondence of the local node number to the global node number. The assembly procedure described here can be used to assemble elements of any shape and type. The procedure can be implemented in a computer with the help of the local-global nodal correspondence.

This completes the first three steps in the finite element modeling of the model Eq. (2.2.1). The remaining three steps of the analysis, namely, the imposition of boundary conditions, solution of equations, and postprocessing of the solution will not be discussed here, but will be reserved for subsequent chapters. It may

be recalled that the derivatives of $T^e(x, y)$ will not be continuous at inter-element boundaries because continuity of the derivatives is not imposed in the model. The weak form of the equation suggests that the primary variable is T , and therefore it should be made continuous across the inter-element boundaries. If additional variables, such as the derivatives of temperature, are carried as nodal variables in the interest of making them continuous across inter-element boundaries, the degree of interpolation (or order of the element) increases. In addition, the continuity of the derivatives, which are not identified as the primary variables in the weak formulation of the problem, may violate the physical principles of the problem.

2.8 Time-Dependent Problems

2.8.1 Introduction

In this section we present the finite element model of time-dependent heat transfer problems in two dimensions. Consider the equation governing transient heat transfer in two dimensions,

$$\rho C \frac{\partial T}{\partial t} - \left[\frac{\partial}{\partial x} \left(k_{xx} \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left(k_{yy} \frac{\partial T}{\partial y} \right) \right] = Q \quad \text{in } \Omega \quad (2.8.1)$$

We wish to analyze Eq. (2.8.1) under appropriate boundary and initial conditions. The boundary conditions for this equation are given by Eqs. (2.2.5), and the initial condition on the temperature is given by

$$T(\mathbf{x}, 0) = T_0(\mathbf{x}) \quad (2.8.2)$$

The numerical solution of the initial-boundary value problem described by Eq. (2.8.1) involves two stages of approximation. The first stage, called *spatial discretization*, involves the development of the weak form of the equation over an element and the spatial approximation of the dependent variable (i.e., temperature) T of the problem. The three-step procedure presented in Section 2.4 for the development of the weak form is still applicable to time-dependent problems. This stage results in a set of ordinary differential equations in time among the nodal values T_j^e of the dependent variable. The second stage consists of a time approximation, called *temporal approximation*, of the ordinary differential equations (i.e., numerical integration of the equations) by finite difference schemes. This stage leads to a set of algebraic equations involving the nodal values T_j^e at time t_{n+1} [= $(n + 1)\Delta t$, where n is an integer and Δt is the time increment] in terms of known values from the previous time step. The finite element model of Eq. (2.8.1) using the *space-time finite elements* can also be developed (see Reddy [4], pp. 490–492, and [8–12]). In this procedure, time is treated as another spatial coordinate. Consequently, one should know the solution at the initial as well as final times (i.e., at the time boundaries). Here we discuss only the two-stage approximation. The semidiscretization is discussed first.

2.8.2 Semidiscretization

We assume, as usual, that the conduction heat transfer region Ω is discretized into an appropriate collection of finite elements (see Figure 2.3.1). The weak form of

Eq. (2.8.1) over an element Ω^e is obtained by the standard procedure: multiply Eq. (2.8.1) with the weight function $w(x, y)$ and integrate over the element, integrate-by-parts (spatially) those terms which involve higher-order derivatives using the gradient or divergence theorem, and replace the coefficient of the weight function in the boundary integral with the secondary variable [i.e., use Eq. (2.2.5b)]. We obtain

$$0 = \int_{\Omega^e} \left[w \left(\rho C \frac{\partial T}{\partial t} - Q_s \right) + k_{xx} \frac{\partial w}{\partial x} \frac{\partial T}{\partial x} + k_{yy} \frac{\partial w}{\partial y} \frac{\partial T}{\partial y} \right] dx dy - \oint_{\Gamma^e} (q_n - q_c) w \, ds \quad (2.8.3)$$

Note that the procedure to obtain the weak form for time-dependent problems is not much different from that used for steady-state problems in Section 2.4. The difference is that all terms of the equations may be functions of time. Also, no integration by parts with respect to time is used, and the weight function w is not a function of time.

The semidiscrete finite element model is obtained from Eq. (2.8.3) by substituting a finite element approximation for the dependent variable, T . In selecting the approximation for T , we assume that the time dependence can be separated from the spatial variation,

$$T(\mathbf{x}, t) \simeq \sum_{j=1}^{n_e} T_j^e(t) \psi_j^e(\mathbf{x}) \quad (2.8.4)$$

The i th differential equation (in time) of the finite element model is obtained by substituting $w = \psi_i^e(\mathbf{x})$ and replacing T by the expression in Eq. (2.8.4):

$$0 = \sum_{j=1}^{n_e} \left(M_{ij}^e \frac{dT_j^e}{dt} + K_{ij}^e T_j^e \right) - Q_i^e - q_i^e \quad (2.8.5)$$

Equation (2.8.5) can be expressed in matrix form as

$$\mathbf{M}^e \dot{\mathbf{T}}^e + \mathbf{K}^e \mathbf{T}^e = \mathbf{Q}^e + \mathbf{q}^e \quad (2.8.6)$$

where a superposed dot on \mathbf{T} denotes a derivative with time ($\dot{\mathbf{T}} = \partial \mathbf{T} / \partial t$), and

$$\begin{aligned} M_{ij}^e &= \int_{\Omega^e} \rho C \psi_i \psi_j \, dx dy \\ K_{ij}^e &= \int_{\Omega^e} \left(k_{xx} \frac{\partial \psi_i}{\partial x} \frac{\partial \psi_j}{\partial x} + k_{yy} \frac{\partial \psi_i}{\partial y} \frac{\partial \psi_j}{\partial y} \right) \, dx dy \\ Q_i^e &= \int_{\Omega^e} \psi_i Q(x, y, t) \, dx dy \\ q_i^e &= \oint_{\Gamma^e} \psi_i (q_n - q_c) \, ds \end{aligned} \quad (2.8.7)$$

The element label “ e ” on ψ_i is omitted for brevity. It should be noted that the capacitance (or mass) matrix \mathbf{M}^e in Eq. (2.8.6) is not a diagonal matrix. It is called a *consistent (mass) matrix* because it is defined consistent with the weak formulation in Eq. (2.8.3).

2.8.3 Temporal Approximation

The ordinary differential Eq. (2.8.6) should be integrated with respect to time to obtain the transient response. Since it is not possible, in general, to integrate these equations analytically, they are further approximated in time to obtain a set of algebraic equations in terms of the nodal temperatures (i.e., a fully discretized model is obtained). In principle, it is possible to use any of the standard methods for the solution of ordinary differential equations. However, practical considerations, such as the computational cost, dictate that only the simpler time integration (or approximation) methods be considered.

The most commonly used time integration methods for Eq. (2.8.6) are all part of a one-parameter family, called the α -family of approximation

$$\mathbf{T}_{n+1} = \mathbf{T}_n + \Delta t[(1 - \alpha)\dot{\mathbf{T}}_n + \alpha\dot{\mathbf{T}}_{n+1}] , \quad 0 \leq \alpha \leq 1 \quad (2.8.8)$$

Using Eq. (2.8.8) in Eq. (2.8.6), we can transform the ordinary differential equations into a set of algebraic equations at time t_{n+1} :

$$\hat{\mathbf{K}}_{n+1}\mathbf{T}_{n+1} = \hat{\mathbf{F}}_{n,n+1} \quad (2.8.9)$$

where

$$\hat{\mathbf{K}}_{n+1} = \mathbf{M} + a_1\mathbf{K}_{n+1} \quad (2.8.10a)$$

$$\hat{\mathbf{F}}_{n,n+1} = \Delta t[\alpha\mathbf{F}_{n+1} + (1 - \alpha)\mathbf{F}_n] + \bar{\mathbf{K}}_n\mathbf{T}_n \quad (2.8.10b)$$

$$\bar{\mathbf{K}}_n = (\mathbf{M} - a_2\mathbf{K}_n), \quad a_1 = \alpha\Delta t, \quad a_2 = (1 - \alpha)\Delta t \quad (2.8.10c)$$

Equation (2.8.9), after assembly and imposition of boundary conditions, is solved at each time step for the nodal values T_j at time $t_n = (n + 1)\Delta t$. At time $t = 0$ (i.e., $n = 0$), the right-hand side is computed using the initial values \mathbf{T}_0 ; the vector \mathbf{F} , which is the sum of the source vector \mathbf{Q}^e and internal flux vector \mathbf{q}^e , is always known, for both times t_n and t_{n+1} , at all nodes at which the solution is unknown (because $Q(x, y, t)$ is a known function of time and the sum of q_j^e at these nodes is zero).

For different values of the parameter α , we obtain several well-known time approximation schemes:

- $\alpha = 0$, the forward difference scheme (conditionally stable)
- $\alpha = 0.5$, the Crank–Nicolson scheme (unconditionally stable)
- $\alpha = \frac{2}{3}$, the Galerkin scheme (unconditionally stable) (2.8.11)
- $\alpha = 1$, the backward difference scheme (unconditionally stable)

For $\alpha \geq 0.5$, the scheme is stable, and for $\alpha < 0.5$ the scheme is stable only if the time step meets certain restrictions (i.e., *conditionally stable schemes*) (see Reddy [4]). For the forward difference scheme the stability requirement is

$$\Delta t < \Delta t_{cr} = \frac{2}{(1 - 2\alpha)\lambda_{\max}}, \quad \alpha < \frac{1}{2} \quad (2.8.12)$$

where λ_{\max} is the largest eigenvalue of the eigenvalue problem associated with the matrix equation (2.8.6):

$$|\hat{\mathbf{K}} - \lambda \mathbf{I}| = 0 \quad (2.8.13)$$

This completes the development of the finite element model of a transient heat transfer problem in two dimensions. Further details on time-dependence will be covered in Chapter 3.

2.9 Axisymmetric Problems

In studying heat transfer problems involving cylindrical geometries, it is convenient to use the cylindrical coordinate system (r, θ, z) (see Figure 2.9.1) to formulate the problem. If the geometry, data, and boundary conditions of the problem are independent of the angular coordinate θ , the problem solution will also be independent of θ . Consequently, a three-dimensional problem is reduced to a two-dimensional one in (r, z) coordinates. Here we present the weak form for an axisymmetric problem.

Consider the partial differential equation governing heat transfer in an axisymmetric geometry

$$-\frac{1}{r} \frac{\partial}{\partial r} \left(r k_{rr} \frac{\partial T}{\partial r} \right) - \frac{\partial}{\partial z} \left(k_{zz} \frac{\partial T}{\partial z} \right) = Q(r, z) \quad (2.9.1)$$

where (k_{rr}, k_{zz}) and Q are the conductivities and internal heat generation per unit volume, respectively. In developing the weak form, we integrate over the elemental volume of the axisymmetric geometry: $rdrd\theta dz$. Since the solution is independent of the θ coordinate, the integration with respect to θ yields a multiplicative constant, 2π .

Following the three-step procedure, we write the weak form of Eq. (2.9.1):

$$0 = 2\pi \int_{\Omega^e} w \left(-\frac{1}{r} \frac{\partial}{\partial r} (r k_{rr} \frac{\partial T}{\partial r}) - \frac{\partial}{\partial z} (k_{zz} \frac{\partial T}{\partial z}) - Q \right) r dr dz$$

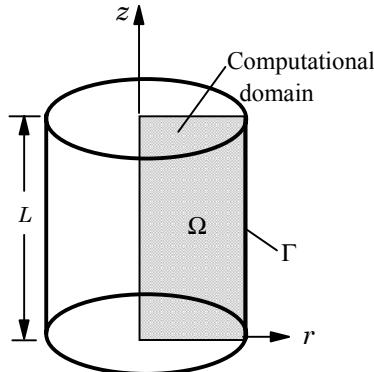


Figure 2.9.1: Domain of an axisymmetric problem.

Integrating the highest-order terms by parts, we obtain

$$0 = 2\pi \int_{\Omega^e} \left(\frac{\partial w}{\partial r} \cdot rk_{rr} \frac{\partial T}{\partial r} + \frac{\partial w}{\partial z} \cdot rk_{zz} \frac{\partial T}{\partial z} - w \cdot rQ \right) dr dz - 2\pi \oint_{\Gamma^e} w \left(rk_{rr} \frac{\partial T}{\partial r} n_r + rk_{zz} \frac{\partial T}{\partial z} n_z \right) ds$$

or

$$0 = 2\pi \int_{\Omega^e} \left(k_{rr} \frac{\partial w}{\partial r} \frac{\partial T}{\partial r} + k_{zz} \frac{\partial w}{\partial z} \frac{\partial T}{\partial z} - wQ \right) r dr dz - \oint_{\Gamma^e} w q_n ds \quad (2.9.2)$$

where w is the weight function and q_n is the normal flux,

$$q_n = 2\pi r \left(k_{rr} \frac{\partial T}{\partial r} n_r + k_{zz} \frac{\partial T}{\partial z} n_z \right) \quad (2.9.3)$$

Note that the weak form (2.9.2) does not differ significantly from that developed for Eq. (2.2.1). The only difference is the presence of r in the integrand. Consequently, Eq. (2.9.2) can be obtained as a special case of Eq. (2.4.8) for $k_{xx} = 2\pi k_{rr} \cdot x$, $k_{yy} = 2\pi k_{zz} \cdot x$, and Q is replaced by $2\pi Q \cdot x$; the coordinates r and z are treated like x and y , respectively.

Let us assume that $T(r, z)$ is represented by the finite element approximation T^e over the element Ω^e :

$$T(r, z) \approx T^e(r, z) = \sum_{j=1}^n T_j^e \psi_j^e(r, z) \quad (2.9.4)$$

The interpolation functions $\psi^e(r, z)$ are the same as those given in Eqs. (2.6.2) and (2.6.9) for linear triangular and rectangular elements, respectively, with $x = r$ and $y = z$. Substitution of Eq. (2.9.4) for T and ψ_i^e for w into the weak form gives the i th equation of the finite element model:

$$0 = 2\pi \sum_{j=1}^n \int_{\Omega^e} \left[\frac{\partial \psi_i^e}{\partial r} \left(k_{rr} T_j^e \frac{\partial \psi_j^e}{\partial r} \right) + \frac{\partial \psi_i^e}{\partial z} \left(k_{zz} T_j^e \frac{\partial \psi_j^e}{\partial z} \right) - wQ \right] r dr dz - \oint_{\Gamma^e} \psi_i^e q_n ds \quad (2.9.5)$$

or

$$0 = \sum_{j=1}^n K_{ij}^e T_j^e - Q_i^e - q_i^e \quad (2.9.6)$$

where

$$K_{ij}^e = 2\pi \int_{\Omega^e} \left(k_{rr} \frac{\partial \psi_i^e}{\partial r} \frac{\partial \psi_j^e}{\partial r} + k_{zz} \frac{\partial \psi_i^e}{\partial z} \frac{\partial \psi_j^e}{\partial z} \right) r dr dz \quad (2.9.7a)$$

$$Q_i^e = 2\pi \int_{\Omega^e} \psi_i^e Q r dr dz \quad (2.9.7b)$$

$$q_i^e = \oint_{\Gamma^e} q_n \psi_i^e ds \quad (2.9.7c)$$

2.10 Library of Finite Elements

2.10.1 Introduction

The objective of this section is to present a library of two-dimensional triangular and rectangular elements of the Lagrange family, i.e., elements over which only the function, not its derivatives, are interpolated. Once we have elements of different shapes and order at our disposal, we can choose appropriate elements and associated interpolation functions for a given problem. The interpolation functions are developed here for regularly shaped elements, called *master elements*. These elements can be used for numerical evaluation of integrals defined on irregularly shaped elements. This requires a transformation of the geometry from the actual element shape to its associated master element. We will discuss the numerical evaluation of integrals in Section 2.11.

2.10.2 Triangular Elements

The three-noded triangular element was developed in Section 2.6. Higher-order triangular elements (i.e., triangular elements with interpolation functions of higher degree) can be systematically developed with the help of the so-called *area coordinates*. For triangular elements, it is possible to construct three nondimensional coordinates $L_i(i = 1, 2, 3)$, which vary in a direction normal to the sides directly opposite each node (see Figure 2.10.1). The coordinates are defined such that

$$L_i = \frac{A_i}{A}, \quad A = \sum_{i=1}^3 A_i \quad (2.10.1)$$

where A_i is the area of the triangle formed by nodes j and k and an arbitrary point P in the element, and A is the total area of the element. For example, A_1 is the area of the shaded triangle which is formed by nodes 2 and 3 and point P . The point P is at a perpendicular distance of s from the side connecting nodes 2 and 3. We have $A_1 = bs/2$ and $A = bh/2$. Hence

$$L_1 = \frac{A_1}{A} = \frac{s}{h} \quad (2.10.2)$$

Clearly, L_1 is zero on side 2–3 (hence, zero at nodes 2 and 3) and has a value of unity at node 1.

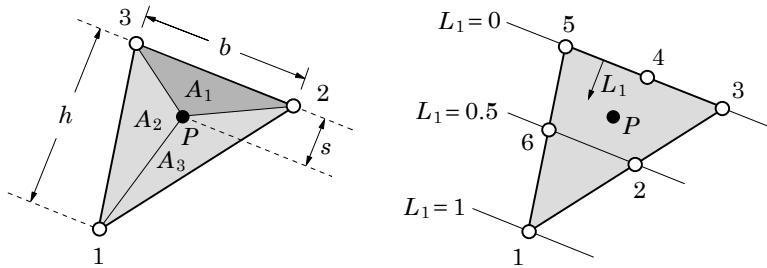


Figure 2.10.1: Definition of area coordinates L_i used for triangular elements.

Thus, L_1 is the interpolation function associated with node 1. Similarly, L_2 and L_3 are the interpolation functions associated with nodes 2 and 3, respectively. In summary, we have

$$\psi_i = L_i \quad (2.10.3)$$

The area coordinates L_i can be used to construct interpolation functions for higher-order triangular elements. For example, a higher-order element with k nodes per side (equally spaced on each side) has a total of n nodes

$$n = \sum_{i=0}^{k-1} (k - i) = k + (k - 1) + \cdots + 1 = \frac{k}{2}(k + 1) \quad (2.10.4)$$

and its degree is equal to $k - 1$. The explicit forms of the interpolation functions for the linear and quadratic elements are recorded below:

$$\{\Psi^e\} = \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \end{Bmatrix} ; \quad \{\Psi^e\} = \begin{Bmatrix} L_1(2L_1 - 1) \\ L_2(2L_2 - 1) \\ L_3(2L_3 - 1) \\ 4L_1L_2 \\ 4L_2L_3 \\ 4L_3L_1 \end{Bmatrix} \quad (2.10.5)$$

Note that the order of the interpolation functions in the above arrays corresponds to the node numbers shown in Figure 2.10.2a. Thus, the first three rows of the vectors in Eq. (2.10.5) correspond to the first three nodes of the linear and quadratic elements, which correspond to the three vertices of the triangular element. The last three rows of the second vector in Eq. (2.10.5) associated with the quadratic element correspond to the midside nodes of the triangular element. A similar node numbering scheme is used for rectangular elements, which are discussed next.

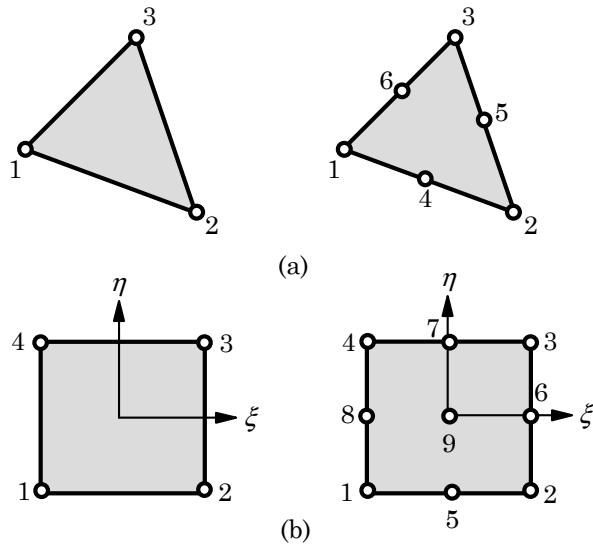


Figure 2.10.2: Linear and quadratic (a) triangular and (b) rectangular elements.

2.10.3 Rectangular Elements

The Lagrange interpolation functions associated with rectangular elements can be obtained from the tensor product of corresponding one-dimensional Lagrange interpolation functions. We take a local coordinate system (ξ, η) such that $-1 \leq (\xi, \eta) \leq 1$. This choice of local coordinate system is dictated by the Gauss quadrature rule used in the numerical evaluation of integrals over the element (see Section 2.12).

The linear and quadratic interpolation functions are given below (see Figure 2.10.2b for the node numbers).

$$\{\Psi^e\} = \frac{1}{4} \begin{Bmatrix} (1 - \xi)(1 - \eta) \\ (1 + \xi)(1 - \eta) \\ (1 + \xi)(1 + \eta) \\ (1 - \xi)(1 + \eta) \end{Bmatrix} \quad (2.10.6)$$

$$\{\Psi^e\} = \frac{1}{4} \begin{Bmatrix} (1 - \xi)(1 - \eta)(-\xi - \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ (1 + \xi)(1 - \eta)(\xi - \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ (1 + \xi)(1 + \eta)(\xi + \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ (1 - \xi)(1 + \eta)(-\xi + \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ 2(1 - \xi^2)(1 - \eta) - (1 - \xi^2)(1 - \eta^2) \\ 2(1 + \xi)(1 - \eta^2) - (1 - \xi^2)(1 - \eta^2) \\ 2(1 - \xi^2)(1 + \eta) - (1 - \xi^2)(1 - \eta^2) \\ 2(1 - \xi)(1 - \eta^2) - (1 - \xi^2)(1 - \eta^2) \\ 4(1 - \xi^2)(1 - \eta^2) \end{Bmatrix} \quad (2.10.7)$$

The *serendipity elements* are those rectangular elements which have no interior nodes. These elements have fewer nodes compared to the higher-order Lagrange elements. The interpolation functions of the serendipity elements are not complete, and they cannot be obtained using tensor products of one-dimensional Lagrange interpolation functions. Instead, an alternative procedure must be employed, as discussed in Reddy [4]. The interpolation functions for the 2-D quadratic serendipity element are given in Eq. (2.10.8) (see Figure 2.10.3). Although the functions are not complete, the serendipity elements have proven to be very effective in most applications.

$$\{\Psi^e\} = \frac{1}{4} \begin{Bmatrix} (1 - \xi)(1 - \eta)(-\xi - \eta - 1) \\ (1 + \xi)(1 - \eta)(\xi - \eta - 1) \\ (1 + \xi)(1 + \eta)(\xi + \eta - 1) \\ (1 - \xi)(1 + \eta)(-\xi + \eta - 1) \\ 2(1 - \xi^2)(1 - \eta) \\ 2(1 + \xi)(1 - \eta^2) \\ 2(1 - \xi^2)(1 + \eta) \\ 2(1 - \xi)(1 - \eta^2) \end{Bmatrix} \quad (2.10.8)$$

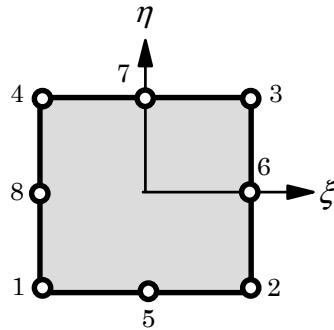


Figure 2.10.3: Quadratic rectangular serendipity element.

2.11 Numerical Integration

2.11.1 Preliminary Comments

An accurate representation of irregular domains (i.e., domains with curved boundaries) can be accomplished by the use of refined meshes and/or irregularly shaped curvilinear elements. For example, a non-rectangular region cannot be represented using rectangular elements; however, it can be represented by quadrilateral elements. Since the interpolation functions are easily derivable for a rectangular element and it is easier to evaluate integrals over rectangular geometries, we transform the finite element integral statements defined over quadrilaterals to a rectangle. The transformation results in complicated expressions for the integrands in terms of the coordinates used for the rectangular element. Therefore, numerical integration is used to evaluate such complicated integrals. The numerical integration schemes, such as the Gauss-Legendre numerical integration scheme, require the integral to be evaluated on a specific domain or with respect to a specific coordinate system. Gauss quadrature, for example, requires the integral to be expressed over a square region $\hat{\Omega}$ of dimension 2 by 2 with respect to the coordinate system, (ξ, η) to be such that $-1 \leq (\xi, \eta) \leq 1$. The transformation of the geometry and the variable coefficients of the differential equation from the problem coordinates (x, y) to the local coordinates (ξ, η) results in algebraically complex expressions, and they preclude analytical (i.e., exact) evaluation of the integrals. Thus, the transformation of a given integral expression, defined over element Ω^e , to one on the domain $\hat{\Omega}$ facilitates the numerical integration. Each element of the finite element mesh is transformed to $\hat{\Omega}$, only for the purpose of numerically evaluating the integrals (see Figure 2.11.1). The element $\hat{\Omega}$ is called a *master element*. For example, every quadrilateral element can be transformed to a square element with a side of length 2 and $-1 \leq (\xi, \eta) \leq 1$ that facilitates the use of Gauss-Legendre quadrature to evaluate integrals defined over the quadrilateral element.

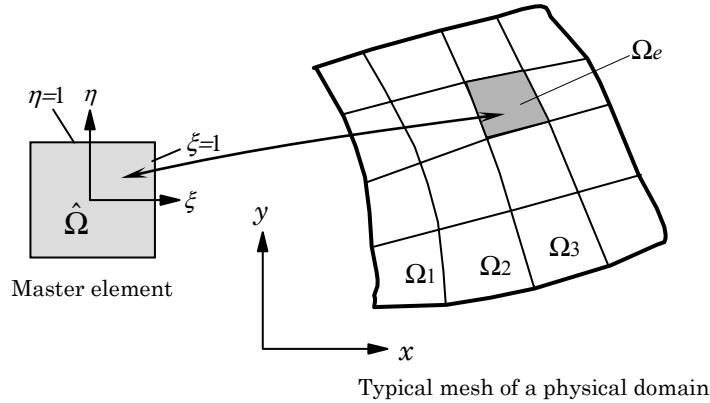


Figure 2.11.1: Transformation of arbitrarily shaped quadrilateral elements to the master rectangular element for numerical evaluation of integral expressions.

The transformation between the actual element Ω^e and the master element $\hat{\Omega}$ [or equivalently, between (x, y) and (ξ, η)] is accomplished by a coordinate transformation of the form

$$x = \sum_{j=1}^m x_j^e \phi_j^e(\xi, \eta), \quad y = \sum_{j=1}^m y_j^e \phi_j^e(\xi, \eta) \quad (2.11.1)$$

where ϕ_j denote the finite element interpolation functions of the master element $\hat{\Omega}$ (see Figure 2.11.1). The coordinates in the master element are chosen to be the natural coordinates (ξ, η) such that $-1 \leq (\xi, \eta) \leq 1$. This choice is dictated by the limits of integration in the Gauss quadrature rule, which is used to evaluate the integrals. For this case, the ϕ_j^e denote the interpolation functions of the four-node rectangular element shown in Figure 2.10.2b (i.e., $m = 4$). The transformation (2.11.1) maps a point (ξ, η) in the master element $\hat{\Omega}$ onto a point (x, y) in element Ω^e , and vice versa if the Jacobian of the transformation is positive-definite. The transformation maps the line $\xi = 1$ in $\hat{\Omega}$ to the line defined parametrically by $x = x(1, \eta)$ and $y = y(1, \eta)$ in the xy -plane. In other words, the master element $\hat{\Omega}$ is transformed, under the linear transformation, into a quadrilateral element (i.e., a four-sided element whose sides are not parallel) in the xy -plane. Conversely, every quadrilateral element of a mesh can be transformed to the same four-noded square (master) element $\hat{\Omega}$ in the (ξ, η) -plane.

In general, the dependent variable(s) of the problem are approximated by expressions of the form

$$u(x, y) = \sum_{j=1}^n u_j^e \psi_j^e(x, y) \quad (2.11.2)$$

The interpolation functions ψ_j^e used for the approximation of the dependent variable, in general, are different from ϕ_j^e used in the approximation of the geometry. Depending on the relative degree of approximations used for the geometry [see Eq. (2.11.1)] and the dependent variable(s) [see Eq. (2.11.2)], the finite element formulations are classified into three categories:

1. *Superparametric* ($m > n$). The approximation used for the geometry is higher order than that used for the dependent variable.
2. *Isoparametric* ($m = n$). Equal degree of approximation is used for both geometry and dependent variables.
3. *Subparametric* ($m < n$). Higher-order approximation of the dependent variable is used.

In the present context, we use only the isoparametric formulations.

2.11.2 Coordinate Transformations

It should be noted that the transformation of a quadrilateral element of a mesh to the master element $\hat{\Omega}$ is solely for the purpose of numerically evaluating the integrals (see Figure 2.11.1). *No transformation of the physical domain or elements is involved in the finite element analysis.* The resulting algebraic equations of the finite element formulation are always in terms of the nodal values of the physical domain. Different elements of the finite element mesh can be generated from the same master element by assigning appropriate global coordinates to each of the elements. Master elements of a different order define different transformations and hence different collections of finite elements within the mesh. For example, a quadratic rectangular master element can be used to generate a mesh of quadratic curvilinear quadrilateral elements. The transformations of a master element should be such that no spurious gaps exist between elements, and no element overlaps occur. For example, consider the element coefficients

$$K_{ij}^e = \int_{\Omega^e} \left[k_{xx}(x, y) \frac{\partial \psi_i^e}{\partial x} \frac{\partial \psi_j^e}{\partial x} + k_{yy}(x, y) \frac{\partial \psi_i^e}{\partial y} \frac{\partial \psi_j^e}{\partial y} + c(x, y) \psi_i^e \psi_j^e \right] dx dy \quad (2.11.3)$$

The integrand (i.e., the expression in the square brackets under the integral) is a function of the global coordinates x and y . We must rewrite it in terms of ξ and η using the transformation (2.11.1). Note that the integrand contains not only functions but also derivatives with respect to the global coordinates (x, y) . Therefore, we must relate $(\frac{\partial \psi_i^e}{\partial x}, \frac{\partial \psi_i^e}{\partial y})$ to $(\frac{\partial \psi_i^e}{\partial \xi}, \frac{\partial \psi_i^e}{\partial \eta})$ using the transformation (2.11.1).

The functions $\psi_i^e(x, y)$ can be expressed in terms of the local coordinates (ξ, η) by means of the transformation (2.11.1). Hence, by the chain rule of partial differentiation, we have

$$\begin{aligned} \frac{\partial \psi_i^e}{\partial \xi} &= \frac{\partial \psi_i^e}{\partial x} \frac{\partial x}{\partial \xi} + \frac{\partial \psi_i^e}{\partial y} \frac{\partial y}{\partial \xi} \\ \frac{\partial \psi_i^e}{\partial \eta} &= \frac{\partial \psi_i^e}{\partial x} \frac{\partial x}{\partial \eta} + \frac{\partial \psi_i^e}{\partial y} \frac{\partial y}{\partial \eta} \end{aligned}$$

or, in matrix notation,

$$\begin{Bmatrix} \frac{\partial \psi_i^e}{\partial \xi} \\ \frac{\partial \psi_i^e}{\partial \eta} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial \psi_i^e}{\partial x} \\ \frac{\partial \psi_i^e}{\partial y} \end{Bmatrix} \quad (2.11.4)$$

which gives the relation between the derivatives of ψ_i^e with respect to the global and local coordinates. The matrix in Eq. (2.11.4) is called the *Jacobian matrix* of the transformation (2.11.1):

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \quad (2.11.5)$$

Note from the expression given for K_{ij}^e in Eq. (2.11.3) that we must relate $(\frac{\partial \psi_i^e}{\partial x}, \frac{\partial \psi_i^e}{\partial y})$ to $(\frac{\partial \psi_i^e}{\partial \xi}, \frac{\partial \psi_i^e}{\partial \eta})$, whereas Eq. (2.11.4) provides the inverse relations. Therefore, Eq. (2.11.4) must be inverted by inverting the Jacobian matrix:

$$\begin{Bmatrix} \frac{\partial \psi_i^e}{\partial x} \\ \frac{\partial \psi_i^e}{\partial y} \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} \frac{\partial \psi_i^e}{\partial \xi} \\ \frac{\partial \psi_i^e}{\partial \eta} \end{Bmatrix} \quad (2.11.6)$$

This requires that the Jacobian matrix \mathbf{J} be nonsingular (i.e., $\det \mathbf{J} \neq 0$).

Using the transformation (2.11.1), we can write

$$\frac{\partial x}{\partial \xi} = \sum_{j=1}^m x_j \frac{\partial \phi_j^e}{\partial \xi}, \quad \frac{\partial y}{\partial \xi} = \sum_{j=1}^m y_j \frac{\partial \phi_j^e}{\partial \xi} \quad (2.11.7a)$$

$$\frac{\partial x}{\partial \eta} = \sum_{j=1}^m x_j \frac{\partial \phi_j^e}{\partial \eta}, \quad \frac{\partial y}{\partial \eta} = \sum_{j=1}^m y_j \frac{\partial \phi_j^e}{\partial \eta} \quad (2.11.7b)$$

and by means of Eq. (2.11.5) one can compute the Jacobian matrix and then its inverse. Thus, given the global coordinates (x_j, y_j) of element nodes and the interpolation functions ϕ_j^e used for geometry, the Jacobian matrix can be evaluated using Eq. (2.11.5). A necessary and sufficient condition for \mathbf{J}^{-1} to exist is that the determinant J , called the Jacobian, be nonzero at every point (ξ, η) in $\hat{\Omega}$:

$$J \equiv \det \mathbf{J} = \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi} \neq 0. \quad (2.11.8)$$

From Eq. (2.11.8) it is clear that the functions $\xi(x, y)$ and $\eta(x, y)$ must be continuous, differentiable, and invertible. Moreover, the transformation should be algebraically simple so that the Jacobian matrix can be easily evaluated. Transformations of the form in Eq. (2.11.1) satisfy these requirements and the requirement that no spurious gaps between elements or overlapping of elements occur.

Returning to numerical evaluation of integrals, we have from Eq. (2.11.6),

$$\begin{Bmatrix} \frac{\partial \psi_i^e}{\partial x} \\ \frac{\partial \psi_i^e}{\partial y} \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} \frac{\partial \psi_i^e}{\partial \xi} \\ \frac{\partial \psi_i^e}{\partial \eta} \end{Bmatrix} \equiv \mathbf{J}^* \begin{Bmatrix} \frac{\partial \psi_i^e}{\partial \xi} \\ \frac{\partial \psi_i^e}{\partial \eta} \end{Bmatrix} \quad (2.11.9)$$

where J_{ij}^* is the element in position (i, j) of the inverse of the Jacobian matrix \mathbf{J} . The element area $dA = dx dy$ in element Ω^e is transformed to

$$dA = J d\xi d\eta \quad (2.11.10)$$

in the master element $\hat{\Omega}$.

Equations (2.11.7)–(2.11.10) provide the necessary relations to transform integral expressions on any element Ω^e to an associated master element $\hat{\Omega}$. For instance, consider the integral expression in Eq. (2.11.3), where k_{xx} , k_{yy} , and c are functions

of x and y . Suppose that the finite element Ω^e can be generated by the master element $\hat{\Omega}^e$. Under the transformation (2.11.1) we can write

$$\begin{aligned} K_{ij}^e &= \int_{\Omega^e} \left[k_{xx}(x, y) \frac{\partial \psi_i^e}{\partial x} \frac{\partial \psi_j^e}{\partial x} + k_{yy}(x, y) \frac{\partial \psi_i^e}{\partial y} \frac{\partial \psi_j^e}{\partial y} + c(x, y) \psi_i^e \psi_j^e \right] dx dy \\ &= \int_{\hat{\Omega}^e} F_{ij}(\xi, \eta) d\xi d\eta \end{aligned} \quad (2.11.11)$$

The discussion presented above is valid for master elements of both rectangular and triangular geometry.

2.11.3 Integration over a Master Rectangular Element

Integrals defined over a rectangular master element $\hat{\Omega}_R$ can be numerically evaluated using the Gauss–Legendre quadrature formulas

$$\int_{\hat{\Omega}_R} F(\xi, \eta) d\xi d\eta = \int_{-1}^1 \int_{-1}^1 F(\xi, \eta) d\xi d\eta \approx \sum_{I=1}^M \sum_{J=1}^N F(\xi_I, \eta_J) W_I W_J \quad (2.11.12)$$

where M and N denote the number of Gauss quadrature points, (ξ_I, η_J) denote the Gauss point coordinates, and W_I and W_J denote the corresponding Gauss weights as shown in Table 2.11.1 (from Table 7.1.2 in Reddy [4]).

The selection of the number of Gauss points is based on the following formula: a polynomial of degree p is integrated exactly employing $N = \text{int}[(p+1)/2] + 1$ integration points. In most cases, the interpolation functions are of the same degree

Table 2.11.1: Gauss quadrature points and weights for rectangular elements.

$$\int_{-1}^1 F(\xi) d\xi = \sum_{I=1}^N F(\xi_I) W_I$$

N	Points, ξ_I	Weights, W_I
1	0.0000000000	2.0000000000
2	± 0.5773502692	1.0000000000
3	0.0000000000 ± 0.7745966692	0.8888888889 0.5555555555
4	± 0.3399810435 ± 0.8611363116	0.6521451548 0.3478548451
5	0.0000000000 ± 0.5384693101 ± 0.9061798459	0.5688888889 0.4786286705 0.2369268850
6	± 0.2386191861 ± 0.6612093865 ± 0.9324695142	0.4679139346 0.3607615730 0.1713244924

in both ξ and η , and therefore one has $M = N$. When the integrand is of a different degree in ξ and η , the number of Gauss points is selected on the basis of the largest-degree polynomial. The minimum allowable quadrature rule is one that yields the area or volume of the element exactly. The maximum degree of the polynomial refers to the degree of the highest polynomial in ξ or η that is present in the integrands of the element matrices of the type in Eq. (2.11.3). Note that the polynomial degree of coefficients as well as J_{ij} should be accounted for in determining the total polynomial degree of the integrand. Of course, the coefficients k_{xx} , k_{yy} , and c and J_{ij} in general may not be polynomials. In those cases, their functional variations must be approximated by a suitable polynomial (for example, by a binomial series) in order to determine the polynomial degree of the integrand.

2.11.4 Integration over a Master Triangular Element

In the preceding section we discussed numerical integration on quadrilateral elements which can be used to represent very general geometries as well as field variables in a variety of problems. Here we discuss numerical integration on triangular elements. Since quadrilateral elements can be geometrically distorted, it is possible to distort a quadrilateral element to obtain a required triangular element by moving the position of the corner nodes, and the fourth corner in the quadrilateral is merged with one of the neighboring nodes. In actual computation, this is achieved by assigning the same global node number to two corner nodes of the quadrilateral element. Thus, master triangular elements can be obtained in a natural way from associated master rectangular elements. Here we discuss the transformations from a master triangular element to an arbitrary triangular element.

We choose the unit right isosceles triangle (see Table 2.11.2) as the master element. An arbitrary triangular element Ω^e can be generated from the master triangular element $\hat{\Omega}_T$ by transformation of the form (2.11.1). The derivatives of ψ_i with respect to the global coordinates can be computed from Eq. (2.11.6), which take the form

$$\begin{Bmatrix} \frac{\partial \psi_i^e}{\partial x} \\ \frac{\partial \psi_i^e}{\partial y} \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} \frac{\partial \psi_i^e}{\partial L_1} \\ \frac{\partial \psi_i^e}{\partial L_2} \end{Bmatrix} \quad (2.11.13a)$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial L_1} & \frac{\partial y}{\partial L_1} \\ \frac{\partial x}{\partial L_2} & \frac{\partial y}{\partial L_2} \end{bmatrix} \quad (2.11.13b)$$

Note that only L_1 and L_2 are treated as linearly independent coordinates because $L_3 = 1 - L_1 - L_2$.

After transformation, integrals on $\hat{\Omega}_T$ have the form

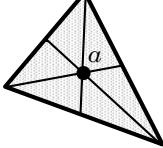
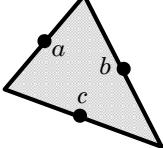
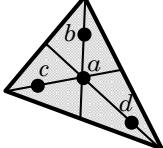
$$\int_{\Omega^e} G(\xi, \eta) d\xi d\eta = \int_{\Omega^e} G(L_1, L_2, L_3) dL_1 dL_2 \quad (2.11.14)$$

which can be approximated by the quadrature formula

$$\int_{\Omega^e} G(L_1, L_2, L_3) dL_1 dL_2 \approx \sum_{I=1}^N G(\mathbf{S}_I) W_I \quad (2.11.15)$$

where W_I and \mathbf{S}_I denote the weights and integration points of the quadrature rule. Table 2.11.2 (from Table 9.3.2 of Reddy [4]) contains the location of integration points and weights for one-point, three-point, and four-point quadrature rules over triangular elements. For evaluation of integrals whose integrands are polynomials of degree equal to or higher than 5 (in any of the area coordinates) the reader must consult books on numerical integration.

Table 2.11.2: Quadrature weights and points for triangular elements.

Number of integration points	Degree of polynomial Order of the residual	Location of integration points					Geometric locations
		L_1	L_2	L_3	W		
1	$O(h^2)$				1	a	
3	$O(h^3)$	1/2 1/2 0	0 1/2 1/2	1/2 0 1/2	1/3	a b c	
4	$O(h^4)$	1/3 0.6 0.2 0.2	1/3 0.2 0.2 0.2	1/3 0.2 0.2 0.6	-27/48 25/48 25/48 25/48	a b c d	

2.12 Modeling Considerations

Here we discuss some aspects of finite element model development. Guidelines concerning mesh generation, boundary flux representation, and imposition of boundary conditions are discussed in very general terms (see Reddy [4] for further details).

2.12.1 Mesh Generation

Generation of a finite element mesh for a given domain should follow the guidelines listed below:

1. The mesh should represent the geometry of the computational domain and boundary flux representation accurately.
2. The mesh should be such that large gradients in the solution (temperature and velocities) are adequately represented.
3. The mesh should not contain elements with very large aspect ratios and/or angular distortions.

Within the above guidelines, the mesh used can be *coarse* (i.e., few elements) or *refined* (i.e., many elements), and may consist of one or more orders and types of elements (e.g., linear and quadratic, triangular and quadrilateral). It should be noted that the choice of element type and mesh is problem dependent. An analyst with physical insight into the process being simulated can make a better choice of elements and mesh for the problem at hand than one who does not have the physical insight. One should evaluate the results obtained in the light of physical understanding and approximate analytical and/or experimental information.

Generation of meshes of a single element type (i.e., linear elements or quadratic elements) is easy, because elements of the same degree are compatible with each other. Mesh refinements involve several options. Refine the mesh by subdividing existing elements into two or more elements of the same type (see Figure 2.12.1a). This is called the *h-version mesh refinement*. Refine the mesh by replacing existing elements by elements of higher order (see Figure 2.12.1b). This type of refinement is called the *p-version mesh refinement*. The *(h,p)-version mesh refinement* is one in which elements are subdivided into two or more elements in some places and replaced by higher-order elements in other places. Generally, local mesh refinements should be such that very small elements are not placed adjacent to very large elements.

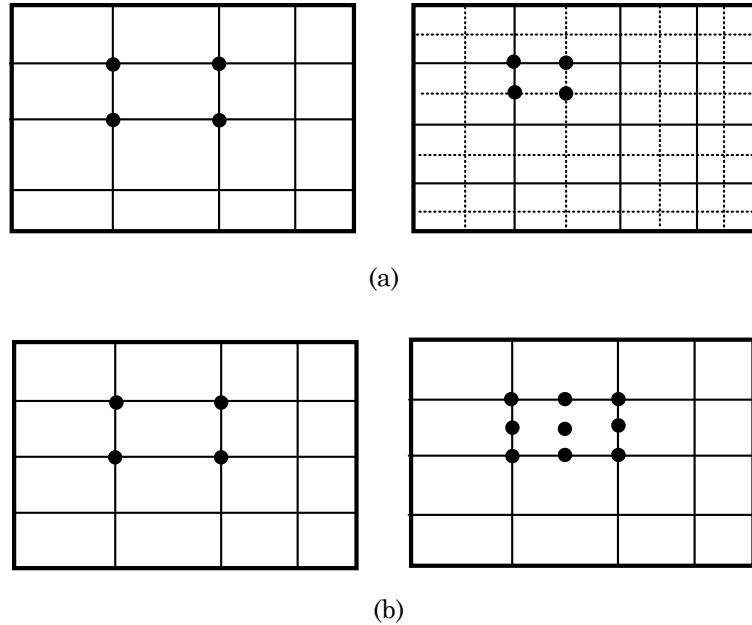


Figure 2.12.1: The two types of mesh refinements in finite element analysis: (a) *h*-refinement (refinement with the same order of elements); (b) *p*-refinement (refinement with higher-order elements).

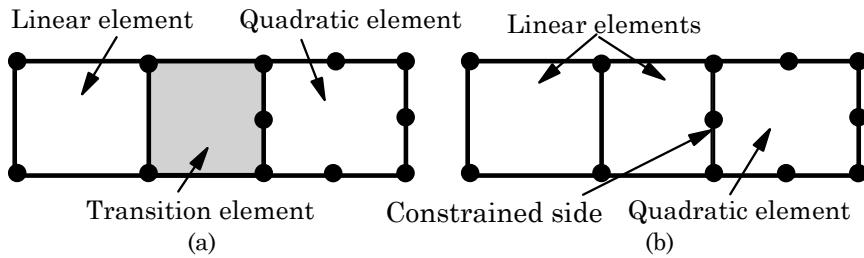


Figure 2.12.2: Connecting linear and quadratic elements with (a) transition elements, and (b) linear constraint equations.

To accomplish local mesh refinements it may be necessary to combine elements of different orders. There are two ways to combine elements of different orders. One way is to use transition elements, which have different numbers of nodes on different sides of the element (see Figure 2.12.2a). Another way is to impose a condition that constrains the midside node to have the same value as that at the node of the lower-order element (see Figure 2.12.2b). Neither combination enforces inter-element continuity of the solution at every point of the entire interface.

2.12.2 Representation of Boundary Flux

When the temperature distribution in the domain is approximated by a set of finite elements, the boundary flux distribution, if known, must be replaced by a set of equivalent nodal heats. The accuracy of the solution depends on the element type and mesh used to represent the domain and the representation of actual flux distribution. Use of linear elements, for example, to represent a curved boundary will change the actual distribution (see Figure 2.12.3). Of course, mesh refinements by *h*-version or *p*-version will improve the representation of the specified boundary flux. For example, if the temperature is approximated using the nine-node quadratic elements, the nodal contributions of a distributed heat flux should not be evaluated using four-node linear elements. This would be an inconsistent computation of nodal fluxes. Mesh refinements make the flux representation more accurate, and hence convergence of the numerical solution to the actual solution can be expected.

2.12.3 Imposition of Boundary Conditions

In most problems one encounters situations where the portion of the boundary on which flux is specified has points in common with the portion of the boundary on which the temperature is specified. In other words, at a few nodal points of the mesh, both the heat input and temperature may be specified, e.g., at an exterior corner. Such points are called *singular points*. Obviously, one cannot impose both boundary conditions at the same point. As a general rule, one should impose the essential (i.e., temperature) boundary condition at the singular points and disregard the flux boundary condition. Of course, if the true situation in a problem is that the flux boundary condition is imposed and the temperature boundary condition is a result of it, then consideration must be given to the former one.

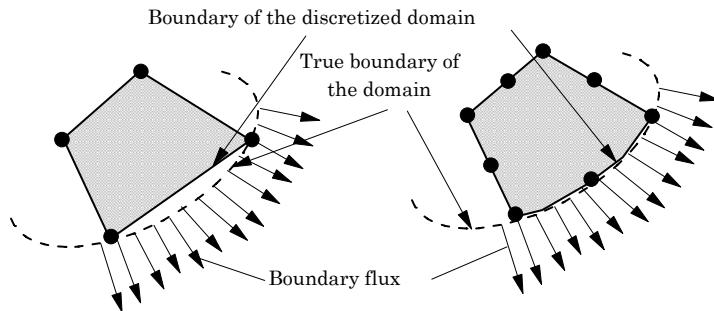


Figure 2.12.3: Representation of boundary flux in finite element analysis.

Another type of singularity one encounters in the analysis of thermal boundary value problems is the specification of two different values of a temperature or heat input at the same boundary point. In the finite element analysis, one must make a choice between the two values or take a weighted average of the two. In any case, one must note that the true boundary condition is replaced by an approximate condition. The closeness of the approximate boundary condition to the true one depends on the size of the element containing the point. It is often necessary to make a mesh refinement in the vicinity of the singular point to obtain an acceptable solution.

In most practical problems a variety of approximations are introduced before reaching the finite element simulation; additional approximations, some of which were described here, are introduced by the computational model. It is important to keep various sources of error in perspective. A feel for the relative proportions of various errors introduced into the analysis helps the analyst to make a decision on selecting a mesh and representing the boundary conditions. In summary, engineering knowledge and experience with the problem being analyzed are an essential part of any engineering analysis.

2.13 Illustrative Examples

In this section simple examples of applications of the finite element method are discussed. In the interest of simplicity, the examples are limited to heat transfer problems on rectangular domains and the use of triangular and rectangular elements. Additional examples can be found in the textbook by Reddy [4]. More complicated problems, which reflect practical situations, will be addressed in the subsequent chapters.

A sample finite element program, *FEM2DHT*, that reflects the ideas presented in this chapter is discussed in Appendix A. This is a simplified version of the program *FEM2D* from Reddy [4]. The program is limited to linear two-dimensional problems, with linear and quadratic triangular and rectangular elements. Plane as well as axisymmetric (see Section 2.9) geometries can be modeled, both conductive and convective boundary conditions can be handled, and steady as well as transient heat transfer problems can be solved using the program. All of the results presented in this section were obtained using program *FEM2DHT* (see Appendix A for data input instructions for using the program).

2.13.1 Example 1

2.13.1.1 Problem description

Consider the conduction heat transfer in a square plate of dimension 2 cm by 2 cm, conductivity k (W/m/ $^{\circ}$ C), and uniform internal heat generation of Q_0 (W/m 3). The edges of the plate are maintained at a temperature of $T_0 = 0$ ($^{\circ}$ C) (see Figure 2.13.1). We wish to determine the steady temperature distribution inside the plate using the finite element method.

The problem can be expressed mathematically as one of solving the equation

$$-k\nabla^2T = Q_0 \quad \text{or} \quad -k\left(\frac{\partial^2T}{\partial x^2} + \frac{\partial^2T}{\partial y^2}\right) = Q_0 \quad \text{in } \Omega \quad (2.13.1a)$$

$$T = T_0 \quad \text{on } \Gamma \quad (2.13.1b)$$

Note that the domain Ω is a unit square, and the boundary Γ consists of the four sides bounding Ω .

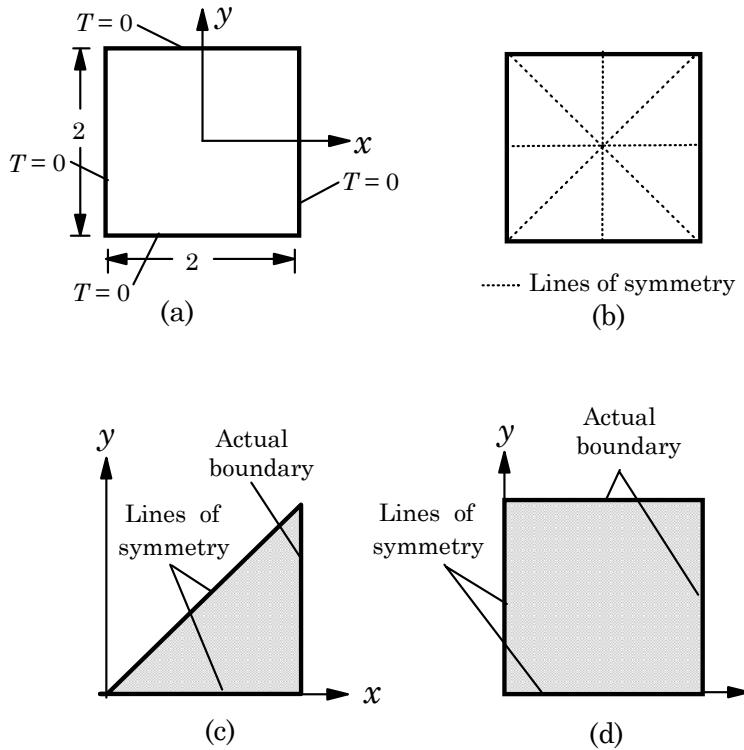


Figure 2.13.1: Heat transfer in a square domain. (a) Geometry and boundary conditions. (b) Problem symmetries. (c) Computational domain for the mesh of triangular elements. (d) Computational domain for the mesh of rectangular elements.

Equation (2.13.1) has an analytical (series) solution, which can be obtained using the method of separation of variables. We wish to compare the finite element solution of the problem with the analytical solution obtained with meshes of triangular elements and rectangular elements. Each mesh consists of only one type of element.

The problem has symmetries that can be exploited in the finite element analysis. A problem possesses symmetry of the solution about a line only when there exists a symmetry of the (1) geometry, (2) material properties, (3) source (i.e., heat generation), and (4) boundary conditions of the problem. Such a line is called the *line of symmetry*. When a line of symmetry exists in a problem, then it is sufficient to model the domain on either side of the line. Then the line of symmetry becomes a part of the boundary of the computational domain. On such boundaries, the normal derivative of the solution (i.e., derivative of the solution with respect to the coordinate normal to the line of symmetry) is zero. In the context of heat transfer problems, this implies that the temperature gradient (or heat flux) across the line of symmetry is zero:

$$q_n \equiv \frac{\partial T}{\partial n} = \hat{\mathbf{n}} \cdot \nabla T = \frac{\partial T}{\partial x} n_x + \frac{\partial T}{\partial y} n_y = 0 \quad (2.13.2)$$

The problem at hand has the symmetry about the $x = 0$ and $y = 0$ axes; it is also symmetric about the diagonal line $x = y$ (see Figure 2.13.1b). Thus, we can use a quadrant of the domain for meshes of rectangular elements and an octant of the domain for meshes of triangular elements to analyze the problem. Of course, it is possible to mix triangular and rectangular elements to represent the domain as well as the solution, but we shall use only one type of element in each mesh and the same degree of approximation for both geometry and solution (i.e., isoparametric formulation).

2.13.1.2 Solution by linear triangular elements

Due to the symmetry along the diagonal $x = y$, we model the triangular domain shown in Figure 2.13.1c. As a first choice we use a uniform mesh of four linear triangular elements to represent the domain (see Figure 2.13.2a), and then a refined mesh (see Figure 2.13.2b) to compare the solutions. In the present case, there is no discretization error involved in the problem because the geometry is exactly represented.

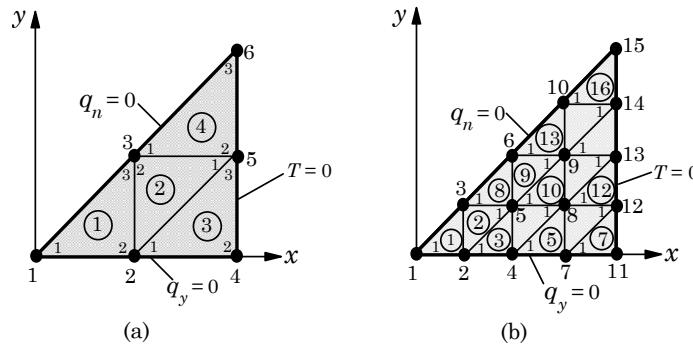


Figure 2.13.2: Discretization of the computational domain with triangular elements: (a) mesh of four elements; (b) mesh of sixteen elements.

We consider element 1 as the typical element with its local coordinate system (\bar{x}, \bar{y}) . Suppose that the element dimensions, i.e., length and height are a and b , respectively. The coordinates of the element nodes are

$$(\bar{x}_1, \bar{y}_1) = (0, 0), (\bar{x}_2, \bar{y}_2) = (a, 0), (\bar{x}_3, \bar{y}_3) = (a, b) \quad (2.13.3)$$

Hence the parameters α_i , β_i , and γ_i are given by

$$\begin{aligned} \alpha_1 &= \bar{x}_2\bar{y}_3 - \bar{x}_3\bar{y}_2 = ab, \quad \alpha_2 = \bar{x}_3\bar{y}_1 - \bar{x}_1\bar{y}_3 = 0, \quad \alpha_3 = \bar{x}_1\bar{y}_2 - \bar{x}_2\bar{y}_1 = 0, \\ \beta_1 &= \bar{y}_2 - \bar{y}_3 = -b, \quad \beta_2 = \bar{y}_3 - \bar{y}_1 = b, \quad \beta_3 = \bar{y}_1 - \bar{y}_2 = 0, \\ \gamma_1 &= -(\bar{x}_2 - \bar{x}_3) = 0, \quad \gamma_2 = -(\bar{x}_3 - \bar{x}_1) = -a, \quad \gamma_3 = -(\bar{x}_1 - \bar{x}_2) = a \end{aligned} \quad (2.13.4)$$

For the mesh shown in Figure 2.13.2a, we have

$$\mathbf{K}^1 = \mathbf{K}^2 = \mathbf{K}^3 = \mathbf{K}^4; \quad \mathbf{Q}^1 = \mathbf{Q}^2 = \mathbf{Q}^3 = \mathbf{Q}^4 \quad (2.13.5)$$

Therefore, the element coefficients K_{ij}^e and Q_i^e ($e = 1, 2, 3, 4$) are given by (note that a denotes the length of side connecting element nodes 1 and 2) [see Eqs. (2.6.6)]

$$\mathbf{K}^e = \frac{k}{2} \begin{bmatrix} \alpha & -\alpha & 0 \\ -\alpha & \gamma & -\beta \\ 0 & -\beta & \beta \end{bmatrix}, \quad \mathbf{Q}^e = \frac{Q_0 ab}{6} \begin{Bmatrix} 1 \\ 1 \\ 1 \end{Bmatrix} \quad (2.13.6)$$

where $\alpha = \frac{b}{a}$, $\beta = \frac{a}{b}$, and $\gamma = \alpha + \beta$. The element matrix in (2.13.6) is valid for the Laplace operator $-\nabla^2$ on any right-angle triangle with sides a and b in which the right angle is at node 2, and the diagonal line of the triangle connects node 3 to node 1. Note that the off-diagonal coefficient associated with the nodes on the diagonal line is zero for a right-angle triangle. These observations can be used to write the element matrix associated with the Laplace operator on any right-angle triangle, i.e., for any element-node numbering system.

The assembled coefficient matrix for the finite element mesh is 6×6 , because there are six global nodes with one unknown per node. The assembled matrix can be obtained directly by using the correspondence between the global nodes and the local nodes, as explained earlier.

The specified boundary conditions on the primary degrees of freedom of the problem are

$$T_4 = T_5 = T_6 = T_0 = 0 \quad (2.13.7)$$

The specified secondary degrees of freedom at nodes 1, 2, and 3 are all zero because the flux is zero there:

$$q_1^1 = 0, \quad q_2^1 + q_3^2 + q_1^3 = 0, \quad q_3^1 + q_2^2 + q_1^4 = 0 \quad (2.13.8)$$

Since T_4 , T_5 , and T_6 are known, the secondary variables at these nodes are unknown, and they can be obtained in the post-computation.

2.13.1.3 Solution by linear rectangular elements

Note that we cannot exploit the symmetry along the diagonal $x = y$ to our advantage when we use a mesh of rectangular elements. Therefore, we use a 2×2 uniform mesh of four linear rectangular elements and a refined 4×4 mesh (see Figure 2.13.3a) to discretize a quadrant of the domain. Once again, no discretization error is introduced in the present case.

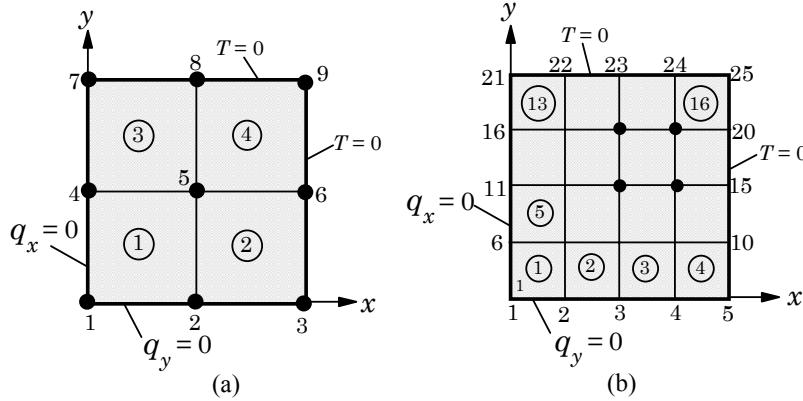


Figure 2.13.3: Discretization of the computational domain with rectangular elements: (a) mesh of four elements; (b) mesh of sixteen elements.

All elements in the mesh are identical. Therefore, we have the following element matrices [see Eqs. (2.6.10a-d)] for a typical element of the mesh ($\alpha = \frac{b}{a}$, $\beta = \frac{a}{b}$):

$$\mathbf{K}^e = \frac{k}{6} \begin{bmatrix} 2(\alpha + \beta) & -2\alpha + \beta & -(\alpha + \beta) & \alpha - 2\beta \\ -2\alpha + \beta & 2(\alpha + \beta) & \alpha - 2\beta & -(\alpha + \beta) \\ -(\alpha + \beta) & \alpha - 2\beta & 2(\alpha + \beta) & -2\alpha + \beta \\ \alpha - 2\beta & -(\alpha + \beta) & -2\alpha + \beta & 2(\alpha + \beta) \end{bmatrix} \quad (2.13.9a)$$

$$\mathbf{Q}^e = \frac{Q_0 ab}{4} \begin{Bmatrix} 1 \\ 1 \\ 1 \end{Bmatrix} \quad (2.13.9b)$$

The boundary conditions on the secondary variables are

$$q_1^1 = 0, \quad q_2^1 + q_1^2 = 0, \quad q_4^1 + q_1^3 = 0 \quad (2.13.10a)$$

and the balance of secondary variables at global node 5 requires

$$q_3^1 + q_4^2 + q_2^3 + q_1^4 = 0 \quad (2.13.10b)$$

2.13.1.4 Discussion of the results

The finite element solutions obtained by two different meshes of triangular elements and two different meshes of rectangular elements are compared in Table 2.13.1 at $x = 0$ for varying y for $k = 1$, $Q_0 = 1$, and $T_0 = 0$. The finite element solution obtained by 16 triangular elements (in an octant) is the most accurate when compared to the

series solution. The accuracy of the triangular element mesh is due to the larger number of elements it has compared to the number of elements in the rectangular element mesh for the same size of the domain.

Table 2.13.1: Convergence of the finite element solutions to the analytical solution of Eq. (2.13.1).

y	Triangular elements		Rectangular elements		Series solution
	4 elements	16 elements	4 elements	16 elements	
0.0	0.3125	0.3013	0.3107	0.2984	0.2947
0.25	0.2709*	0.2805	0.2759*	0.2824	0.2789
0.50	0.2292	0.2292	0.2411	0.2322	0.2293
0.75	0.1146*	0.1393	0.1205*	0.1414	0.1397
1.0	0.0000	0.0000	0.0000	0.0000	0.0000

* Interpolated values.

The finite element nodal temperatures obtained with the 4×4 mesh of linear triangular elements in a quadrant are compared in Table 2.13.2 with the finite difference solution obtained with a 4×4 rectangular grid and with the exact (or series) solution. The node numbering scheme is the same as that used in Figure 2.13.3b for a rectangular element mesh. The central difference approximation is used to approximate the second derivatives in the equation. The finite element nodal temperatures are slightly more accurate, except at node 1, than those obtained with the finite difference method. One should remember that the finite element method minimizes the residual of the approximation of the differential equation over each element and not the error in the nodal values. In other words, the nodal values are adjusted such that the weighted-integral of the error in the approximation of the differential equation is minimized over each element.

The solution T and its gradient can be computed at any interior point of the domain using the finite element approximation of $T(x, y)$ [see Eq. (2.3.1)]. We have

$$T^e(x, y) = \sum_{j=1}^n T_j^e \psi_j^e(x, y) \quad (2.13.11a)$$

$$\frac{\partial T^e}{\partial x} = \sum_{j=1}^n T_j^e \frac{\partial \psi_j^e}{\partial x} \quad (2.13.11b)$$

$$\frac{\partial T^e}{\partial y} = \sum_{j=1}^n T_j^e \frac{\partial \psi_j^e}{\partial y} \quad (2.13.11c)$$

where T_j^e is the value of $T^e(x, y)$ at the j th node of the element, and $\psi_j^e(x, y)$ ($j = 1, 2, \dots, n$) are the interpolation functions. Note that for a linear triangular element the components $(\frac{\partial T^e}{\partial x}, \frac{\partial T^e}{\partial y})$ of the temperature gradient vector are constants over an entire element, whereas $\frac{\partial T^e}{\partial x}$ is linear in y and $\frac{\partial T^e}{\partial y}$ is linear in x for a linear rectangular element.

Table 2.13.2: Comparison of the finite element solution (FES) with the finite difference solution (FDS) and analytical (series) solution of Eq. (2.13.1). A 4×4 mesh of linear triangular elements is used in the finite element method and a comparable mesh is used in the finite difference method.

Node	Analytical	FDS	Error	FES	Error
1	0.2947	0.2911	0.0036	0.3013	-0.0066
2	0.2789	0.2755	0.0034	0.2805	-0.0016
3	0.2293	0.2266	0.0027	0.2292	0.0001
4	0.1397	0.1381	0.0016	0.1392	0.0005
5	0.0000	0.0000	0.0000	0.0000	0.0000
7	0.2642	0.2609	0.0033	0.2645	-0.0003
8	0.2178	0.2151	0.0027	0.2172	0.0006
9	0.1333	0.1317	0.0016	0.1327	0.0006
10	0.0000	0.0000	0.0000	0.0000	0.0000
13	0.1811	0.1787	0.0024	0.1801	0.0010
14	0.1127	0.1110	0.0017	0.1117	0.0010
15	0.0000	0.0000	0.0000	0.0000	0.0000
19	0.0728	0.0711	0.0017	0.0715	0.0013
20	0.0000	0.0000	0.0000	0.0000	0.0000
25	0.0000	0.0000	0.0000	0.0000	0.0000

2.13.2 Example 2

Consider a cylinder of height $L = 1$ cm, radius $R = 1$ cm, thermal conductivity $k = 25$ W/(m°C), and constant internal heat generation of $Q_0 = 5 \times 10^8$ W/m³. The top and bottom faces of the cylinder are assumed to be insulated and the boundary surface is maintained at $T_0 = 100^\circ\text{C}$. The assumption of an insulated boundary condition at the top and bottom of the cylinder makes the heat flow one-dimensional along the radial direction. For this case, the steady temperature along the radial direction is given by

$$T(r) = T_0 + \frac{Q_0 R^2}{4k} \left(1 - \frac{r^2}{R^2} \right) \quad (2.13.12)$$

Table 2.13.3 contains the finite element solution obtained with various finite element meshes along the r -coordinate. Only one element is used in the z -direction. The 5×1 mesh of nine-node quadratic elements gives exact values at the nodes.

Next we consider a cylinder of height $L = 2$ cm, radius $R = 1$ cm, thermal conductivity $k = 25$ W/(m°C), and constant internal heat generation of $Q_0 = 5 \times 10^8$ W/m³. The top and bottom faces and the surface of the cylinder are maintained at $T_0 = 100^\circ\text{C}$. For this case, the steady solution is two-dimensional. Exploiting the symmetry about $z = 1$ cm, a 10×10 mesh of linear rectangular elements is used. The results are presented in Table 2.13.4.

Table 2.13.3: Finite element solutions of a 1-D axisymmetric problem.

$r \times 10^{-2}$	5LR*	10LR	5QR
0.0	611.92	603.56	600.00
0.1	—	596.89	595.00
0.2	585.25	581.33	580.00
0.3	—	556.00	555.00
0.4	523.03	520.76	520.00
0.5	—	475.58	475.00
0.6	421.69	420.43	420.00
0.7	—	355.30	355.00
0.8	280.74	280.19	280.00
0.9	—	195.09	195.00
1.0	100.00	100.00	100.00

*5LR = 5 linear rectangular elements; 10LR = 10 linear rectangular elements; 5QR = 5 quadratic rectangular elements.

2.13.3 Example 3

Consider a plane wall of height $b = 1$ cm, thickness $a = 1$ cm, thermal conductivity $k = 18$ W/(m°C), and constant internal heat generation of $Q_0 = 7.2 \times 10^7$ W/m³. The boundaries at $y = 0, b$ of the domain are assumed to be insulated, the boundary at $x = 0$ is maintained at $T_0 = 50^\circ\text{C}$, and the boundary at $x = a$ is exposed to ambient temperature $T_c = 100^\circ\text{C}$. The film coefficient is $h_c = 200$ W/(m²°C). The assumption of an insulated boundary condition at $y = 0, b$ of the domain makes the heat flow one-dimensional along the x -direction. For this case, the temperature is given by

$$T(x) = 50 + 5 \frac{x}{a} + 200 \left(1.9 - \frac{x}{a} \right) \frac{x}{a} \quad (2.13.13)$$

Table 2.13.4: Finite element solutions of a 2-D axisymmetric problem.

$r(z = 10^{-3})$	Temp.	$r(z = 10^{-2})$	Temp.	$z(r = 0.0)$	Temp.
0.000	195.86	0.000	504.64	0.000	100.00
0.001	195.03	0.001	499.82	0.001	195.86
0.002	193.06	0.002	488.52	0.002	274.19
0.003	189.78	0.003	469.93	0.003	337.35
0.004	185.07	0.004	443.70	0.004	387.55
0.005	178.74	0.005	409.43	0.005	426.72
0.006	170.50	0.006	366.68	0.006	456.54
0.007	159.91	0.007	314.92	0.007	478.35
0.008	146.20	0.008	253.61	0.008	493.20
0.009	127.81	0.009	182.16	0.009	501.81
0.010	100.00	0.010	100.00	0.010	504.64

For this problem, any finite element mesh would give the exact solution at the nodes. For example, the temperature values at $x/a = 0.2, 0.4, 0.6, 0.8$, and 1.0 are $119, 172, 209, 230$, and 235 , respectively.

Next we consider a square domain, $a = b = 1$ cm, thermal conductivity $k = 18$ W/(m°C), and constant internal heat generation of $Q_0 = 7.2 \times 10^7$ W/m³. The boundaries at $y = 0, b$ and $x = 0$ are maintained at $T_0 = 50^\circ\text{C}$, and the boundary at $x = a$ is exposed to an ambient temperature of $T_c = 100^\circ\text{C}$. The film coefficient is taken to be $h_c = 200$ W/(m²°C). The symmetry about $y = 0.005$ m allows us to use a 10×5 mesh of linear rectangular elements to solve the problem. The results are presented in Table 2.13.5.

The program *FEM2DHT* discussed in Appendix A can be used to solve a variety of linear, two-dimensional heat transfer problems with conduction and convection boundary conditions. For a Fortran source of the program, contact the first author.

Table 2.13.5: The finite element solutions of a two-dimensional heat transfer problem with convective boundary condition.

$r(z = 10^{-3})$	Temp.	$r(z = 10^{-2})$	Temp.	$z(r = 0.0)$	Temp.
0.000	50.000	0.000	50.000	0.000	50.000
0.001	55.768	0.001	61.396	0.001	63.002
0.002	59.303	0.002	69.702	0.002	72.765
0.003	61.717	0.003	75.714	0.003	79.991
0.004	63.416	0.004	80.055	0.004	85.279
0.005	64.623	0.005	83.178	0.005	89.109
0.006	65.481	0.006	85.405	0.006	91.848
0.007	66.083	0.007	86.964	0.007	93.763
0.008	66.500	0.008	88.015	0.008	95.038
0.009	66.796	0.009	88.664	0.009	95.789
0.010	67.135	0.010	88.960	0.010	96.067

Problems

The following problems are designed to test the understanding of the ideas presented in this chapter. Some of the exercise problems are reproduced from the textbook by Reddy [4]. For additional examples and exercise problems, the reader may consult [4].

- 2.1** Develop the weak form of the vector equation (2.2.3) over an element

$$-\nabla \cdot (k\nabla T) = Q \quad \text{in } \Omega \quad (1)$$

subjected to the boundary conditions of the form

$$T = \hat{T} \text{ on } \Gamma_T \quad \text{and} \quad \hat{\mathbf{n}} \cdot (k\nabla T) + q_c = q \text{ on } \Gamma_q \quad (2)$$

- 2.2** Use the weak form given in Eq. (2.9.2) to develop the finite element model of heat transfer problems with convective boundary conditions. Show that the finite element model is of the form

$$(\mathbf{K}^e + \mathbf{H}^e)\mathbf{T}^e = \mathbf{Q}^e + \mathbf{P}^e + \mathbf{q}^e \quad (1)$$

where

$$K_{ij}^e = 2\pi \int_{\Omega^e} \left(k_{rr} \frac{\partial \psi_i^e}{\partial r} \frac{\partial \psi_j^e}{\partial r} + k_{zz} \frac{\partial \psi_i^e}{\partial z} \frac{\partial \psi_j^e}{\partial z} \right) r dr dz \quad (2)$$

$$H_{ij}^e = \oint_{\Gamma^e} h_c \psi_i^e \psi_j^e ds, \quad Q_i^e = 2\pi \int_{\Omega^e} \psi_i^e Q r dr dz \quad (3)$$

$$q_i^e = \oint_{\Gamma^e} q_n \psi_i^e ds, \quad P_i^e = 2\pi \oint_{\Gamma^e} h_c T_c \psi_i^e ds \quad (4)$$

- 2.3** Modify the weak form (2.9.2) of an axisymmetric problem to include convective heat transfer boundary condition, and develop the finite element model. Note that the finite element model will be the same as in Problem 2.2, except for the definitions of \mathbf{H}^e and \mathbf{P}^e .
- 2.4** Evaluate the coefficients H_{ij}^e and P_i^e (due to the convective boundary condition) in Problem 2.2 for the case of a linear triangular element.
- 2.5** Repeat Problem 2.4 for a linear rectangular element.
- 2.6** Solve Example 1 of Section 2.13.1 with the mesh shown in Figure P2.6. Use the element matrices given in Eqs. (2.13.6) and (2.13.9).

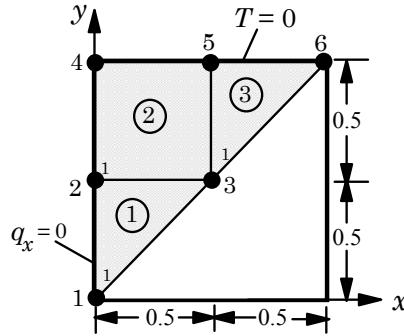


Figure P2.6

- 2.7** Consider steady-state heat conduction in a square region of side $2a$. Assume that the medium has a conductivity of $k = 30 \text{ W}/(\text{m}^\circ\text{C})$ and a uniform heat generation of $Q_0 = 10^7 \text{ W}/\text{m}^3$. For the boundary conditions and mesh shown in Figure P2.7, write the finite element algebraic equations for nodes 1, 3, and 7. Take $h_c = 60 \text{ W}/(\text{m}^2 \circ\text{C})$, $T_c = 0.0^\circ\text{C}$, $T_0 = 100^\circ\text{C}$, $q_0 = 2 \times 10^5 \text{ W}/\text{m}^2$, and $a = 1 \text{ cm}$.

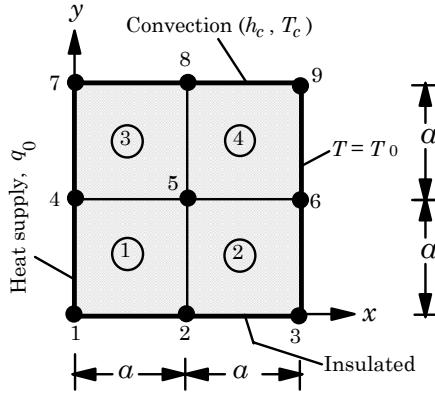


Figure P2.7

- 2.8** Repeat Problem 2.7 for nodes 4, 5, and 9.

- 2.9** For the convection heat transfer problem shown in Figure P2.9, write the four finite element equations for the unknown temperatures. Assume that the thermal conductivity of the material is $k = 5 \text{ W}/(\text{m}^\circ\text{C})$, the convection heat transfer coefficient on the left surface is $h_c = 28 \text{ W}/(\text{m}^2\text{ }^\circ\text{C})$, and the internal heat generation is zero.

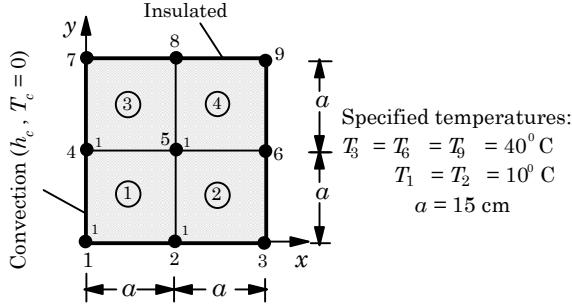


Figure P2.9

- 2.10** The domain shown in Figure P2.10 has its base maintained at 300°C and is exposed to convection on its remaining boundary. Write the finite element equations at nodes 7 and 10. Take $a = 2\text{cm}$, $b = 1\text{cm}$, $h_c = 40 \text{ W}/(\text{m}^2\text{ }^\circ\text{C})$, $T_c = 20^\circ\text{C}$, $k = 5 \text{ W}/(\text{m}^\circ\text{C})$, $T_e = 20^\circ\text{C}$, and $T_0 = 300^\circ\text{C}$.

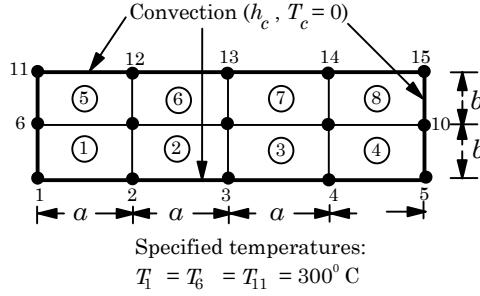


Figure P2.10

- 2.11** Solve the two-dimensional problem in Example 2 of Section 2.13.2 for the case in which the boundary $z = L$ is insulated. Use 5×10 mesh of linear rectangular elements in the full domain.
- 2.12** Repeat Problem 2.11 for the case where the surface $r = R$ is exposed to an ambient temperature of $T_c = 100^\circ\text{C}$. Take the film coefficient to be $h_c = 200 \text{ W}/(\text{m}^2 \text{ }^\circ\text{C})$.
- 2.13** (*Galerkin time approximation scheme [8,11]*) Consider the differential equation

$$a \frac{du}{dt} + bu = f(t), \quad t_n < t < t_{n+1} \quad (1)$$

where a and b are constants and f is a function of time t . Using linear approximation of the form

$$u(t) = u_n \psi_n(t) + u_{n+1} \psi_{n+1}(t) \quad (2)$$

$$\psi_n(t) = \frac{(t_{n+1} - t)}{\Delta t}, \quad \psi_{n+1}(t) = \frac{(t - t_n)}{\Delta t} \quad (3)$$

derive the Galerkin finite element equations for a typical time element in interval (t_n, t_{n+1}) . Assuming that u_n is known, compute the solution u_{n+1} from the element equations, and then

compare the result with that obtained with the α -family of approximation in Eqs. (2.8.9) and (2.8.10).

- 2.14** Repeat Problem 2.13 for the case of quadratic interpolation between times, (t_{n-1}, t_n, t_{n+1}) . Solve the third equation of the resulting finite element equations for u_{n+1} in terms of u_{n-1} and u_n and time step Δt .
- 2.15** (*Collocation time approximation*) Consider the differential equation in time,

$$a \frac{du}{dt} + bu = f, \quad t_n < t < t_{n+1} \quad (1)$$

where a and b are constants and f is a function of time. Using linear approximation of the form $u(t) = u_n\psi_n(t) + u_{n+1}\psi_{n+1}(t)$, where $\psi_n(t) = (t_{n+1}-t)/\Delta t$ and $\psi_{n+1}(t) = (t-t_n)/\Delta t$, derive the one-point collocation finite element equations (i.e., set $w = \delta(t - t_0)$, where $\delta(\cdot)$ denotes the Dirac delta function, in Eq. (1) of Problem 2.13) for a typical time element, (t_n, t_{n+1}) , and compare the result with that obtained with the α -family of approximation. Use the collocation point to be $t_0 = (1 - \alpha)t_n + \alpha t_{n+1}$.

References for Additional Reading

1. S. G. Mikhlin, *Variational Methods in Mathematical Physics*, (translated from the Russian by T. Boddington), Pergamon Press, Oxford (1964).
2. S. G. Mikhlin, *The Numerical Performance of Variational Methods*, (translated from the Russian by R. S. Anderssen), Wolters-Noordhoff, The Netherlands (1971).
3. J. N. Reddy, *Energy Principles and Variational Methods in Applied Mechanics*, 2nd ed., John Wiley & Sons, New York (2002).
4. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).
5. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis*, Oxford University Press, Oxford (2004).
6. J. N. Reddy, *Applied Functional Analysis and Variational Methods in Engineering*, McGraw-Hill, New York (1986); reprinted by Krieger Publishing, Melbourne, Florida (1991).
7. J. N. Reddy and M. L. Rasmussen, *Advanced Engineering Analysis*, John Wiley & Sons, New York (1982); reprinted by Krieger Publishing, Melbourne, Florida (1990).
8. J. H. Argyris and O. W. Scharpf, “Finite Elements in Time and Space,” *Aeronautical Journal of the Royal Society*, **73**, 1041–1044 (1969).
9. K. Rektorys, *Variational Methods in Mathematics, Science and Engineering*, D. Reidel Publishing Co., Boston, MA (1980).
10. R. J. Sobey, “Hermitian Space-Time Finite Elements for Estuarine Mass Transport,” *International Journal for Numerical Methods in Fluids*, **2**, 277–297 (1982).
11. W. L. Wood, “Control of Crank–Nicolson Noise in the Numerical Solution of the Heat Conduction Equation,” *International Journal for Numerical Methods in Engineering*, **11**, 1059–1065 (1977).
12. J. P. Pontaza and J. N. Reddy, “Space-Time Coupled Spectral/hp Least-Squares Finite Element Formulation for the Incompressible Navier–Stokes Equations,” *Journal of Computational Physics*, **197**(2), 418–459 (2004).

3

Conduction Heat Transfer

3.1 Introduction

In this chapter we present the finite element analysis of 3D conduction heat transfer and related diffusion problems. Though conduction is the least complex mode of thermal energy transport, it plays a major role in industrial design, manufacturing and engineering. The prediction of temperature distributions, thermal performance and thermal stresses in complex geometries has become a routine part of engineering analysis, due in large part to the finite element method and availability of general purpose computer programs. The conduction problem also provides a convenient framework to discuss various advanced aspects of the finite element method and numerical algorithms. The introduction to the finite element method presented in Chapter 2 aids the development presented here for the general 3D heat conduction problem.

As described in the previous chapters, the appropriate mathematical description of thermal conduction in a 3D solid (or stationary fluid) region, Ω , is given by Eq. (1.5.4). In the interest of brevity, the subscript “ s ,” which refers to a solid region, is omitted throughout the chapter. The Cartesian component form of the equation is [see Eq.(1.5.10)]

$$\rho C \frac{\partial T}{\partial t} = \frac{\partial}{\partial x_i} \left(k_{ij} \frac{\partial T}{\partial x_j} \right) + Q \quad (3.1.1)$$

where T is the temperature, ρ is the density, C is the specific heat, k_{ij} are the Cartesian components of the conductivity tensor (symmetric), and Q is the internal heat generation per unit volume. Summation on repeated subscripts is assumed (i.e., i and j are summed over the range of $i, j = 1, 2, 3$). All of the variables may be functions of position $\mathbf{x} = (x_1, x_2, x_3)$ and time t . We wish to solve Eq. (3.1.1) under appropriate boundary and initial conditions. The boundary conditions for this equation are given by the equations [see Eq. (1.10.10a,b)]:

$$T = f^T(s_k, t) \quad \text{on } \Gamma_T, \quad \text{for } t > 0 \quad (3.1.2a)$$

$$-\left(k_{ij} \frac{\partial T}{\partial x_j} \right) n_i = q_a + q_c + q_r = f^q(s_k, t) \quad \text{on } \Gamma_q, \quad \text{for } t > 0 \quad (3.1.2b)$$

Here s_k denotes the coordinates of a point on the boundary surface, q_a is an applied flux and the convective and radiative flux are

$$q_c = h_c(s_k, T, t)(T - T_c), \quad q_r = h_r(s_k, T, t)(T - T_r) \quad (3.1.2c)$$

The initial condition on the temperature is given by

$$T(x_j, 0) = T_0(x_j) \quad (3.1.3)$$

The heat transfer problem described by Eqs. (3.1.1)–(3.1.3) differs from that discussed in Chapter 2 in several respects. First, the problem considered here is for general three-dimensional geometries. Second, the boundary conditions are general enough to include conduction, convection, and radiation heat transfer between the solid regions and the surrounding environment. Third, the material coefficients, ρ , C , k_{ij} , h_c , and h_r can be functions of temperature, making the problem a nonlinear one.

As described in Section 2.8, the finite element model of the initial-boundary value problem described by Eqs. (3.1.1)–(3.1.3) is developed in two steps: (1) *spatial discretization*, in which the weak form of Eq. (3.1.1) over a typical element is developed and the spatial approximation of the dependent variable (i.e., temperature) T of the problem is assumed to obtain a set of ordinary differential equations in time among the nodal values T_j^e of the dependent variable; (2) *temporal approximation* of the ordinary differential equations obtained in the first step is carried out using some appropriate method such as a finite difference approximation. This step leads to a set of algebraic equations involving the nodal values T_j^e at time $t_{n+1} [= (n + 1)\Delta t$, where n is an integer and Δt is the time increment] in terms of known values from the previous time step. Note that time-independent (steady) boundary value problems replace the second or temporal approximation step with the invocation of some type of iterative algorithm depending on the nonlinearity of the application. See Section 2.8 for an account of these two steps when applied to a two-dimensional problem. Here we develop the finite element models of three-dimensional, transient heat conduction problems using the two-step procedure described above. We begin with the development of the semidiscrete finite element model of Eqs. (3.1.1) and (3.1.2).

3.2 Semidiscrete Finite Element Model

Suppose that the conduction heat transfer region Ω is discretized into an appropriate collection of finite elements (see Figure 3.2.1). The weak form of Eqs. (3.1.1) and (3.1.2) over a typical element Ω^e is obtained by multiplying Eq. (3.1.1) with the weight function $w(\mathbf{x})$ and integrating over the element, integrating-by-parts (spatially) those terms which involve higher-order derivatives, and replacing the coefficient of the weight function in the boundary integral with the secondary variable [i.e., use Eq. (3.1.2b)]. We obtain

$$0 = \int_{\Omega^e} \left[w \left(\rho C \frac{\partial T}{\partial t} - Q \right) + k_{ij} \frac{\partial w}{\partial x_i} \frac{\partial T}{\partial x_j} \right] d\mathbf{x} + \oint_{\Gamma^e} (q_a + q_c + q_r) w \, ds \quad (3.2.1)$$

where summation on repeated subscripts, i and j , is used. Note that no integration by parts with respect to time is used, and w is not a function of time.

The semidiscrete finite element model is obtained from Eq. (3.2.1) by substituting a finite element approximation for the dependent variable, T . In selecting the approximation for T , we assume that the time dependence can be separated from the spatial variation,

$$T(\mathbf{x}, t) \simeq \sum_{i=1}^{n_e} T_i^e(t) \psi_i^e(\mathbf{x}) \quad (3.2.2a)$$

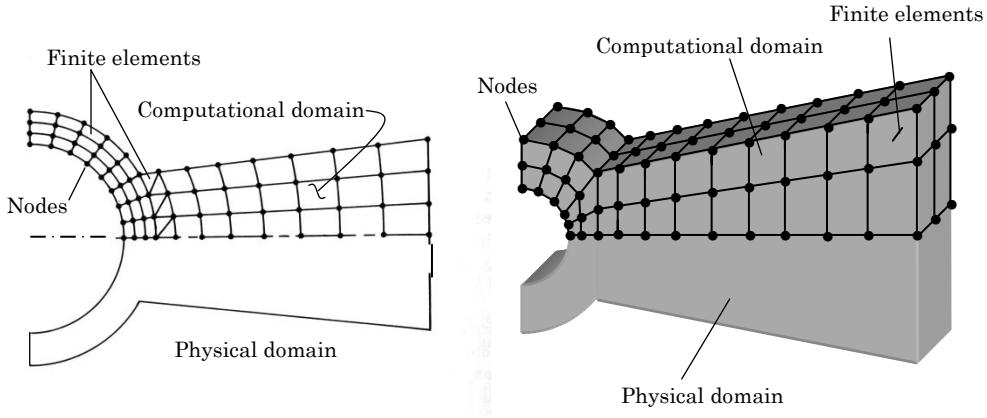


Figure 3.2.1: Finite element discretization of a (a) two-dimensional domain, and (b) three-dimensional domain.

which can be written in matrix notation as

$$T(\mathbf{x}, t) = (\Psi^e)^T \mathbf{T}^e \quad (3.2.2b)$$

where $(\Psi^e)^T$ is a $(1 \times n_e)$ vector of interpolation functions of the element Ω^e

$$(\Psi^e)^T = \{\psi_1^e \ \psi_2^e \ \dots \ \psi_{n_e}^e\} \quad (3.2.2c)$$

\mathbf{T}^e is a $(n_e \times 1)$ vector of nodal temperatures, superscript $(\cdot)^T$ denotes the transpose of a vector or matrix, and n_e is the number of nodal points in an element. The i th differential equation (in time) of the semidiscrete, weak-form Galerkin finite element model is obtained by substituting $w(\mathbf{x}) = \psi_i^e(\mathbf{x})$ ($i = 1, 2, \dots, n_e$) and replacing T by the interpolation in Eq. (3.2.2a):

$$0 = \sum_{j=1}^{n_e} \left(M_{ij}^e \frac{dT_j^e}{dt} + K_{ij}^e T_j^e \right) - Q_i^e + q_i^e \quad (3.2.3)$$

Equation (3.2.3) can be expressed in matrix form as

$$\mathbf{M}^e(\mathbf{T}^e) \dot{\mathbf{T}}^e + \mathbf{K}^e(\mathbf{T}^e) \mathbf{T}^e = \mathbf{Q}^e(\mathbf{T}^e) - \mathbf{q}^e(\mathbf{T}^e) \quad (3.2.4)$$

where a superposed dot on T denotes a derivative with time $\dot{\mathbf{T}} = \partial \mathbf{T} / \partial t$, and

$$M_{ij}^e = \int_{\Omega^e} \rho C \psi_i \psi_j \, d\mathbf{x}, \quad K_{ij}^e = \int_{\Omega^e} k_{mn} \frac{\partial \psi_i}{\partial x_m} \frac{\partial \psi_j}{\partial x_n} \, d\mathbf{x} \quad (3.2.5a)$$

$$Q_i^e = \int_{\Omega^e} \psi_i Q(\mathbf{x}, t) \, d\mathbf{x}, \quad q_i^e = \oint_{\Gamma^e} \psi_i (q_a + q_c + q_r) \, ds \quad (3.2.5b)$$

where summation on repeated indices ($m, n = 1, 2, 3$) is assumed, and the element label “ e ” on ψ is omitted for brevity. The element coefficient matrices and vectors

are written in the vector form as

$$\begin{aligned}\mathbf{M}^e &= \int_{\Omega^e} \rho C \Psi \Psi^T d\mathbf{x} \\ \mathbf{K}^e &= \int_{\Omega^e} \frac{\partial \Psi}{\partial x_m} k_{mn} \frac{\partial \Psi^T}{\partial x_n} d\mathbf{x} \\ \mathbf{Q}^e &= \int_{\Omega^e} \Psi Q d\mathbf{x} \\ \mathbf{q}^e &= \oint_{\Gamma^e} \Psi (q_a + q_c + q_r) d\mathbf{s}\end{aligned}\quad (3.2.6)$$

Equation (3.2.4) is written in a form to show the most general nonlinear case in which material properties, boundary conditions, and volumetric sources depend on temperature. The implementation of variable material properties and coefficients will be discussed in a subsequent section.

The process of assembly for two-dimensional problems was discussed in Chapter 2; the assembly process for three-dimensional geometries is completely analogous. The assembled matrices are symbolically represented as

$$\mathbf{M} = \sum_e \mathbf{M}^e; \quad \mathbf{K} = \sum_e \mathbf{K}^e; \quad \mathbf{F} = \sum_e \mathbf{F}^e, \quad \mathbf{F}^e = \mathbf{Q}^e + \mathbf{q}^e \quad (3.2.7)$$

In Eq. (3.2.7) the sum is taken over all elements in the mesh, and the element matrices are defined by Eq. (3.2.6). Once the form of the element interpolation functions Ψ^e is known and the element geometry is specified, the integrals in Eq. (3.2.6) can be evaluated, and the global (assembled) equations are then constructed through use of Eq. (3.2.7):

$$\mathbf{M}(\mathbf{T}) \dot{\mathbf{T}} + \mathbf{K}(\mathbf{T}) \mathbf{T} = \mathbf{F}(\mathbf{T}) \quad (3.2.8)$$

Before we consider the time or temporal approximation of Eqs. (3.2.4) or Eq. (3.2.8), it is informative to discuss the geometries of elements and interpolation functions used in Eq. (3.2.2a–c) and the numerical evaluation of the coefficients defined in Eq. (3.2.5) or (3.2.6). We will return to Eq. (3.2.8) in Section 3.7 to discuss its solution for steady-state and transient problems.

3.3 Interpolation Functions

3.3.1 Preliminary Comments

The equations for an individual element, as indicated by Eq. (3.2.6), require the specification of the element interpolation functions $\psi_i^e(\mathbf{x})$ for the approximation of the temperature and a description of the element geometry. In practical applications, elements based on linear and quadratic polynomials are generally used. The use of an isoparametric formulation is also standard in practice; this allows complex geometric regions to be easily and accurately described. Two-dimensional linear and quadratic elements were presented in Chapter 2. In this section, a number of commonly used three-dimensional elements are presented (see also [1–5]). Some specialty elements are described later in this chapter.

3.3.2 Hexahedral (Brick) Elements

Brick elements represent the most commonly used finite elements for three-dimensional analysis, and the straight-sided, linear eight-node brick element is the most cost-effective choice. The interpolation functions of the linear element (see Figure 3.3.1) are given in terms of the normalized coordinates (ξ, η, ζ) as follows:

$$\{\Psi^e\} = \frac{1}{8} \begin{Bmatrix} (1 - \xi)(1 - \eta)(1 - \zeta) \\ (1 + \xi)(1 - \eta)(1 - \zeta) \\ (1 + \xi)(1 + \eta)(1 - \zeta) \\ (1 - \xi)(1 + \eta)(1 - \zeta) \\ (1 - \xi)(1 - \eta)(1 + \zeta) \\ (1 + \xi)(1 - \eta)(1 + \zeta) \\ (1 + \xi)(1 + \eta)(1 + \zeta) \\ (1 - \xi)(1 + \eta)(1 + \zeta) \end{Bmatrix} \quad (3.3.1)$$

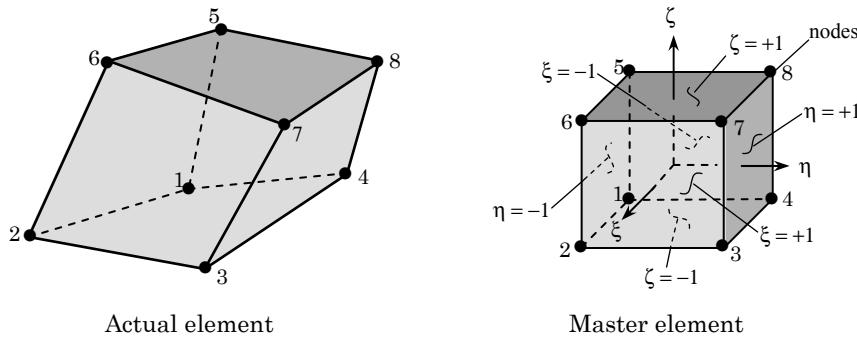


Figure 3.3.1: The linear (eight-node) brick element.

The quadratic shape functions for the twenty-node serendipity element (see Figure 3.3.2) are given in Eq. (3.3.2).

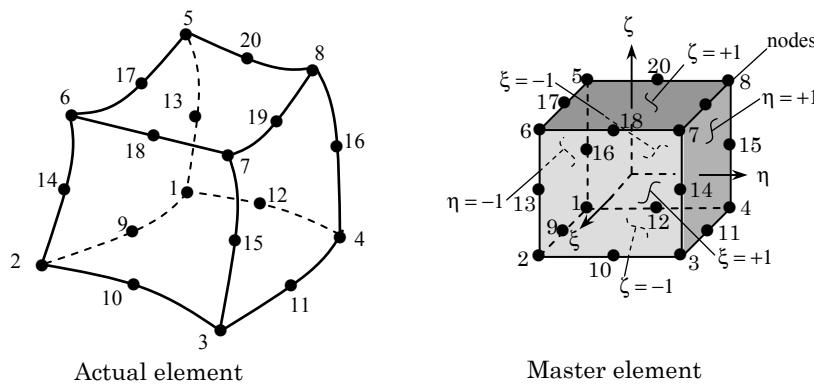


Figure 3.3.2: The (twenty-node) quadratic brick element.

$$\{\Psi^e\} = \frac{1}{8} \left\{ \begin{array}{l} (1-\xi)(1-\eta)(1-\zeta)(-\xi-\eta-\zeta-2) \\ (1+\xi)(1-\eta)(1-\zeta)(\xi-\eta-\zeta-2) \\ (1+\xi)(1+\eta)(1-\zeta)(\xi+\eta-\zeta-2) \\ (1-\xi)(1+\eta)(1-\zeta)(-\xi+\eta-\zeta-2) \\ (1-\xi)(1-\eta)(1+\zeta)(-\xi-\eta+\zeta-2) \\ (1+\xi)(1-\eta)(1+\zeta)(\xi-\eta+\zeta-2) \\ (1+\xi)(1+\eta)(1+\zeta)(\xi+\eta+\zeta-2) \\ (1-\xi)(1+\eta)(1+\zeta)(-\xi+\eta+\zeta-2) \\ 2(1-\xi^2)(1-\eta)(1-\zeta) \\ 2(1+\xi)(1-\eta^2)(1-\zeta) \\ 2(1-\xi^2)(1+\eta)(1-\zeta) \\ 2(1-\xi)(1-\eta^2)(1-\zeta) \\ 2(1-\xi)(1-\eta)(1-\zeta^2) \\ 2(1+\xi)(1-\eta)(1-\zeta^2) \\ 2(1+\xi)(1+\eta)(1-\zeta^2) \\ 2(1-\xi)(1+\eta)(1-\zeta^2) \\ 2(1-\xi^2)(1-\eta)(1+\zeta) \\ 2(1+\xi)(1-\eta^2)(1+\zeta) \\ 2(1-\xi^2)(1+\eta)(1+\zeta) \\ 2(1-\xi)(1-\eta^2)(1+\zeta) \end{array} \right\} \quad (3.3.2)$$

As in the case of the two-dimensional quadratic (nine-node) Lagrange element, a brick element with 27 nodes may also be constructed; the shape functions are not shown for this element.

The transformation between the actual element Ω^e and the master element $\hat{\Omega}$ [or equivalently, between $(x_1, x_2, x_3) = (x, y, z)$ and (ξ, η, ζ)] is accomplished by a coordinate transformation of the form

$$x = \sum_{i=1}^{n_g} x_i^e \phi_i^e(\xi, \eta, \zeta), \quad y = \sum_{i=1}^{n_g} y_i^e \phi_i^e(\xi, \eta, \zeta), \quad z = \sum_{i=1}^{n_g} z_i^e \phi_i^e(\xi, \eta, \zeta) \quad (3.3.3)$$

where n_g is the number of nodes describing the geometry of the element. If $n_g < n_e$, where n_e is the number of nodes used to interpolate the solution [see Eq. (3.2.2a)], the element is termed *subparametric*; when $n_g > n_e$ the element is called *superparametric*. The most common case, $n_g = n_e$ and $\phi_j^e = \psi_j^e$ is known as the *isoparametric formulation*. It is not uncommon to use, for example, subparametric formulation with respect to one set of dependent variables and isoparametric formulation with respect to another set (e.g., classical beam and plate bending elements; see [2-5]).

3.3.3 Tetrahedral Elements

The standard tetrahedral elements are a three-dimensional version of the triangular elements. The four-node linear and ten-node quadratic tetrahedral elements are shown in Figure 3.3.3. The volume coordinates, L_i , are used to describe the interpolation functions for linear and quadratic elements, where $L_1+L_2+L_3+L_4 = 1$. The interpolation functions are

$$\{\Psi^e\} = \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \end{Bmatrix}; \quad \{\Psi^e\} = \begin{Bmatrix} L_1(2L_1 - 1) \\ L_2(2L_2 - 1) \\ L_3(2L_3 - 1) \\ L_4(2L_4 - 1) \\ 4L_1L_2 \\ 4L_2L_3 \\ 4L_3L_1 \\ 4L_1L_4 \\ 4L_1L_4 \\ 4L_2L_4 \\ 4L_3L_4 \end{Bmatrix} \quad (3.3.4)$$

The parametric mapping in Eq. (3.3.3) is easily defined for a master element. Note that other higher-order tetrahedrons may be defined that have nodes at the element centroid and the center of each triangular face.

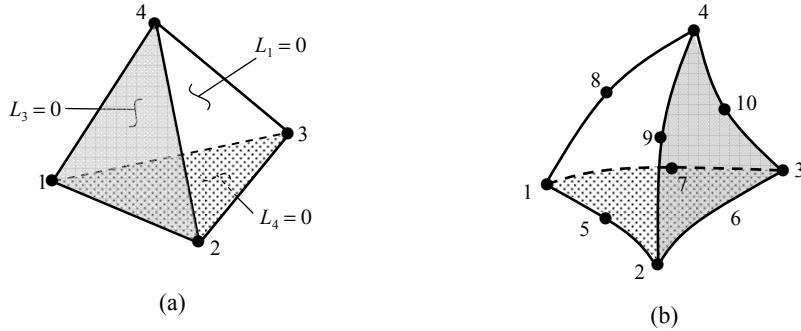


Figure 3.3.3: Linear and quadratic tetrahedral elements.

3.3.4 Prism Elements

A prism or wedge element is often useful in three-dimensional geometries especially for transitioning between hexahedral and tetrahedral elements. The shape functions for the six-node linear element (see Figure 3.3.4a) are given by

$$\{\Psi^e\} = \frac{1}{2} \begin{Bmatrix} L_1(1 - \zeta) \\ L_2(1 - \zeta) \\ L_3(1 - \zeta) \\ L_1(1 + \zeta) \\ L_2(1 + \zeta) \\ L_3(1 + \zeta) \end{Bmatrix} \quad (3.3.5)$$

The interpolation functions for the fifteen-node quadratic prism element (see Figure 3.3.4b) are

$$\{\Psi^e\} = \frac{1}{2} \left\{ \begin{array}{l} L_1[(2L_1 - 1)(1 - \zeta) - (1 - \zeta^2)] \\ L_2[(2L_2 - 1)(1 - \zeta) - (1 - \zeta^2)] \\ L_3[(2L_3 - 1)(1 - \zeta) - (1 - \zeta^2)] \\ L_1[(2L_1 - 1)(1 + \zeta) - (1 - \zeta^2)] \\ L_2[(2L_2 - 1)(1 + \zeta) - (1 - \zeta^2)] \\ L_3[(2L_3 - 1)(1 + \zeta) - (1 - \zeta^2)] \\ 4L_1L_2(1 - \zeta) \\ 4L_2L_3(1 - \zeta) \\ 4L_3L_1(1 - \zeta) \\ 2L_1(1 - \zeta^2) \\ 2L_2(1 - \zeta^2) \\ 2L_3(1 - \zeta^2) \\ 4L_1L_2(1 + \zeta) \\ 4L_2L_3(1 + \zeta) \\ 4L_3L_1(1 + \zeta) \end{array} \right\} \quad (3.3.6)$$

The area coordinates, L_i (see Section 2.10), are used to describe the functional variation in the triangular cross section of the prism elements, while a standard normalized coordinate, ζ , describes the variation in the axial direction. Note that $L_1 + L_2 + L_3 = 1$. A coordinate transformation of the form given in (3.3.3) is easily defined to map the actual prism into a master element.

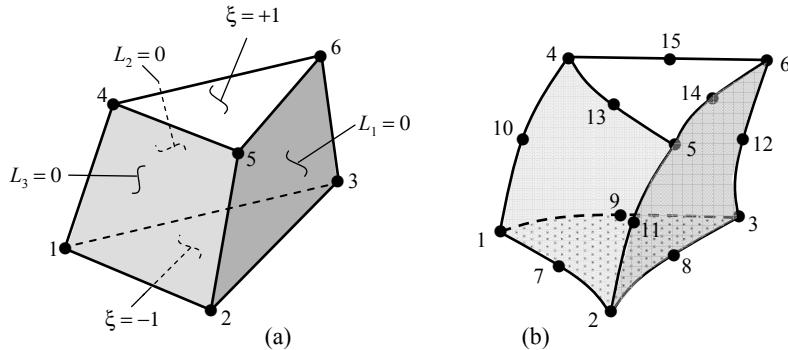


Figure 3.3.4: (a) The six-node linear prism element. (b) The fifteen-node quadratic prism element.

3.3.5 Pyramid Elements

Another element that may be used for transition between tetrahedral and hexahedral elements is the pyramid element with four triangular faces and one quadrilateral face. This is a relatively uncommon element because of the difficulty in specifying conformal shape functions for the pyramid. The use of both rational functions and degenerate (hexahedral) elements has been proposed for developing the appropriate pyramid shape functions; both of these techniques lead to elements with significant computational drawbacks. Recent work by Bluck and Walker [6] has led to the

development of polynomial basis functions of arbitrary order for pyramids. As no significant utilization of these elements has been demonstrated, pyramid elements will not be discussed any further. Prism elements remain the preferred elements for transition applications.

3.4 Numerical Integration

The evaluation of the various finite element coefficient matrices in (3.2.5) requires the integration of combinations of the interpolation functions and their spatial derivatives over the area or volume of the element. The numerical integration ideas described in Section 2.11 can be easily extended to three dimensions. Here we give the pertinent equations for the three-dimensional case.

For an isoparametric formulation [2-4], the following relations, based on the chain rule of differentiation, can be derived for brick elements:

$$\begin{Bmatrix} \frac{\partial \psi_i}{\partial \xi} \\ \frac{\partial \psi_i}{\partial \eta} \\ \frac{\partial \psi_i}{\partial \zeta} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \begin{Bmatrix} \frac{\partial \psi_i}{\partial x} \\ \frac{\partial \psi_i}{\partial y} \\ \frac{\partial \psi_i}{\partial z} \end{Bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \begin{Bmatrix} \frac{\partial \psi_i}{\partial x} \\ \frac{\partial \psi_i}{\partial y} \\ \frac{\partial \psi_i}{\partial z} \end{Bmatrix} \quad (3.4.1)$$

where $[J]$ is the Jacobian of the transformation from global coordinates (x, y, z) to the local element coordinates (ξ, η, ζ) . The parametric transformation defined in Eq. (3.3.3) can be used to define the components of $[J]$. Inverting Eq. (3.4.1), we obtain the global spatial derivatives of the interpolation functions in terms of the local derivatives:

$$\begin{Bmatrix} \frac{\partial \psi_i}{\partial x} \\ \frac{\partial \psi_i}{\partial y} \\ \frac{\partial \psi_i}{\partial z} \end{Bmatrix} = \begin{bmatrix} J_{11}^* & J_{12}^* & J_{13}^* \\ J_{21}^* & J_{22}^* & J_{23}^* \\ J_{31}^* & J_{32}^* & J_{33}^* \end{bmatrix} \begin{Bmatrix} \frac{\partial \psi_i}{\partial \xi} \\ \frac{\partial \psi_i}{\partial \eta} \\ \frac{\partial \psi_i}{\partial \zeta} \end{Bmatrix} \quad (3.4.2)$$

where \mathbf{J}^* is the inverse of the Jacobian matrix \mathbf{J} . The components J_{ij}^* are complicated functions of the components of \mathbf{J} that can in principle be obtained by analytically inverting the 3×3 Jacobian matrix. In practice, the Jacobian is usually inverted numerically at each integration point.

In performing numerical integration over the element volume, it is necessary to transform the integrand and limits of integration from the global coordinates to the local element coordinates. The differential elemental volume transforms according to

$$d\mathbf{x} = dx \ dy \ dz = J \ d\xi \ d\eta \ d\zeta \quad (3.4.3)$$

The integration limits for the integrals transform to the limits on the local coordinates (ξ, η, ζ) , i.e., -1 to $+1$. In the previous equations the (ξ, η, ζ) coordinates for a brick element were used for purposes of explanation. Similar relations for a tetrahedral element can be derived by replacing (ξ, η, ζ) with (L_1, L_2, L_3) . The variable L_4 does not enter the formulae due to the relation $L_1 + L_2 + L_3 + L_4 = 1$. Hybrid coordinates, such as those utilized in the prism element, are treated in an analogous manner [2].

To illustrate further the development of a computational form for typical entries in Eq. (3.2.5) or (3.2.6), we consider the cross (xy) component in the (1, 2) coefficient of the diffusion matrix \mathbf{K}^e of Eq. (3.2.6):

$$(K_{12}^e)_{xy} = \int_{\Omega^e} k_{xy} \frac{\partial \psi_1}{\partial x} \frac{\partial \psi_2}{\partial y} dx dy dz \quad (3.4.4)$$

which will be evaluated for a three-dimensional, brick element. From the definitions in Eqs. (3.4.2) and (3.4.3) then

$$\begin{aligned} (K_{12}^e)_{xy} &= \int_{-1}^{+1} \int_{-1}^{+1} \int_{-1}^{+1} k_{xy} \left(J_{11}^* \frac{\partial \psi_1}{\partial \xi} + J_{12}^* \frac{\partial \psi_1}{\partial \eta} + J_{13}^* \frac{\partial \psi_1}{\partial \zeta} \right) \times \\ &\quad \left(J_{21}^* \frac{\partial \psi_2}{\partial \xi} + J_{22}^* \frac{\partial \psi_2}{\partial \eta} + J_{23}^* \frac{\partial \psi_2}{\partial \zeta} \right) |J| d\xi d\eta d\zeta \end{aligned} \quad (3.4.5)$$

Note that since the thermal conductivity could vary over the element, it must be treated as a function of (ξ, η, ζ) and be included in the evaluation of the integral. Variable coefficients are discussed further in Section 3.9. The above coefficient is typical of those encountered in building finite element equations. Each component of the integral in Eq. (3.4.5) is of the form

$$I = \int_{-1}^{+1} \int_{-1}^{+1} \int_{-1}^{+1} F(\xi, \eta, \zeta) d\xi d\eta d\zeta \quad (3.4.6)$$

The Gauss quadrature formula (2.11.12) discussed in Section 2.11 can be readily extended to three dimensions to evaluate the integral expression I :

$$\int_{-1}^{+1} \int_{-1}^{+1} \int_{-1}^{+1} F(\xi, \eta, \zeta) d\xi d\eta d\zeta \approx \sum_{I=1}^M \sum_{J=1}^N \sum_{K=1}^P F(\xi_I, \eta_J, \zeta_K) W_I W_J W_K \quad (3.4.7)$$

For brick elements it is typical to employ a product rule as shown in (3.4.7) with $M = N = P = 2$ or 3. Other element types (e.g., tetrahedral elements) are also evaluated using quadrature formulas similar to (3.4.7) but slightly different in form [see Eq. (2.11.15)]. Hybrid elements, such as the prism, generally utilize combination quadrature formulas such as (2.11.15) in the triangular plane and a one-dimensional Gauss formula along the axis. For additional discussion of quadrature rules and their utilization, the reader may consult [2, 3, 5] and [8–14].

3.5 Computation of Surface Flux

The nodal flux q_i^e due to applied boundary conditions need only be computed for those element sides that coincide with the boundary of the problem domain; contributions from interior element boundaries are cancelled by adjoining elements. The surface flux vector is given by

$$\mathbf{q}^e = \oint_{\Gamma^e} \Psi q_i n_i ds = \oint_{\Gamma^e} \Psi(q_a + q_c + q_r) ds \quad (3.5.1)$$

where Γ^e denotes the boundary surface of the element and $q_i n_i$ (sum on i) is the total heat flux normal to the surface.

The computation of the indicated surface integrals is most easily carried out in the normalized or natural coordinate system for the face (edge) of an element. This requires that the elemental surface area ds , or element edge length ds , be related to the local surface coordinates. Consider the typical quadrilateral element face shown in Figure 3.5.1, where the vectors \mathbf{e}_1 and \mathbf{e}_2 are defined as being tangent to the curvilinear coordinates, ξ_s and η_s . The \mathbf{e} vectors are not necessarily unit vectors; the ξ_s and η_s coordinates are assumed to be the natural coordinates for the element face. The elemental area ds in terms of the global coordinates (x, y, z) is related to an elemental area in surface coordinates by

$$ds = |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.5.2)$$

where \mathbf{J}_s is the Jacobian of the coordinate transformation and $|\cdot|$ indicates the determinant. The determinant of the Jacobian can be written in terms of the vectors $(\mathbf{e}_1, \mathbf{e}_2)$ as

$$|\mathbf{J}_s| = |\mathbf{e}_1 \times \mathbf{e}_2| = [(\mathbf{e}_1 \cdot \mathbf{e}_1)(\mathbf{e}_2 \cdot \mathbf{e}_2) - (\mathbf{e}_1 \cdot \mathbf{e}_2)^2]^{\frac{1}{2}} \quad (3.5.3)$$

The vectors \mathbf{e}_i can be expressed in terms of the global coordinates by

$$\mathbf{e}_1 = \left\{ \begin{array}{l} \frac{\partial x}{\partial \xi_s} \\ \frac{\partial y}{\partial \xi_s} \\ \frac{\partial z}{\partial \xi_s} \end{array} \right\}, \quad \mathbf{e}_2 = \left\{ \begin{array}{l} \frac{\partial x}{\partial \eta_s} \\ \frac{\partial y}{\partial \eta_s} \\ \frac{\partial z}{\partial \eta_s} \end{array} \right\} \quad (3.5.4)$$

Using the transformation (3.3.3) in Eq. (3.5.4), we can write

$$\mathbf{e}_1 = \left\{ \begin{array}{l} \sum_{i=1}^{n_s} x_i^e \frac{\partial \hat{\psi}_i^e}{\partial \xi_s} \\ \sum_{i=1}^{n_s} y_i^e \frac{\partial \hat{\psi}_i^e}{\partial \xi_s} \\ \sum_{i=1}^{n_s} z_i^e \frac{\partial \hat{\psi}_i^e}{\partial \xi_s} \end{array} \right\}; \quad \mathbf{e}_2 = \left\{ \begin{array}{l} \sum_{i=1}^{n_s} x_i^e \frac{\partial \hat{\psi}_i^e}{\partial \eta_s} \\ \sum_{i=1}^{n_s} y_i^e \frac{\partial \hat{\psi}_i^e}{\partial \eta_s} \\ \sum_{i=1}^{n_s} z_i^e \frac{\partial \hat{\psi}_i^e}{\partial \eta_s} \end{array} \right\} \quad (3.5.5)$$

where the $\hat{\cdot}$ indicates the restriction of the interpolation functions to an element face and n_s is the number of nodes defining the surface. The functions $\hat{\psi}_i$ may be either linear or quadratic, depending on the type of mapping used to describe the element geometry. Equation (3.5.5) provides a means for computing $|\mathbf{J}_s|$, which can be used in Eq. (3.5.2).

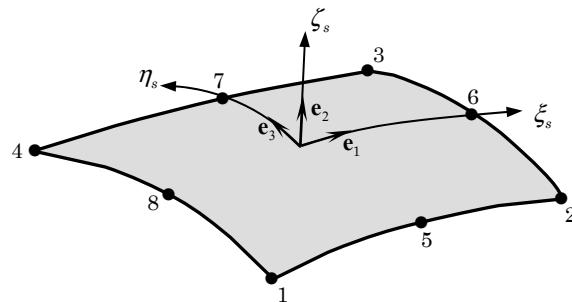


Figure 3.5.1: Nomenclature for surface flux computation.

With the above definitions, the terms in Eq. (3.5.1) can be considered individually. For the case of an applied heat flux q_a , the contribution to \mathbf{q}^e is given by

$$\mathbf{q}_a = \oint_{\Gamma_e} \hat{\Psi} q_a \, d\mathbf{s} \quad (3.5.6)$$

where $\hat{\Psi}$ is the vector of interpolation functions that are restricted to the element boundary. Employing Eq. (3.5.2), Eq. (3.5.6) can be written in a more explicit form

$$\mathbf{q}_a = \int_{-1}^1 \int_{-1}^1 \hat{\Psi}(\xi, \eta) q_a(\xi, \eta) |\mathbf{J}_s| \, d\xi_s \, d\eta_s \quad (3.5.7)$$

Once the distribution of the normal flux $q_a(\xi, \eta)$ is known for the element, Eq. (3.5.7) can be evaluated using numerical quadrature as discussed in Section 2.11. The order of the quadrature used to evaluate \mathbf{q}_a is typically the same as that used on the volume integrals defining the conductivity [K^e] and capacitance [M^e] matrices. For a three-dimensional element, the surface shape functions $\hat{\Psi}$ correspond directly to the interpolation functions of the two-dimensional triangular and quadrilateral elements. In the two-dimensional case, $\hat{\Psi}$ corresponds to the one-dimensional interpolation functions

$$\{\hat{\Psi}^e\} = \frac{1}{2} \begin{Bmatrix} (1-\xi) \\ (1+\xi) \end{Bmatrix} \quad (3.5.8)$$

for the linear case and

$$\{\hat{\Psi}^e\} = \frac{1}{2} \begin{Bmatrix} (1-\xi)\xi \\ 2(1-\xi^2) \\ (1+\xi)\xi \end{Bmatrix} \quad (3.5.9)$$

for the quadratic case, where ξ is the normalized coordinate along the element edge. Also, for the two-dimensional case, the integral in Eq. (3.5.7) simplifies to an integral over the edge, and the definition of the incremental edge length reduces to

$$ds = |\mathbf{J}_s| d\xi_s = \Delta d\xi_s \quad (3.5.10)$$

where

$$\Delta = \left[\left(\frac{\partial x}{\partial \xi_s} \right)^2 + \left(\frac{\partial y}{\partial \xi_s} \right)^2 \right]^{\frac{1}{2}} \quad (3.5.11)$$

The contributions to \mathbf{q}^e due to convection and radiation are treated in an analogous manner. For the convection boundary condition, we have

$$\mathbf{q}_c = \oint_{\Gamma_e} \hat{\Psi} q_c \, d\mathbf{s} = \oint_{\Gamma_e} \hat{\Psi} h_c(T - T_c) \, d\mathbf{s} \quad (3.5.12)$$

where the definition in Eq. (3.1.2c) has been used. The temperature on the element surface is given by

$$T(\xi, \eta) = (\hat{\Psi}(\xi, \eta))^T \mathbf{T} \quad (3.5.13)$$

where $\hat{\Psi}$ are surface or edge interpolation functions. Substituting this relation into Eq. (3.5.12) and using the definition for $d\mathbf{s}$ yields

$$\begin{aligned}\mathbf{q}_c &= \left(\int_{-1}^1 \int_{-1}^1 h_c(\xi, \eta) \hat{\Psi}(\xi, \eta) \hat{\Psi}^T(\xi, \eta) |\mathbf{J}_s| d\xi_s d\eta_s \right) \mathbf{T} \\ &\quad - \int_{-1}^1 \int_{-1}^1 h_c(\xi, \eta) \hat{\Psi}(\xi, \eta) T_c(\xi, \eta) |\mathbf{J}_s| d\xi_s d\eta_s\end{aligned}\quad (3.5.14a)$$

$$= \mathbf{CT} - \mathbf{F}_{hc} \quad (3.5.14b)$$

where

$$\mathbf{C} = \int_{-1}^1 \int_{-1}^1 h_c \hat{\Psi} \hat{\Psi}^T |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.5.15a)$$

$$\mathbf{F}_{hc} = \int_{-1}^1 h_c T_c \hat{\Psi} |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.5.15b)$$

For a given variation of h_c and T_c , the matrix \mathbf{C} and vector \mathbf{F}_{hc} can be computed. Note that in Eq. (3.5.14b) the first term \mathbf{CT} contains the vector of unknown (surface) nodal temperatures and will thus be moved to the left-hand side of the matrix Eq. (3.2.4). When h_c is a function of temperature, the problem is nonlinear. More discussion on the solution of nonlinear problems will be presented in the sequel.

A computation similar to the convective boundary condition may be carried out for the radiation boundary condition. We obtain

$$\mathbf{q}_r = \mathbf{RT} - \mathbf{F}_{hr} \quad (3.5.16)$$

where

$$\mathbf{R} = \int_{-1}^1 \int_{-1}^1 h_r \hat{\Psi} \hat{\Psi}^T |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.5.17a)$$

$$\mathbf{F}_{hr} = \int_{-1}^1 \int_{-1}^1 h_r T_r \hat{\Psi} |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.5.17b)$$

Again, the term \mathbf{RT} will be moved to the left-hand side of Eq. (3.2.4) while \mathbf{F}_{hr} is retained in the source vector. Equations (3.5.17a,b) are highly nonlinear due to the fact that both \mathbf{R} and \mathbf{F}_{hr} are functions of the unknown temperature field \mathbf{T} , since the heat transfer coefficient, h_r , is an explicit function of temperature.

3.6 Semidiscrete Finite Element Model

Returning to the finite element model in Eq. (3.2.4), we have

$$\mathbf{M}^e(\mathbf{T}^e) \dot{\mathbf{T}}^e + \mathbf{K}^e(\mathbf{T}^e) \mathbf{T}^e = \mathbf{Q}^e(\mathbf{T}^e) - \mathbf{q}^e(\mathbf{T}^e) \quad (3.6.1)$$

or using the definition of the boundary fluxes in Eqs. (3.5.1), (3.5.7), (3.5.14b) and (3.5.16)

$$\mathbf{M} \dot{\mathbf{T}} + \mathbf{KT} = \mathbf{Q} - \mathbf{q}_a - \mathbf{CT} + \mathbf{F}_{hc} - \mathbf{RT} + \mathbf{F}_{hr} \quad (3.6.2)$$

Rearranging the terms yields the following matrix equation of ordinary differential equations in time:

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} = \hat{\mathbf{F}} \quad (3.6.3)$$

with

$$\hat{\mathbf{K}} = \mathbf{K} + \mathbf{C} + \mathbf{R}, \quad \hat{\mathbf{F}} = \mathbf{Q} - \mathbf{q}_a + \mathbf{F}_{hc} + \mathbf{F}_{hr} \quad (3.6.4)$$

In general, \mathbf{M} , $\hat{\mathbf{K}}$, and $\hat{\mathbf{F}}$ are functions of temperature and/or time.

3.7 Solution of Nonlinear Equations

3.7.1 Preliminary Comments

The discussion of solution procedures for the discretized form of the heat conduction problem naturally divides itself into two sections according to the time-independent (steady-state) or transient nature of the problem. Solution algorithms for the two types of problems are described here. For most practical conduction problems, the finite element model ultimately involves the solution of a large, sparse, symmetric system of nonlinear algebraic equations. In most cases, the nonlinear equations are linearized first and then solved, often, via a direct method (e.g., Gauss elimination), which is tailored to the finite element formulation (e.g., frontal [15,16] or skyline [17,18] methods). The methods are termed direct methods because they produce a solution to the matrix system in a fixed number of steps; the solution would be exact if infinite precision arithmetic were possible. However, despite the historical popularity of direct methods, there has been a renewed and rapidly growing interest in iterative methods. Iterative methods provide an approximate solution to the matrix problem that usually improves with continued iteration; the number of iterations required to achieve a specified accuracy is not known a priori. The need to analyze extremely large three-dimensional problems and the move to parallel computer architectures has intensified the exploration of iterative methods, such as the preconditioned conjugate gradient (PCG) method. The symmetric, positive-definite matrix forms found in diffusion problems are well suited to iterative methods. These techniques have now become the standard matrix solution algorithms for many types of finite element applications. The matrix solution problem will not be discussed in depth here but may be found in books on linear algebra and matrices (e.g., see [19,20]), and in a condensed form in Appendix B.

In the following sections, a number of linearization algorithms for nonlinear algebraic equations will be described for both steady and transient problems. The iterative methods discussed here for nonlinear problems are basically linearization procedures, and these methods should not be confused with the iterative methods used for the solution of linear algebraic equations.

3.7.2 Steady-State Problems

For problems that are independent of time, the matrix equation (3.6.3) is simplified to

$$\hat{\mathbf{K}}(\mathbf{T})\mathbf{T} = \hat{\mathbf{F}}(\mathbf{T}) \quad (3.7.1)$$

which is a system of nonlinear algebraic equations. Consider first the case where $\hat{\mathbf{K}}$ and $\hat{\mathbf{F}}$ are not functions of the temperature. Then Eq. (3.7.1) reduces to a linear

matrix problem, which can be solved directly, without iteration, by the standard matrix methods noted above.

When Eq. (3.7.1) is nonlinear (e.g., due to temperature-dependent properties, heat source, and/or boundary conditions), it should be linearized. For many mildly nonlinear conduction problems, a simple successive substitution iteration scheme, also known as the *Picard method*, is suitable. The Picard scheme is given by

$$\hat{\mathbf{K}}(\mathbf{T}^n)\mathbf{T}^{n+1} = \hat{\mathbf{F}}(\mathbf{T}^n) \quad (3.7.2)$$

where superscript n denotes the iteration number. By evaluating the coefficient matrix $\hat{\mathbf{K}}$ and right-hand side vector $\hat{\mathbf{F}}$ using the solution \mathbf{T}^n from the previous iteration, we have linearized the problem. The linearized problem is solved for \mathbf{T}^{n+1} (using any matrix solution method), which is then used to evaluate the coefficient matrix and right-hand vector for the next iteration. This procedure is continued until the root-mean-square value of the difference between the solution vectors at two consecutive iterations (normalized with respect to the current solution) is reduced to a value less than a preassigned tolerance, ϵ :

$$\frac{(\mathbf{T}^{n+1} - \mathbf{T}^n)^T(\mathbf{T}^{n+1} - \mathbf{T}^n)}{(\mathbf{T}^{n+1})^T(\mathbf{T}^{n+1})} < \epsilon^2 \quad (3.7.3)$$

The value of ϵ may be taken to be in the range of $10^{-2} - 10^{-4}$.

Typically, problems with slow variations of the material properties, heat transfer coefficients, or heat sources are effectively solved using the Picard method; convergence is normally obtained in a few iterations for problems of this type. An improvement in convergence rate can sometimes be realized by use of a relaxation formula where

$$\hat{\mathbf{K}}(\mathbf{T}^n)\mathbf{T}^* = \hat{\mathbf{F}}(\mathbf{T}^n) \quad (3.7.4a)$$

and

$$\mathbf{T}^{n+1} = \gamma\mathbf{T}^n + (1 - \gamma)\mathbf{T}^* \quad 0 \leq \gamma \leq 1 \quad (3.7.4b)$$

Conduction problems with strong nonlinearities, such as those with radiation boundary conditions, boiling heat transfer coefficients, and Arrhenius type heat sources require iterative techniques that are more powerful than the Picard method. A fairly simple algorithm for this type of problem consists of incrementally approaching the final solution through a series of intermediate solutions. These intermediate solutions may be of physical interest or may simply be a means to obtain the required solution. The formal algorithms used to implement this procedure are termed continuation methods and can be used with either of the iterative methods cited above.

Assume that the solution of Eq. (3.7.1) depends continuously on some real parameter, λ . For heat conduction problems, λ could be the magnitude of a volumetric source or the magnitude of a boundary condition. Then Eq. (3.7.1) can be written in general as

$$\hat{\mathbf{K}}(\mathbf{T}, \lambda)\mathbf{T} = \hat{\mathbf{F}}(\mathbf{T}, \lambda) \quad (3.7.5)$$

which suggests the zeroth order continuation method

$$\hat{\mathbf{K}}(\mathbf{T}_\lambda^n, \lambda^m)\mathbf{T}_\lambda^{n+1} = \hat{\mathbf{F}}(\mathbf{T}_\lambda^n, \lambda^m) \quad (3.7.6)$$

where Eq. (3.7.6) is solved for a series of problems with increasing values of the continuation parameter $\lambda^m = \lambda^{m-1} + \Delta\lambda$. The converged solution, T_λ , at one value of λ is used as the starting solution at the next higher value of λ ; the iterative method in Eq. (3.7.2) or (3.7.4a) is used at each value of λ to achieve a converged solution.

The application of second-order iteration schemes, such as Newton's method (see Section 4.6), is another approach to the highly nonlinear problem. However, for some types of nonlinearities, Newton's method produces an unsymmetric, tangent or Jacobian matrix. The additional computational cost and complexity of solving an unsymmetric system of equations has discouraged the use of such procedures. We will not consider Newton's method for the simple conduction problem but will defer discussion of the algorithm to the later sections on coupled conduction/radiation problems and viscous flows. The most popular alternative for strongly nonlinear problems is to solve the steady-state heat transfer problem via a "false transient" procedure. Generally, implicit time integration formulas, such as those described below, are used to advance the solution through time and asymptotically approach the required steady-state solution. The cost-effectiveness of such a procedure relies heavily on the judicious choice of the integration time step; the step must be large enough to reach steady state quickly but small enough to maintain stability of the solution procedure. The combination of a simple, first-order accurate backward Euler method (see Section 4.6) with an adaptive time step control [21,22] would be an appropriate scheme for most nonlinear conduction problems of this type.

3.7.3 Transient Problems

3.7.3.1 General formulation

The semidiscretization process employed in the finite element method produces a system of coupled, nonlinear ordinary differential equations of the form shown in (3.6.3) for the time-dependent conduction problem. There are a number of possible methods for solving or integrating the system in Eq. (3.6.3), including many of the classical methods used for ordinary differential equations (ODE). However, the (usually) large size of the equation system and the need to take advantage of the special (banded) form of the matrices in Eq. (3.6.3) eliminate many integration methods from consideration.

Finite element time integration methods are usually direct integration methods that employ some relatively simple discrete approximation for the time derivative. Mode superposition methods [23] are also applicable to the system in Eq. (3.6.3), but unlike direct integration, this technique works with a transformed equation set. Mode superposition as an integration method will only be discussed briefly in a later section, as it is not a primary method for general thermal analysis. Aspects of modal analysis will be utilized when discussing the stability of integration methods. The discrete time approximation needed for a direct integration method may be obtained via a finite element or finite difference technique, or by introducing more formal ODE ideas such as the linear multistep methods (LMS) [24]. Finite elements in time have been described by Argyris and Scharpf [25], Donea [26], Bettencourt, et al. [27], and others. Because these techniques result in many of the same integration methods produced by finite difference approximations, they will not be considered here as a separate method.

The procedure for developing a finite difference based direct integration method begins by dividing the integration interval of interest ($0 \leq t \leq t_{final}$) into a number of discrete segments denoted by $\Delta t = t^{n+1} - t^n$, where the superscript denotes the time step number. Within each interval Δt , the temperature and temperature rate are allowed to vary according to some prescription that can be expressed as a simple difference formula. For example, assume that the temperature varies linearly over the interval Δt ; the time-rate-of change of the temperature over the interval Δt can be approximated by the backward difference $\dot{\mathbf{T}}^{n+1} = (\mathbf{T}^{n+1} - \mathbf{T}^n) / \Delta t$. Evaluating (3.6.3) at the $n + 1$ time step and using the above definition produces

$$\frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^{n+1} - \mathbf{T}^n) + \hat{\mathbf{K}} \mathbf{T}^{n+1} = \hat{\mathbf{F}}^{n+1} \quad (3.7.7)$$

Equation (3.7.7) represents a system of algebraic equations for the unknown temperatures \mathbf{T}^{n+1} that can be successively solved at each new time plane.

Other difference approximations for $\dot{\mathbf{T}}$ are possible and lead to other types of recursive formulas for the solution of the temperature field as a function of time. The most popular method used for diffusion problems was introduced in Section 2.8 and consists of the following approximation:

$$\begin{aligned} \dot{\mathbf{T}}^a &= \frac{1}{\Delta t} (\mathbf{T}^{n+1} - \mathbf{T}^n) \\ \mathbf{T}^a &= (1 - \theta) \mathbf{T}^n + \theta \mathbf{T}^{n+1} \end{aligned} \quad (3.7.8)$$

for $0 \leq \theta \leq 1$. When Eq. (3.6.3) is evaluated at \mathbf{T}^a and the above definitions are used, the resulting algorithm, which goes by a variety of names including the generalized trapezoid method, the generalized Crank–Nicolson method, or the generalized midpoint rule, is given by

$$\frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^a) (\mathbf{T}^{n+1} - \mathbf{T}^n) + \theta \hat{\mathbf{K}}(\mathbf{T}^a) \mathbf{T}^{n+1} = \hat{\mathbf{F}}(\mathbf{T}^a) - (1 - \theta) \hat{\mathbf{K}}(\mathbf{T}^a) \mathbf{T}^n \quad (3.7.9)$$

where superscript n indicates the time level, Δt is the time step, and θ is a parameter that determines where in the time interval, t_n to $t_n + \Delta t = t_{n+1}$, the equation will be evaluated.

The choice of $\theta = 0$ produces the Euler (forward difference) method, which is an explicit method that can be rewritten for the general nonlinear case as

$$\frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^n) \mathbf{T}^{n+1} = \frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^n) \mathbf{T}^n - \hat{\mathbf{K}}(\mathbf{T}^n) \mathbf{T}^n + \hat{\mathbf{F}}(\mathbf{T}^n) \quad (3.7.10)$$

and may be rearranged to clearly show its explicit nature

$$\mathbf{T}^{n+1} = \mathbf{T}^n + \Delta t \left[\mathbf{M}^{-1}(\mathbf{T}^n) \hat{\mathbf{F}}(\mathbf{T}^n) - \mathbf{M}^{-1}(\mathbf{T}^n) \hat{\mathbf{K}}(\mathbf{T}^n) \mathbf{T}^n \right] \quad (3.7.11)$$

For effective implementation, the algorithm in Eq. (3.7.11) requires that the capacitance matrix, \mathbf{M} , be easily invertible, i.e., be diagonal. The reduction of the consistent capacitance matrix to a diagonal or “lumped” form can be accomplished by various procedures (e.g., row-sum technique [2,5,23,24]) for some types of finite

elements (generally linear elements) but is not a universally available option. Also, the explicit nature of Eq. (3.7.11) implies a stability condition in terms of a maximum allowable time step, the thermal diffusivity of the material, and the finite element mesh spacing (see [24,28]); the stability condition can produce a prohibitively small time step for some problems. The Euler method is first-order accurate in time (i.e., $O(\Delta t)$). Despite these limitations, explicit procedures are in common use in conduction problems, especially for analyses involving short-time transient phenomena, such as thermal shocks. The main attraction of an explicit scheme is the fact that a matrix solution is not required and computer storage requirements are therefore minimal.

For the cases for which $\theta > 0$, Eq. (3.7.9) produces a family of implicit methods which requires the solution of a matrix problem at each time step. For example, when $\theta = 1$, Eq. (3.7.9) reduces to the backward Euler (backward difference) method

$$\left[\frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^{n+1}) + \hat{\mathbf{K}}(\mathbf{T}^{n+1}) \right] \mathbf{T}^{n+1} = \frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^{n+1}) \mathbf{T}^n + \hat{\mathbf{F}}(\mathbf{T}^{n+1}) \quad (3.7.12)$$

which is a fully implicit scheme. This is an unconditionally stable method (i.e., there is no time step restriction; see a later section for a discussion of convergence and stability of numerical integration methods), and is first-order accurate in time. In the general nonlinear case, the matrix problem presented in Eq. (3.7.12) requires an iterative solution for \mathbf{T}^{n+1} within each time step; the linear case can be solved without nonlinear iteration, though a matrix solution is still required. The Picard scheme would be an acceptable choice for iteratively solving Eq. (3.7.12) at time t_{n+1} . In treating the nonlinear form of (3.7.12), the use of predictor-corrector schemes, extrapolation methods, and quasi-linearization can often reduce the computational effort at each step without significantly reducing the accuracy of the method. For mild nonlinearities and modest time steps, a quasi-linearization of Eq. (3.7.12) yields

$$\left[\frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^n) + \hat{\mathbf{K}}(\mathbf{T}^n) \right] \mathbf{T}^{n+1} = \frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^n) \mathbf{T}^n + \hat{\mathbf{F}}(\mathbf{T}^n) \quad (3.7.13)$$

which is now a single-step, non-iterative method for the solution of \mathbf{T}^{n+1} . An extrapolation procedure of the form

$$\mathbf{T}^* = \frac{3}{2} \mathbf{T}^n - \frac{1}{2} \mathbf{T}^{n-1} \quad (3.7.14)$$

in conjunction with

$$\left[\frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^*) + \hat{\mathbf{K}}(\mathbf{T}^*) \right] \mathbf{T}^{n+1} = \frac{1}{\Delta t} \mathbf{M}(\mathbf{T}^*) \mathbf{T}^n + \hat{\mathbf{F}}(\mathbf{T}^*) \quad (3.7.15)$$

will often produce a better solution than (3.7.13) and allow a larger time step. This form is attractive as a false transient procedure for mildly nonlinear problems. For strong nonlinearities the coupling of a forward Euler predictor (3.7.11) with (3.7.12) as a corrector is a viable option (see [21]). Such methods are often used in practice with good success though the experience of the analyst is a crucial element in selecting the proper method.

We close this discussion with a brief outline of the only second-order accurate method in this family of approximations, namely, the Crank–Nicolson method (i.e., $\theta = \frac{1}{2}$). Like the backward difference method, this is an unconditionally stable method, and it requires an iterative solution at each time step for the general nonlinear case. From Eq. (3.7.9) we have

$$\left[\frac{2}{\Delta t} \mathbf{M}(\mathbf{T}^a) + \hat{\mathbf{K}}(\mathbf{T}^a) \right] \mathbf{T}^a = \frac{2}{\Delta t} \mathbf{M}(\mathbf{T}^a) \mathbf{T}^n + \hat{\mathbf{F}}(\mathbf{T}^a) \quad (3.7.16a)$$

with the new solution recovered from the definition

$$\mathbf{T}^{n+1} = 2\mathbf{T}^a - \mathbf{T}^n \quad (3.7.16b)$$

Though Eq. (3.7.16a) has no time step restriction for stability, it does suffer from “noise” or temporal oscillations in the solution when the time step exceeds a critical value (see the later section on Convergence and Stability). Several modifications have been proposed to correct or filter this problem (see [4,29,30]). Predictor-corrector, extrapolation, and quasi-linearization schemes can also be used with (3.7.16a) to reduce the computational effort at each time step. The Crank–Nicolson scheme and its variants are probably the most frequently used procedures for heat conduction problems.

3.7.3.2 Predictor-corrector methods

The time integration methods outlined above are single-step methods that work effectively when an appropriate time step Δt is selected. The time step selection, however, is crucial to accuracy though in many physical problems there is little guidance available to precisely determine a suitable value for Δt . A cost-effective solution to time step selection and a general increase in integrator performance is available through use of predictor-corrector methods. The method described here was originally developed by Gresho et al. [21]; improvements to the original method are outlined in [22].

An implicit integration method that is second-order accurate in time can be developed by combining an explicit prediction method with the Crank–Nicolson (trapezoid) method described previously. A second-order, variable step, explicit method is the Adams–Bashforth predictor given by

$$\mathbf{T}_p^{n+1} = \mathbf{T}^n + \frac{\Delta t_n}{2} \left[\left(2 + \frac{\Delta t_n}{\Delta t_{n-1}} \right) \dot{\mathbf{T}}^n - \left(\frac{\Delta t_n}{\Delta t_{n-1}} \right) \dot{\mathbf{T}}^{n-1} \right] \quad (3.7.17)$$

where $\Delta t_n = t_{n+1} - t_n$ and $\Delta t_{n-1} = t_n - t_{n-1}$. This formula can be used to predict the temperature solution vector given two “acceleration” vectors from previous times; no matrix solution is required.

A compatible corrector formula for use with (3.7.17) is available in the form of the trapezoid rule. When applied to the conduction equation, the trapezoid rule produces

$$\left[\frac{2}{\Delta t_n} \mathbf{M}(\mathbf{T}^a) + \hat{\mathbf{K}}(\mathbf{T}^a) \right] \mathbf{T}^{n+1} = \frac{2}{\Delta t_n} \mathbf{M}(\mathbf{T}^a) \mathbf{T}^n + \mathbf{M}(\mathbf{T}^a) \dot{\mathbf{T}}^n + \hat{\mathbf{F}}(\mathbf{T}^a) \quad (3.7.18)$$

which is a rearranged form of the Crank–Nicolson method given in (3.7.16a). Equation (3.7.18) is observed to be a nonlinear, algebraic system for the vector \mathbf{T}^{n+1} and can again be solved using an iterative procedure such as Picard’s method.

The major steps in the time integration procedure utilizing (3.7.17) and (3.7.18) are outlined here. At the beginning of each time step it is assumed that all of the required solution and “acceleration” vectors are known and the time increment for the next step has been selected. To advance the solution from time t_n to time t_{n+1} then requires the following steps:

1. A tentative solution vector, \mathbf{T}_p^{n+1} , is computed using the predictor Eq. (3.7.17).
2. The corrector Eq. (3.7.18) is solved for the “true” solution, \mathbf{T}^{n+1} . This involves the iterative solution of (3.7.18) via Picard’s method. The predicted values \mathbf{T}_p^{n+1} are used to estimate \mathbf{T}^a and initialize the equation for the iteration procedure.
3. The “acceleration” vectors are updated using the new solution \mathbf{T}^{n+1} and the “inverted” forms of the corrector formulas, that is

$$\dot{\mathbf{T}}^{n+1} = \frac{2}{\Delta t_n} (\mathbf{T}^{n+1} - \mathbf{T}^n) - \dot{\mathbf{T}}^n \quad (3.7.19)$$

4. A new integration time-step is computed. The time-step selection process is based on an analysis of the time truncation errors in the predictor and corrector formulas as described in the next section. If a constant time-step is being used, this step is omitted.
5. Return to step 1 for next time increment.

In actual implementation the Picard iteration process in step 2 is not carried to absolute convergence. Rather, a one-step correction is employed as advocated in [21,22]. This procedure is quite efficient and can be very accurate provided the time step is suitably controlled.

3.7.3.3 Time step control

The implicit time integration procedures outlined above can be used with a fixed, user specified time step or a time step that changes only at certain points during the integration interval. However, the a priori selection and modification of a reasonable integration time step can be a difficult task, especially for a complex problem. One of the benefits of using the predictor-corrector algorithms described here is that they provide a rational basis for dynamically selecting the time step. It should be noted that the method outlined here is not the only approach available for adaptive time step control. The work of Valli, et al. [31,32] presents the use of Proportional-Integral-Derivative (PID) control to the integration of finite element equation systems.

The detailed derivation of the time step selection formula is omitted here. The reader interested in further details is referred to [21,22]. The general ideas for the time step selection process come from the well-established procedures for solving ordinary differential equations. By comparing the time truncation errors for two time integration methods of comparable order, a formula can be developed to predict the next time step based on a user specified error tolerance. In the present case, the time truncation errors for the explicit predictor and implicit corrector steps are analyzed.

The time step estimation formula is given by Gresho, et al. [21] as

$$\Delta t_{n+1} = \Delta t_n \left(b \cdot \frac{\epsilon^t}{d_{n+1}} \right)^m \quad (3.7.20)$$

where $m = 1/3$, $b = 3(1 + \Delta t_{n-1}/\Delta t_n)$ for this second-order scheme. The user-specified error tolerance for the integration process is ϵ^t , which typically has a default value of 0.001. The quantity d_{n+1} is an appropriate norm on the integration error, which is defined as the difference between the predicted solution and the corrected value. Generally, the following norm is used

$$d_{n+1} = \frac{1}{\sqrt{NT_{max}}} \left[\sum_{i=1}^N \left(T_i^{(n+1)} - T_{ip}^{(n+1)} \right)^2 \right]^{\frac{1}{2}} \quad (3.7.21)$$

where N is the total number of nodes in the problem, T_{max} is a representative (constant) temperature scale for the problem, and T_i and T_{ip} are the corrected and predicted temperatures at the nodes.

3.7.3.4 Initialization

The predictor, Eq. (3.7.17), requires that one or more acceleration or rate vectors be available at each time in order to estimate a new solution vector. At the beginning of a transient solution these vectors are not generally available and thus a special starting procedure must be used. The approach normally taken is to use the dissipative backward Euler method for the first few steps and then switch to the standard second-order predictor-corrector methods. This procedure has the advantage that any nonphysical features of the numerical model are quickly damped by the backward Euler scheme.

For the first time step, the implicit, backward Euler scheme is used alone; the second step uses a forward Euler predictor and backward Euler corrector. Both of these steps use a fixed, user-supplied time step. At the third step, the usual predictor/corrector integration procedure begins and automatic time step selection is started. The initial time step supplied by the user to start the problem should be very conservative to prevent large time step reductions when the automatic selection procedure takes control.

3.7.3.5 Linear multi-step methods

Most of the time integration methods described above were obtained by introducing relatively simple finite difference expressions for the time derivative into the semidiscrete finite element equation. This approach resulted in single step integration formulas in which the solution at the new time relies only on the solution data from the previous time. Multistep methods that rely on data from more than one previous time step are also possible. These methods may be derived from “higher-order” finite difference approximations for \dot{T} . However, a more formal approach with rigorous mathematical support involves the linear multi-step (LMS) methods developed for the solution of ordinary differential equations. Here we will outline a few applications of LMS to the finite element conduction problems and illustrate how some of the previous methods can be obtained as special cases of LMS. Further details on LMS are available in [33].

Consider the first-order system of equations expressed by

$$\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, t) \quad (3.7.22)$$

A general LMS of order k for the solution of Eq. (3.7.22) is written as

$$\sum_{i=0}^k \left\{ \alpha_i \mathbf{y}^{n-i} + \Delta t \beta_i \mathbf{f}(\mathbf{y}^{n-i}, t_{n-i}) \right\} \quad (3.7.23)$$

Through the selection of appropriate values of the coefficients α_i and β_i and the order k , a number of well-known methods may be described. Note that if $\beta_0 = 0$, the resulting method is *explicit*, and if $\beta_0 \neq 0$ then the integration method is *implicit*. For single-step cases where $k = 1$, the selection of $\alpha_0 = 1$, $\alpha_1 = -1$, $\beta_0 = 0$, $\beta_1 = -1$ produces the explicit method of Eq. (3.7.9), while the choice $\alpha_0 = 1$, $\alpha_1 = -1$, $\beta_0 = -1$, $\beta_1 = 0$ corresponds to the implicit backward Euler scheme in (3.7.12).

Predictor-corrector methods are closely associated with LMS methods due to the commonality with the form in (3.7.23) for explicit and implicit integration methods. As noted previously, the Adams-Basforth-Moulton family of methods has been used in a finite element context by Gresho, et al. [21]. An explicit two step ($k = 2$), second-order form of (3.7.23) selects $\alpha_0 = 1$, $\alpha_1 = -1$, $\alpha_2 = 0$, $\beta_0 = 0$, $\beta_1 = -3/2$, $\beta_2 = 1/2$, which produces

$$\mathbf{y}^{n+1} = \mathbf{y}^n + \frac{3}{2} \Delta t \mathbf{f}(\mathbf{y}^n, t_n) - \frac{1}{2} \Delta t \mathbf{f}(\mathbf{y}^{n-1}, t_{n-1}) \quad (3.7.24)$$

Equation (3.7.24) is the constant time step form of (3.7.17) and would function as the predictor equation. The implicit, second-order companion to (3.7.24) is a single step method ($k = 1$) with coefficients $\alpha_0 = 1$, $\alpha_1 = -1$, $\beta_0 = -1/2$, $\beta_1 = -1/2$. This leads to

$$\mathbf{y}^{n+1} = \mathbf{y}^n + \frac{\Delta t}{2} \left\{ \mathbf{f}(\mathbf{y}^n, t_n) + \mathbf{f}(\mathbf{y}^{n+1}, t_{n+1}) \right\} \quad (3.7.25)$$

which is recognized as the trapezoidal rule. Rewriting (3.6.3) in the form of (3.7.22), we obtain

$$\dot{\mathbf{T}} = -\mathbf{M}^{-1} \mathbf{K} \mathbf{T} + \mathbf{M}^{-1} \mathbf{F} \quad (3.7.26)$$

and substituting into the formulas in Eqs. (3.7.24) and (3.7.25) produces the predictor-corrector equations shown in (3.7.17) and (3.7.18).

Other first- and second-order methods can be derived from (3.7.23) as well as higher-order schemes. However, practical limits on function evaluations and stability considerations limit most LMS methods to those outlined here. The most significant aspect of the LMS formula is its use in the area of predictor-corrector procedures.

3.7.3.6 Convergence and stability

To conclude the sections on direct time integration methods, some discussion of convergence of the numerical solution to the actual solution is required. The time integration methods of the previous sections can all be written in symbolic form as

$$\mathbf{T}^{n+1} = -\mathbf{G} \mathbf{T}^n + \mathbf{H} \quad (3.7.27)$$

where \mathbf{G} represents the discretized form of the relevant differential operator. A time integration method is said to be *convergent* if, for fixed time t_n and time step Δt , the numerical solution \mathbf{T}^n converges to its true value $\mathbf{T}(t_n)$ as $\Delta t \rightarrow 0$. To prove convergence of (3.7.27), the well-known Lax Equivalence Theorem [34] may be invoked, which allows the consistency and stability of \mathbf{G} to be necessary and sufficient conditions for convergence. The numerical scheme (3.7.27) is said to be *consistent* with the original continuous problem if the errors (e.g., round-off and truncation errors) go to zero in the limit as $\Delta t \rightarrow 0$. For the diffusion problems considered here, consistency can be proved as shown in [24,28].

Since a time integration scheme is approximate, error is introduced into the solution \mathbf{T}^{n+1} at each time step. The error in the solution at one time step feeds to the next time step due to the recursive use of Eq. (3.7.27). If the error is ultimately bounded, the time integration scheme is said to be *stable*. Thus, the stability of the time integration method in (3.7.27) implies that as \mathbf{G} is recursively applied to each new vector \mathbf{T}^n , any errors that occur ultimately decay. That is, \mathbf{G} is a bounded operator with a bounded spectral radius. For this discussion, the source of error is unimportant, though we recognize that it may occur due to approximation of the differential equations and/or boundary conditions (truncation error) or due to finite precision computation (round-off errors).

There are a variety of techniques for assessing the stability of methods such as (3.7.27). For finite element applications, the spectral or modal methods are generally preferred. Here we will only sketch the procedure, without proof, and illustrate the stability conditions for the θ -family of integration methods.

The basic semidiscrete equation

$$\mathbf{M}\dot{\mathbf{T}} + \mathbf{K}\mathbf{T} = \mathbf{F} \quad (3.7.28)$$

is difficult to analyze since it represents a large number of equations (degrees of freedom) that are coupled through the \mathbf{M} and \mathbf{K} matrices. A standard approach to this difficulty is to rewrite Eq. (3.7.28) in terms of its normal modes and thereby produce a system of uncoupled equations that are more easily studied. The generalized eigenvalue problem associated with (3.7.28) is (no sum on i)

$$\mathbf{K}\phi_i - \lambda_{(i)}\mathbf{M}\phi_i = \mathbf{0} \quad (3.7.29)$$

where λ_i is the eigenvalue corresponding to the eigenvector ϕ_i . Assume that the eigenvectors are orthonormalized with respect to the capacitance matrix \mathbf{M} such that

$$\phi_i^T \mathbf{M} \phi_j = \delta_{ij} \quad (3.7.30)$$

where δ_{ij} are the components of the unit tensor (Kronecker delta). Then multiplying Eq. (3.7.29) by ϕ_j^T and using Eq. (3.7.30) yields

$$\phi_j^T \mathbf{K} \phi_i - \lambda_{(i)} \phi_j^T \mathbf{M} \phi_i = \mathbf{0} \quad (3.7.31)$$

or

$$\phi_j^T \mathbf{K} \phi_i - \lambda_{(i)} \delta_{ij} = \mathbf{0} \quad (3.7.32)$$

which shows that the eigenvectors are also orthogonal with respect to the diffusion matrix \mathbf{K} .

Since the eigenvectors form a basis for the semidiscrete system in (3.7.28), the solution for \mathbf{T} may be represented as a linear combination of the eigenvectors,

$$\mathbf{T}(\mathbf{x}, t) = \sum_i \phi_i(\mathbf{x}) \alpha_i(t) \quad (3.7.33)$$

where α_i are the generalized coordinates. Substituting Eq. (3.7.33) into Eq. (3.7.28), premultiplying with ϕ_j^T , and using the orthogonal properties in Eqs. (3.7.30) and (3.7.32) leads to the result

$$\dot{\alpha}_i (\phi_j^T \mathbf{M} \phi_i) + \alpha_i (\phi_j^T \mathbf{K} \phi_i) = \phi_j^T \mathbf{F} \quad (3.7.34)$$

and

$$\dot{\alpha}_i \delta_{ij} + \alpha_i \lambda_{(i)} \delta_{ij} = f_j \quad (3.7.35)$$

or

$$\dot{\alpha}_j + \alpha_j \lambda_{(j)} = f_j \quad (3.7.36)$$

Note that Eq. (3.7.36) represents the decoupled modal equation for each degree of freedom (and eigenvalue) in the original system (3.7.28). If Eq. (3.7.36) can be integrated in time for each α_i , then the solution \mathbf{T} can be constructed from the eigenvector expression in (3.7.33). This is the basis for mode superposition solution techniques (see [23,24] and the next section).

The single degree of freedom equations in (3.7.36) serve another purpose and allow study of the stability properties of various time integration methods. To demonstrate this idea, the time derivative and difference approximations for the θ method, which are defined in Eq. (3.7.8), can be applied to Eq. (3.7.36) to give

$$\frac{\alpha_i^{n+1} - \alpha_i^n}{\Delta t} + \lambda_{(i)}(1 - \theta)\alpha_i^n + \lambda_{(i)}\theta\alpha_i^{n+1} = f_i^a \quad (3.7.37)$$

This can be rearranged as

$$\alpha_i^{n+1} = \frac{1 - \lambda_{(i)}\Delta t(1 - \theta)}{1 + \lambda_{(i)}\Delta t\theta} \alpha_i^n + \frac{\Delta t f_i^a}{1 + \lambda_{(i)}\Delta t\theta} \quad (3.7.38)$$

which is of the same form as given in (3.7.27) with

$$G_i = \frac{1 - \lambda_{(i)}\Delta t(1 - \theta)}{1 + \lambda_{(i)}\Delta t\theta} \quad (3.7.39)$$

As stated previously, the stability of the integration method is proved by showing that the amplification matrix \mathbf{G} is a bounded operator. For the single degree of freedom equations, the analogous requirement is that the amplification factor, G_i , be less than unity, i.e., $|G_i| < 1$. The restriction on G_i then is

$$-1 < \frac{1 - \lambda_i \Delta t(1 - \theta)}{1 + \lambda_i \Delta t\theta} < 1 \quad (3.7.40)$$

where θ must be between 0 and 1, and (3.7.40) must hold for all the modes, λ_i of the system. The right-hand inequality imposes no restrictions on the values $\lambda_i, \Delta t$, or θ ; the left-hand inequality requires that $\lambda_i \Delta t < 2/(1 - 2\theta)$ when $\theta < 1/2$ but has no restrictions for $\theta \geq 1/2$. When there are no restrictions on the integration time step the method is termed *unconditionally stable* or simply *stable*. This analysis shows that all θ methods with $\theta \geq 1/2$ are unconditionally stable and these include the backward difference method ($\theta = 1$) in Eq. (3.7.12) and the Crank–Nicolson method ($\theta = 1/2$) in Eq. (3.7.16). The conditionally stable θ method is the Euler method ($\theta = 0$) in Eq. (3.7.10), where the time step is limited by $\lambda_i \Delta t_{crit} < 2$.

To ensure stability for all modes in the system, the time step must be limited by the largest eigenvalue in the system or $\Delta t_{crit} < 2/\lambda_{max}$. Recall that the λ_i are related to the system matrices \mathbf{M} and \mathbf{K} ; it can be shown that $\lambda_{max} \sim \alpha/h_{min}^2$, where $\alpha = k/\rho C$ is the thermal diffusivity (k is the thermal conductivity, ρ is the density, and C is the specific heat) and h_{min} is the characteristic dimension of the smallest finite element in the mesh. Thus for an explicit method the time step is limited by $\Delta t_{crit} < \bar{C}h_{min}^2$ (where \bar{C} is a constant), which shows the strong influence of mesh size (refinement) on performance of the time integration method. Though the unconditionally stable methods ($\theta \geq 1/2$) have no stability restrictions on the time step, there are time step ranges which will produce a nonphysical oscillatory response in the solution. The oscillation limit corresponds to the point $\lambda_i \Delta t_{osc}$ where $G_i = 0$. As an example, for the Crank–Nicolson method ($\theta = 1/2$) the oscillation limit is $\lambda_i \Delta t_{osc} = 2$, and for $\Delta t > 2/\lambda_i$ the solution will exhibit an oscillatory decay. Oscillatory behavior can be filtered to produce acceptable solutions as shown by Wood and Lewis [29] and Wood [30].

3.7.3.7 Mode superposition methods

To conclude the discussion of transient solution methods, the mode superposition method is briefly outlined. To find the modal response to the unforced, semidiscrete conduction equation

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} = \mathbf{0} \quad (3.7.41)$$

assume that the temperature field can be represented as

$$\mathbf{T} = \phi_i e^{-\lambda_{(i)}t} ; \quad \dot{\mathbf{T}} = -\lambda_{(i)}\phi_i e^{-\lambda_{(i)}t} \quad (3.7.42)$$

where $\lambda_{(i)}$ is the eigenvalue corresponding to the ϕ_i eigenvector. When (3.7.42) is substituted into (3.7.41) the result is

$$\mathbf{K}\phi_i - \lambda_{(i)}\mathbf{M}\phi_i = 0 \quad (3.7.43)$$

for an individual eigenvector or

$$\mathbf{K}\Phi - \Lambda\mathbf{M}\Phi = \mathbf{0} \quad (3.7.44)$$

for the eigenspectrum where Φ is a matrix of eigenvectors

$$\Phi = [\phi_1, \phi_2, \phi_3, \dots, \phi_n] \quad (3.7.45)$$

and Λ is a diagonal matrix of eigenvalues.

As noted in the previous section, the eigenvectors may be assumed orthonormal with respect to the capacitance matrix \mathbf{M} or

$$\phi_i^T \mathbf{M} \phi_j = \delta_{ij} \quad (3.7.46)$$

which leads to the eigenvectors being orthogonal with respect to the diffusion matrix \mathbf{K} . The eigenvectors are therefore a basis for the discretized conduction equation and the solution vector \mathbf{T} can be represented in terms of this new basis as

$$\mathbf{T} = \Phi^T(\mathbf{x})\alpha(t) \quad (3.7.47)$$

where α is a vector of generalized coordinates. Substitution of (3.7.47) into the original semidiscrete conduction equation, premultiplying by Φ^T and using the orthogonality conditions leads to

$$\dot{\alpha} + \Lambda\alpha = \Phi^T \mathbf{F} = \mathbf{f} \quad (3.7.48)$$

which is the system equivalent of the individual modal equations in (3.7.36). Note that the equations for the generalized coordinates in (3.7.48) are decoupled as the eigenvalue matrix Λ is diagonal.

The individual modal equations in (3.7.48) may be integrated in time, either analytically or numerically, depending on the time dependence of the forcing function \mathbf{F} . Initial conditions for the generalized coordinates are derived from

$$\Phi^T \mathbf{M} \mathbf{T}(t_0) = \alpha(t_0) = \alpha^0 ; \quad \Phi^T \mathbf{M} \dot{\mathbf{T}}(t_0) = \dot{\alpha}(t_0) = \dot{\alpha}^0 \quad (3.7.49)$$

The general solution for the i th component of (3.7.48) may be written as

$$\alpha_i(t) = \alpha_i^0 e^{-\lambda_{(i)} t} + e^{-\lambda_{(i)} t} \int^t e^{\lambda_{(i)} \tau} f_i(\tau) d\tau \quad (3.7.50)$$

where the integral may be analytically evaluated for simple f_i functions. Direct numerical time integration methods would be used for (3.7.48) when the f_i function is complex. Once the generalized coordinates are found as a function of time, the finite element temperature field can be constructed using (3.7.47).

The solution to (3.7.48) depends directly on the ability to compute the spectrum of eigenvalues (Λ, Φ) of the original finite element system in (3.7.41). Specialized numerical methods are available for solving the generalized eigenvalue problem in (3.7.44) but these procedures can be expensive for very large finite element models. The general effectiveness of the modal analysis method is derived from the assumption that not all of the spectrum of eigenvalues needs to be computed. Only relatively few of the lowest modes (largest eigenvalues), which contain the majority of the system “energy,” need to be found to adequately represent the entire system. Eigenvalue solvers for computing a small percentage of the spectrum are very efficient and by limiting the solution to a few generalized coordinates an effective computational model with few degrees of freedom can be developed. The number of modes to retain (and compute) is dependent on the spatial distribution and energy content of the forcing function. The relative size of the eigenvalues

correlates with the importance of the eigenvectors and can assist in the selection of the number of modes.

The model analysis method can be very effective for a limited class of problems. First, note that the method is normally considered only for linear conduction problems where the eigenvalue problem only needs to be determined once. For nonlinear applications the required eigenvalue spectrum would have to be recomputed every time step or at least every few time steps. This repetitive computation can become prohibitively expensive. The availability of very fast matrix solvers generally makes direct integration methods more effective for nonlinear problems. However, for linear problems with a response that can be captured in a few modes, the mode superposition method can be a very efficient technique. Also, the method is very suitable when multiple forcing functions are to be considered. Finally, the ideas of “reduced-order” models are based on mode superposition techniques. In this approach, complex finite element models are transformed to models with significantly fewer degrees of freedom while retaining the overall response of the system. These models, using Proper Orthogonal Decomposition (POD) or the method of snapshots, are increasingly used to simulate thermal problems in design and control applications. Examples of this type of analysis can be found, for example, in [35–37].

3.8 Radiation Solution Algorithms

The enclosure radiation problem described in Section 1.12 is often required as part of a complete thermal analysis and must be solved in conjunction with the finite element heat conduction equations outlined above. Radiation coupling with a fluid mechanics problem is also possible though there is little algorithmic difference compared to the conduction problem; both types of problem involve a coupling at the boundary. An energy balance for each surface in the enclosure leads to the following system of equations [see Eq. (1.12.1)]

$$\sum_{j=1}^N \left[\frac{\delta_{kj}}{\epsilon_j} - F_{k-j} \left(\frac{1-\epsilon_j}{\epsilon_j} \right) \right] \bar{q}_j = \sum_{j=1}^N (\delta_{kj} - F_{k-j}) \sigma \bar{T}_j^4 \quad (3.8.1)$$

which are often rewritten

$$\sum_{j=1}^N [\delta_{kj} - (1-\epsilon_k)F_{k-j}] \bar{q}_j^o = \epsilon_k \sigma \bar{T}_k^4 \quad (3.8.2)$$

$$\bar{q}_k = \bar{q}_k^o - \sum_{j=1}^N F_{k-j} \bar{q}_j^o \quad (3.8.3)$$

where \bar{q}_j^o is the outgoing radiative flux for each surface, \bar{q}_k is the net flux from each surface, δ_{kj} is the unit tensor, σ is the Stefan–Boltzmann constant, and F_{k-j} are radiation view (configuration) factors [see Eq. (1.12.2)]. The over-bar denotes a quantity that is uniform over each surface. When the surface temperatures \bar{T}_k of all surfaces are known, Eq. (3.8.1) forms a set of linear algebraic equations for the unknown, net surface fluxes, \bar{q}_j . Likewise, Eq. (3.8.2) forms a set of linear algebraic

equations for the unknown, outgoing surface fluxes, \bar{q}_j^o and Eq. (3.8.3) transforms the outgoing fluxes to the net surface fluxes.

Each of the above formulations can be written as a matrix problem. The net flux Eq. (3.8.1) becomes

$$(\mathbf{I} - \mathbf{F}_{vf}\rho)\bar{\mathbf{q}} = (\mathbf{I} - \mathbf{F}_{vf})\epsilon\sigma\bar{\mathbf{T}}^4 = (\mathbf{I} - \mathbf{F}_{vf})\epsilon\mathbf{E}_b \quad (3.8.4)$$

or more compactly

$$\mathbf{A}(\bar{\mathbf{T}})\bar{\mathbf{q}} = \mathbf{D}(\bar{\mathbf{T}})\bar{\mathbf{T}} \quad (3.8.5)$$

The outgoing flux form of the Eqs. (3.8.2) and (3.8.3) is expressed in matrix form as

$$(\mathbf{I} - \mathbf{F}_{vf}\rho)\bar{\mathbf{q}}^o = \mathbf{A}(\bar{\mathbf{T}})\bar{\mathbf{q}}^o = \epsilon\mathbf{E}_b \quad (3.8.6)$$

$$\bar{\mathbf{q}} = (\mathbf{I} - \mathbf{F}_{vf})\bar{\mathbf{q}}^o \quad (3.8.7)$$

In the above, \mathbf{I} is the identity matrix, \mathbf{F}_{vf} is the matrix of view factors, $\rho = \mathbf{I} - \epsilon$ and ϵ are diagonal matrices of reflectances and emittances, respectively and \mathbf{E}_b is a vector representing the black-body emissive power. The \mathbf{A} matrix is the radiative flux coefficient matrix, with a temperature dependence due to surface property variations, and \mathbf{D} is the surface temperature coefficient matrix with a cubic temperature dependence. Note that the matrix \mathbf{A} is a full matrix due to the surface to surface coupling represented by the view factors $F_{k-j} = \mathbf{F}_{vf}$. This characteristic, along with the possible temperature dependencies, suggests the use of an iterative solution method for the matrix problems in (3.8.5) and (3.8.6) rather than a direct matrix factorization.

In all of the formulations, the assumptions of the radiation model require that the values of the surface temperatures $\bar{\mathbf{T}}$ and surface fluxes $\bar{\mathbf{q}}$ be constant over each surface. The relationship between these discrete surface temperature and flux variables and the finite element nodal temperatures and boundary flux variables remains to be determined. An obvious assumption is for the faces (edges) of the finite elements bounding the enclosure to be used as the description of the enclosure surface. In this case the (constant) surface flux obtained from Eq. (3.8.5) or Eq.(3.8.7) can be used directly as applied boundary conditions in the finite element equations [see Eq. (3.5.6)]. Likewise, the constant surface temperature required in (3.8.5)–(3.8.7) can be obtained by an appropriate transformation on the element face. That is, $\bar{\mathbf{T}} = \boldsymbol{\Pi}\mathbf{T}$ where $\boldsymbol{\Pi}$ is a projector. The definition of the projector may take a variety of forms including simple averaging, centroidal shape function evaluation or area-weighted averaging. By defining common variables for the conduction Eq. (3.2.8) and the radiation Eqs. (3.8.5)–(3.8.7), the physical processes can be coupled and a solution strategy defined. Note that the use of element faces as an enclosure description is not the only method of coupling the equations. Another possibility is to use the nodes of the finite element model as the centroid of a control area made up of surface contributions from all the elements attached to the node. Nodal point temperatures are then common between the conduction and radiation problem, though boundary surface fluxes now require interpolation. This construction is more finite volume in nature and also leads to a slight increase in complexity when defining data for the view factor computation.

In the following, attention is restricted to the case of element faces forming the description of the enclosure.

Having defined the common variables for the two equations, a solution algorithm for the coupled problem is required. A variety of methods have been developed and some of these have been studied and compared by Hogan and Gartling [38]. Because of the strong nonlinearity in the conduction/radiation problem (i.e., \bar{T}^4 boundary condition), an implicit algorithm that treats $\bar{\mathbf{q}}$ and \mathbf{T} simultaneously is the most desirable from a convergence and performance point of view. The most obvious approach for the two equation sets in two unknowns is the direct substitution of the radiative flux from (3.8.5) into the heat conduction boundary condition. Inverting the coefficient matrix in (3.8.5) produces

$$\bar{\mathbf{q}} = \mathbf{A}^{-1} \bar{\mathbf{D}}(\bar{\mathbf{T}}) \bar{\mathbf{T}} \quad (3.8.8)$$

Substituting (3.8.8) into (3.5.6) as an applied flux, projecting the surface temperature to the nodal temperature, then the conduction Eq. (3.6.3) becomes

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} + \hat{\mathbf{K}}_r\mathbf{T} = \hat{\mathbf{F}} \quad (3.8.9)$$

where the new flux term is given by

$$\hat{\mathbf{K}}_r = \oint_{\Gamma_e} \hat{\Psi} \bar{\mathbf{q}} \, ds = \oint_{\Gamma_e} \hat{\Psi} \mathbf{A}^{-1} \bar{\mathbf{D}}(\bar{\mathbf{T}}) \, ds \quad (3.8.10)$$

Note that $\bar{\mathbf{D}}$ is different from the original operator \mathbf{D} because of the temperature projection.

Equation (3.8.9) is a nonlinear equation in terms of the nodal point temperatures only. This system could be solved by any of the iterative or time integration methods described in the previous section. However, there are some severe drawbacks to this algorithm. The inverse of \mathbf{A} in Eq. (3.8.8) is nontrivial for any enclosure with more than a moderate number of surfaces. Recall that \mathbf{A} is a full matrix as is its inverse. Also, if \mathbf{A} is a function of temperature then the inverse would be required multiple times, corresponding to each iteration or each time step in the conduction solution. The matrix-matrix and matrix-vector multiplications in (3.8.10) are also a significant computational burden for the generation of the flux boundary condition. Note that Eq. (3.8.9) could be rearranged by rewriting Eq. (3.8.10) as a standard radiation boundary condition [see Eq. (3.5.16)] where the \mathbf{R} matrix contains a coefficient h_r based on \bar{T}^3 . This does not alleviate any of the disadvantages of the algorithm. The basic method remains suitable only for enclosures with a small number of surfaces. A demonstration of this type of algorithm and details of its implementation can be found in [39] and [40].

Another implicit method leaves Eqs (3.6.3) and (3.8.5) as distinct equation sets but solves them simultaneously, as a fully coupled system. Considering the time-independent case for simplicity, Eqs. (3.6.3) and (3.8.5) can be written as a combined matrix problem as

$$\begin{bmatrix} \hat{\mathbf{K}} & \mathbf{B} \\ -\bar{\mathbf{D}}(\bar{\mathbf{T}}) & \mathbf{A} \end{bmatrix} \begin{Bmatrix} \mathbf{T} \\ \bar{\mathbf{q}} \end{Bmatrix} = \begin{Bmatrix} \hat{\mathbf{F}} \\ \mathbf{0} \end{Bmatrix} \quad (3.8.11)$$

where \mathbf{B} contains the boundary contributions from the enclosure flux and $\hat{\mathbf{F}}$ only retains contributions from other boundary conditions and source terms; $\bar{\mathbf{D}}$ indicates

that \mathbf{D} is modified when $\bar{\mathbf{T}}$ is rewritten in terms of the nodal point temperatures. The nonlinear algebraic system in (3.8.11) can be solved with any of the iterative methods described previously, though the best algorithm is a Newton (or Newton-Raphson) scheme that will be discussed below. However, first it is appropriate to list the advantages and disadvantages of this formulation, especially as compared to the scheme in (3.8.9). The disadvantages are that the algebraic system in (3.8.11) is larger than the previous method by the number of surfaces (M) in the enclosure; M can easily be on the order of the number of nodes in the problem. Also, the full matrix \mathbf{A} is part of the overall matrix problem with its attendant memory requirements and increased operation count for solution. An advantage of the algorithm is the strong coupling between the radiation and conduction processes which leads to an improved rate of convergence. Avoiding the matrix inversion of \mathbf{A} is the second and most important advantage. Not only is the solution of \mathbf{A} less computational work than an inversion, but the case of a temperature dependent emissivity is easily accommodated.

A Picard iteration method, such as (3.7.2), converges linearly, at best. When applied to a large or very large system like (3.8.11), successive substitution would not be a cost effective method. A second-order method, like Newton's method, that converges quadratically is a more appropriate choice for highly nonlinear systems such as (3.8.11). Rewriting (3.8.11) in terms of residuals

$$\begin{Bmatrix} \mathbf{R}_T \\ \mathbf{R}_q \end{Bmatrix} = \begin{bmatrix} \hat{\mathbf{K}} & \mathbf{B} \\ -\bar{\mathbf{D}}(\mathbf{T}) & \mathbf{A} \end{bmatrix} \begin{Bmatrix} \mathbf{T} \\ \bar{\mathbf{q}} \end{Bmatrix} - \begin{Bmatrix} \hat{\mathbf{F}} \\ \mathbf{0} \end{Bmatrix} \quad (3.8.12)$$

Newton's method applied to (3.8.12) produces

$$\begin{bmatrix} \mathbf{J}_{TT} & \mathbf{J}_{Tq} \\ \mathbf{J}_{qT} & \mathbf{J}_{qq} \end{bmatrix}^n \begin{Bmatrix} \Delta \mathbf{T} \\ \Delta \bar{\mathbf{q}} \end{Bmatrix}^{n+1} = \begin{bmatrix} \frac{\partial \mathbf{R}_T}{\partial \mathbf{T}} & \frac{\partial \mathbf{R}_T}{\partial \bar{\mathbf{q}}} \\ \frac{\partial \mathbf{R}_q}{\partial \mathbf{T}} & \frac{\partial \mathbf{R}_q}{\partial \bar{\mathbf{q}}} \end{bmatrix}^n \begin{Bmatrix} \Delta \mathbf{T} \\ \Delta \bar{\mathbf{q}} \end{Bmatrix}^{n+1} = - \begin{Bmatrix} \mathbf{R}_T \\ \mathbf{R}_q \end{Bmatrix}^n \quad (3.8.13)$$

where the unknowns are changes in the temperature and uniform radiation flux between the $n + 1$ and n iterations. For completeness, the operational form of Newton's method is written as

$$\begin{bmatrix} \hat{\mathbf{K}}(\mathbf{T}^n) & \mathbf{B} \\ -4\bar{\mathbf{D}}(\mathbf{T}^n) & \mathbf{A} \end{bmatrix} \begin{Bmatrix} \Delta \mathbf{T} \\ \Delta \bar{\mathbf{q}} \end{Bmatrix} = \begin{Bmatrix} -\hat{\mathbf{K}}(\mathbf{T}^n)\mathbf{T}^n - \mathbf{B}\bar{\mathbf{q}}^n + \hat{\mathbf{F}}(\mathbf{T}^n) \\ -\mathbf{A}\bar{\mathbf{q}}^n + \bar{\mathbf{D}}(\mathbf{T}^n)\mathbf{T}^n \end{Bmatrix} \quad (3.8.14)$$

The matrix problem in Eq. (3.8.14) is unsymmetric but can still be solved using an iterative matrix solver (conjugate gradient) with a preconditioner. Further details of this algorithm may be found in Hogan and Gartling [38].

The “fully coupled” algorithms listed above are still not generally applicable to large, geometrically complex, three-dimensional problems because of the significant memory and/or computational burden of the formulation. In order to reduce the computational requirements a decoupled procedure must be considered. The simplest method is a staggered or cyclic iteration between the two equation sets. Typically, the finite element conduction solution is advanced a time step or iteration, at which point a new set of surface temperatures are available. After reducing the finite element surface temperature distribution to a set of constant surface temperatures, Eq. (3.8.5) can be solved for the net surface flux. The net surface flux

provides boundary conditions to the next finite element solution cycle. Alterations to this basic scheme might include multiple correction steps with either fixed or the updated radiative flux or the fixing of the radiative flux for several steps if the temperature changes are small. For steady problems, this cyclic procedure can be written using (3.8.5) as

$$\mathbf{A}\bar{\mathbf{q}}^{n+1} = \bar{\mathbf{D}}(\bar{\mathbf{T}}^n)\bar{\mathbf{T}}^n \quad (3.8.15)$$

$$\hat{\mathbf{K}}(\mathbf{T}^n)\mathbf{T}^{n+1} = \hat{\mathbf{F}}(\mathbf{T}^n) - \mathbf{B}\bar{\mathbf{q}}^{n+1} \quad (3.8.16)$$

The cyclic solution procedure is a standard approach for this type of problem though it is not without its drawbacks. When the simulation is time dependent this type of algorithm is usually very successful, since the time stepping procedure usually produces only “small” changes in the temperature field within each time step. Incremental changes in surface temperature lead to incremental changes in surface flux, and rapid convergence of both the radiation and conduction solutions, i.e., the two equation sets are always “close” to equilibrium. Adaptive time step, predictor-corrector techniques can be used to advantage with this decoupled approach with the radiation solution being evaluated after the predictor step.

For steady solutions that are approached iteratively, the decoupled algorithm tends to be less successful. Large changes in the temperature field between iterations are typical and can easily be amplified by large variations in surface flux due to the strong (fourth power) dependence on surface temperature. Very slow convergence, or more often, divergence, of the algorithm is observed. Severe underrelaxation of the solution vectors [see Eqs. (3.7.4a,b)] may improve the solution process as will incrementation of the boundary conditions and/or material properties. In some cases, the most cost-effective method for steady enclosure radiation problems is the decoupled method with a time-dependent approach (false transient) to steady state.

A typical method for improving the convergence of the cyclic procedure is to make the algorithm more implicit. A semi-implicit method developed from the fully coupled Newton’s method was proposed in [41,42] and also studied in [38]. Segregating the equations in (3.8.14) and simplifying to directly produce updates at $n + 1$ yields

$$\mathbf{A}\bar{\mathbf{q}}^{n+1} = \bar{\mathbf{D}}(\bar{\mathbf{T}}^n)\bar{\mathbf{T}}^n \quad (3.8.17)$$

$$\hat{\mathbf{K}}(\mathbf{T}^n)\mathbf{T}^{n+1} + \mathbf{B}(\bar{\mathbf{q}}^{n+1} - \bar{\mathbf{q}}^n) = \hat{\mathbf{F}}(\mathbf{T}^n) - \mathbf{B}\bar{\mathbf{q}}^n \quad (3.8.18)$$

The radiation equation is a simple Picard update for the surface fluxes. The $\Delta\bar{\mathbf{q}}$ term in the conduction equation can be expanded in a Taylor series as

$$\Delta\bar{\mathbf{q}} = (\bar{\mathbf{q}}^{n+1} - \bar{\mathbf{q}}^n) = \frac{\partial\bar{\mathbf{q}}}{\partial\mathbf{T}}|_{T^n}(\mathbf{T}^{n+1} - \mathbf{T}^n) \quad (3.8.19)$$

To define the derivative in (3.8.19) the net flux is written in terms of the black-body emissive power and the irradiation as

$$\bar{\mathbf{q}}^{n+1} = \epsilon\mathbf{E}_b^{n+1} - \epsilon\mathbf{G}^{n+1} = \epsilon\sigma(\bar{\mathbf{T}}^{n+1})^4 - \epsilon\mathbf{G}^{n+1} \quad (3.8.20)$$

where \mathbf{G} is the far-field irradiation. Assuming that the variation of \mathbf{G} between iterations is relatively small and linearizing the black-body emissive power then

$$\bar{\mathbf{q}}^{n+1} \approx \epsilon\sigma(\bar{\mathbf{T}}^n)^3\bar{\mathbf{T}}^{n+1} - \epsilon\mathbf{G}^n = \bar{h}_r(\bar{\mathbf{T}}^n)\mathbf{T}^{n+1} - \bar{\mathbf{q}}_G^n \quad (3.8.21)$$

Using this result in the Taylor series then

$$\Delta \bar{\mathbf{q}} = \left(4\epsilon\sigma(\bar{\mathbf{T}})^3 - \frac{\partial\epsilon\mathbf{G}}{\partial\mathbf{T}}\right)(\mathbf{T}^{n+1} - \mathbf{T}^n) \approx (4\epsilon\sigma(\bar{\mathbf{T}}^n)^3)(\mathbf{T}^{n+1} - \mathbf{T}^n) \quad (3.8.22)$$

where the variation in \mathbf{G} is neglected. Substituting the above result into the conduction equation yields

$$[\hat{\mathbf{K}}(\mathbf{T}^n) + 4\mathbf{B}\bar{h}_r]\mathbf{T}^{n+1} = \hat{\mathbf{F}}(\mathbf{T}^n) + 3\mathbf{B}\bar{h}_r\mathbf{T}^n + \mathbf{B}\bar{\mathbf{q}}_G^n \quad (3.8.23)$$

The cyclic solution of (3.8.17) and (3.8.23) are designated the segregated semi-implicit Newton (SSN) method and can be demonstrated to have better convergence properties than the simple cyclic algorithm in (3.8.15)–(3.8.16). The use of an approximate Jacobian in (3.8.23) is responsible for the improvement in convergence. A Picard version of this algorithm was identified in [38] but has not found much use in application because the SSN is better at virtually the same computational cost.

The last part of the radiation solution that requires comment is the view factor matrix, \mathbf{F}_{vf} . View factors for use in enclosure radiation computations were defined in Section 1.10 as purely geometric factors. The computation of such factors is usually performed by specialized programs (e.g., see [43–45]) that are external to the basic finite element program. For plane, two-dimensional geometries with no obstructing (shadowing) surfaces, the view factor computation can be done with simplified analytic methods such as Hottel's cross-string method [46]. More complex geometries in two and three dimensions require the use of numerical procedures, such as double area integration, hemicube, or Monte Carlo techniques. A comparative study of a number of view factor algorithms is presented in [47].

3.9 Variable Properties

In all of the previous discussion we have mentioned the possible occurrence of variable material properties, boundary conditions, and heat sources. The previous sections on solution methods showed how such temperature dependence produced nonlinearities in the matrix problem and how various solution algorithms were set up to account for this behavior. In the present section three other aspects of variable properties will be discussed: methods for the implementation of variable coefficients into the finite element matrix formulation, the treatment of latent heat effects during phase change, and the use of anisotropic conductivities.

3.9.1 Temperature Dependent Properties

Consider first the problem of evaluating finite element matrices that contain a variable coefficient. To focus the discussion, a typical diffusion term from the conduction equation will be examined. From Eq. (3.2.6) the xx -component of the diffusion matrix is

$$\mathbf{K}_{xx} = \int_{\Omega^e} \frac{\partial\Psi}{\partial x} k \frac{\partial\Psi^T}{\partial x} d\mathbf{x} \quad (3.9.1)$$

where for simplicity an isotropic conductivity has been assumed. As seen in Section 3.4, it is typical to evaluate element integrals, such as the one in Eq. (3.9.1), by use of numerical quadrature. When the conductivity is constant the matrix in (3.9.1)

is independent of temperature and may be constructed once and stored as a two-dimensional array for subsequent use in a solution procedure. However, when the conductivity is variable, the matrix in (3.9.1) must be reevaluated each time new values of the temperature are produced during the solution procedure. One standard method for treating this variable coefficient simply involves recomputing the matrix for each new set of conductivities. For each element the temperature is evaluated at the quadrature points via the basis functions, thus permitting the conductivity to be evaluated at the quadrature locations. The matrix coefficients are then computed by the standard quadrature procedure. The work involved in such a procedure can become large when considering anisotropic conductivities (three conductivity matrices in two dimensions, six conductivity matrices in three dimensions) and/or higher-order elements (higher-order quadrature schemes).

A second approach to this problem makes use of an interpolation function for the variable coefficient. Let the conductivity be approximated by

$$k = \Psi^T \mathbf{k} \quad (3.9.2)$$

where Ψ is an appropriate set of element basis functions and \mathbf{k} are nodal point values of the conductivity. Inserting (3.9.2) into (3.9.1) gives

$$\mathbf{K}_{xx} = \int_{\Omega_e} \frac{\partial \Psi}{\partial x} (\Psi^T \mathbf{k}) \frac{\partial \Psi^T}{\partial x} dx \quad (3.9.3)$$

The above integral may be evaluated (via quadrature) once and stored as a triple subscripted array (hypermatrix). During the solution process, the conductivity is evaluated at each node as a function of the nodal point temperature, and a simple inner product in (3.9.3) produces the needed two-dimensional diffusion matrix. Such a procedure eliminates the need for recomputing element-level matrices, albeit at the expense of some additional storage and I/O operations.

Though we are concerned in the present example with the treatment of variable material properties it is important to realize that both of the above procedures can be used with any type of variable coefficient. In the heat conduction problem variable heat capacities, convective and radiative heat transfer coefficients, and volumetric heat sources are all candidates for either of the described methods.

3.9.2 Phase Change Properties

One particularly difficult type of property variation is the latent heat effect that occurs in the simulation of phase change problems. As described in Section 1.11, latent heat effects can be adequately treated by use of an enthalpy method with a temperature-dependent specific heat. That is, from Eq. (1.11.6) we have

$$C^*(T) = C(T) + L \delta(T - T_t) \quad (3.9.4)$$

where C^* is the effective specific heat, C is the normal specific heat, L is the latent heat, δ is the Dirac delta function, and T_t is the transition temperature. For application in a finite element procedure, Eq. (3.9.4) is replaced by

$$C^*(T) = C(T) + L \delta^*(T - T_t, \Delta T) \quad (3.9.5)$$

where δ^* is the delta form function; δ^* has a large but finite value in the interval centered about T_t and is zero outside the interval (see Figure 3.9.1).

The interval ΔT is often referred to as the “mushy” zone and corresponds to the difference between the liquidus, T_l , and solidus, T_s , temperatures for the material. Note that Eq. (3.9.5) is thus an approximation for the behavior of pure materials that change phase at a specified temperature, T_t , but is accurate for non-pure substances that have truly distinct liquidus and solidus temperatures.

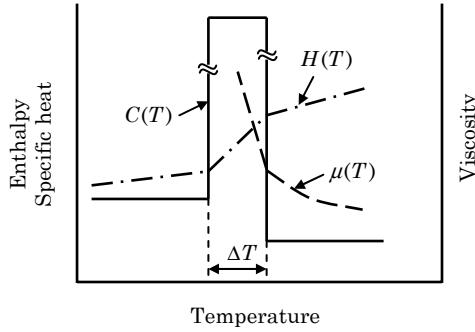


Figure 3.9.1: Definition of material properties for phase change computation.

Though the effective capacitance model described above is useful in accounting for latent heat effects, a few words of caution should be issued with regard to the time integration of this type of phase change model. Since the transition temperature interval, ΔT , is often small compared to the overall temperature variation in a conduction problem, it is important to note that there are some severe practical limitations on the time integration procedure. In general, the time-stepping algorithm must be controlled such that every node that “changes phase” is forced to attain a temperature value in the interval bracketed by ΔT . If a nodal point does not “land” in the ΔT range but simply steps over this temperature interval, the latent heat effect is lost for that node and an incorrect temperature response and energy balance will result. Various methods for dealing with this difficulty have been proposed (see [48,49]) including broadening the ΔT range, placing a limit on the maximum temperature change that can occur during a time step, and using special forms of a predictor-corrector algorithm. The problem is a numerically delicate one and requires experience and care for the accurate analysis of complex applications.

A number of useful alternatives within the effective capacitance method make direct use of the enthalpy, H , versus temperature curve, and differ only in how they evaluate the slope of H versus T . In one method, H is first computed at the nodes of an element (knowing T); the heat capacity at an integration point within the element is then found from

$$C_p = \left[\frac{\nabla H \cdot \nabla H}{\nabla T \cdot \nabla T} \right]^{1/2} \quad (3.9.6)$$

where ∇H and ∇T are evaluated via the element shape functions. This technique will maintain its accuracy as long as the phase boundary passes through each element

and does not skip over an element. This is certainly a more robust technique than the standard effective capacitance method.

In Section 1.11 the variable heat source method for phase change problems was also outlined. The time and temperature dependent heat source representing the latent heat was given as

$$Q_{lh} = \rho L \delta(T - T_t) \frac{\partial T}{\partial t} \quad (3.9.7)$$

which can be rewritten using some of the previous definitions as

$$Q_{lh} = \rho L \delta^*(T - T_t, \Delta T) = \rho C_{lh} \frac{\partial T}{\partial t} \quad (3.9.8)$$

The effective capacitance C_{lh} is evaluated by the same methods as described above and is subject to many of the same time integration difficulties as the previous algorithm. By treating the latent heat as a source term, the overall energy balance can be more easily maintained. It is straightforward to determine which nodes pass through the ΔT interval as a result of the time step and then construct the correct source term for the subsequent time step. The integration process would typically lag the evaluation of the source term anyhow, since the temperature rate is difficult to predict. Though the energy balance and individual nodal temperature response are easier to control, the temporal accuracy of the method may be limited.

3.9.3 Anisotropic Properties

In closing this section, we mention briefly the treatment of anisotropic thermal conductivities. In two dimensions, the most general physically realistic, material model is assumed to have an orthotropic conductivity, i.e., the components $k_{11} \neq k_{22}$ and $k_{12} = k_{21} = 0$ when referred to the principal material axes (x_1, x_2, x_3). Since the global finite element coordinate frame will generally not be aligned with the material axes, a coordinate transformation is required. Referring to the sketch in Figure 3.9.2 the global components of the conductivity matrix, being the components of a second-order tensor, can be obtained from (see Reddy [50], pp. 108–112)

$$\begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} k_{11} & 0 \\ 0 & k_{22} \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \quad (3.9.9a)$$

or when multiplied out ($k_{xy} = k_{yx}$)

$$[k] = \begin{bmatrix} k_{11} \cos^2 \theta + k_{22} \sin^2 \theta & -k_{11} \sin \theta \cos \theta + k_{22} \sin \theta \cos \theta \\ -k_{11} \sin \theta \cos \theta + k_{22} \sin \theta \cos \theta & k_{11} \sin^2 \theta + k_{22} \cos^2 \theta \end{bmatrix} \quad (3.9.9b)$$

These last expressions give the needed conductivity components for use in the finite element diffusion matrix \mathbf{K} defined in Eq. (3.2.5a). In the most general two-dimensional case, three matrices are defined by \mathbf{K} for the xx , yy , and xy components of the conductivity tensor as given in Eq. (3.9.9a). It is of course possible for the principal material conductivity components, k_{11} and k_{22} , to be independent functions of temperature, and for the local orientation angle, θ , to vary from one material to another and from element to element in the finite element mesh.

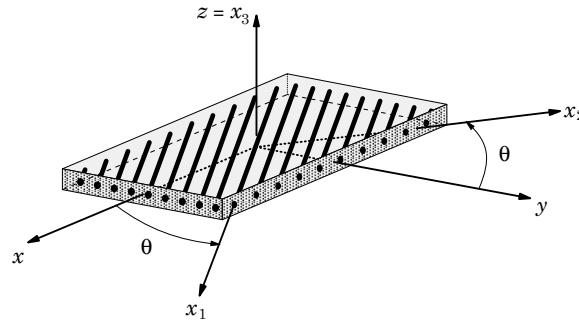


Figure 3.9.2. Notation for anisotropic conductivity.

The three-dimensional transformation for an orthotropic material has the same form but is slightly more complex. In tensor notation, the conductivity components in the global coordinate system are obtained from

$$k_{ij} = C_{in} k_{nm}^p C_{jm} \quad (3.9.10)$$

where k_{nm}^p is the conductivity tensor in the material coordinate frame and C_{in} are the direction cosines between the axes of the two coordinate systems. For an orthotropic material, k_{nm}^p is diagonal and (3.9.10) can be written as

$$[k] = \begin{bmatrix} C_{11} & C_{21} & C_{31} \\ C_{12} & C_{22} & C_{32} \\ C_{13} & C_{23} & C_{33} \end{bmatrix} \begin{bmatrix} k_{11} & 0 & 0 \\ 0 & k_{22} & 0 \\ 0 & 0 & k_{33} \end{bmatrix} \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} = \mathbf{C}^T \mathbf{k} \mathbf{C} \quad (3.9.11)$$

which has the same form as the two-dimensional case in (3.9.9). Note that the three-dimensional form of k_{ij} now requires six matrices to define the diffusion operator \mathbf{K} ; the principal conductivities may again be temperature dependent and their orientation may vary over the domain.

As a final note on representing anisotropic properties, the use of low-order, simplex elements (triangles and tetrahedrons) is strongly discouraged. The limited derivative capability of these elements leads to very poor accuracy for anisotropic material models. This lack of accuracy persists even with highly refined meshes. Low-order quadrilateral and hexahedral elements have far superior performance in this application though highly deformed (stretched) elements are still problematic. Higher order elements of all shapes are generally preferred and more reliable for modeling anisotropy.

3.10 Post-Processing Operations

3.10.1 Heat Flux

An important post-processing operation in many heat conduction problems involves the computation of the heat flux from various surfaces or bodies. The diffusion flux associated with the conduction equation can be computed on an element-by-element basis. In the following only the heat flux associated with two-dimensional

problems will be specifically considered since the flux in the three-dimensional case is completely analogous. Fourier's law provides the definition of the conductive heat flux as

$$q_x = -\left(k_{xx}\frac{\partial T}{\partial x} + k_{xy}\frac{\partial T}{\partial y}\right), \quad q_y = -\left(k_{yx}\frac{\partial T}{\partial x} + k_{yy}\frac{\partial T}{\partial y}\right) \quad (3.10.1)$$

The flux components in Eq. (3.10.1) can be computed at time t and location x_i using the standard finite element approximation for T

$$T(x_i, t) = \Psi^T(x_i)\mathbf{T}(t) \quad (3.10.2)$$

and the components k_{xx} , $k_{xy} = k_{yx}$, and k_{yy} of the conductivity tensor, \mathbf{k} . Also needed are the relations for the global derivatives of the interpolation functions in terms of their derivatives with respect to the element local coordinates [see Eqs. (2.11.4)–(2.11.6)]. That is,

$$\begin{Bmatrix} \frac{\partial \Psi_i}{\partial x} \\ \frac{\partial \Psi_i}{\partial y} \end{Bmatrix} = \begin{bmatrix} J_{11}^* & J_{12}^* \\ J_{21}^* & J_{22}^* \end{bmatrix} \begin{Bmatrix} \frac{\partial \Psi_i}{\partial \xi} \\ \frac{\partial \Psi_i}{\partial \eta} \end{Bmatrix} \quad (3.10.3a)$$

where J_{ij}^* are the elements of the inverse of the Jacobian matrix [see Eq. (2.11.5)] and are given by

$$J_{11}^* = \frac{1}{J} \frac{\partial y}{\partial \eta}, \quad J_{12}^* = -\frac{1}{J} \frac{\partial y}{\partial \xi}, \quad J_{21}^* = -\frac{1}{J} \frac{\partial x}{\partial \eta}, \quad J_{22}^* = \frac{1}{J} \frac{\partial x}{\partial \xi}, \quad J = J_{11}J_{22} - J_{12}J_{21} \quad (3.10.3b)$$

Note that (x, y) are functions of the local (element) coordinates, (ξ, η) through the parametric mapping (3.3.3) (see Sections 2.11, 3.4 and 3.5).

Using these definitions, the heat flux components can be expressed as

$$\begin{aligned} q_x &= -k_{xx} \left(J_{11}^* \frac{\partial \Psi^T}{\partial \xi} \mathbf{T} + J_{12}^* \frac{\partial \Psi^T}{\partial \eta} \mathbf{T} \right) \\ &\quad - k_{xy} \left(J_{21}^* \frac{\partial \Psi^T}{\partial \xi} \mathbf{T} + J_{22}^* \frac{\partial \Psi^T}{\partial \eta} \mathbf{T} \right) \end{aligned} \quad (3.10.4a)$$

$$\begin{aligned} q_y &= -k_{yx} \left(J_{11}^* \frac{\partial \Psi^T}{\partial \xi} \mathbf{T} + J_{12}^* \frac{\partial \Psi^T}{\partial \eta} \mathbf{T} \right) \\ &\quad - k_{yy} \left(J_{21}^* \frac{\partial \Psi^T}{\partial \xi} \mathbf{T} + J_{22}^* \frac{\partial \Psi^T}{\partial \eta} \mathbf{T} \right) \end{aligned} \quad (3.10.4b)$$

The definitions in (3.10.4a,b) are sufficient to define the flux components at any point (ξ_0, η_0) in the element; for optimal accuracy, the 2×2 Gauss points within the element are usually selected for the evaluation of derivatives. Extrapolation and averaging of flux values at the nodes is usually employed if a continuous flux field is required.

In addition to the local components of the flux vector, the heat flux normal to the element edge is often of importance. By definition

$$q_n = \mathbf{q} \cdot \hat{\mathbf{n}} \quad (3.10.5a)$$

$$\mathbf{q} = q_x \hat{\mathbf{e}}_x + q_y \hat{\mathbf{e}}_y \quad (3.10.5b)$$

$$\hat{\mathbf{n}} = n_x \hat{\mathbf{e}}_x + n_y \hat{\mathbf{e}}_y \quad (3.10.5c)$$

and thus

$$q_n = q_x n_x + q_y n_y \quad (3.10.6)$$

In order to employ Eq. (3.10.6), the components of the normal vector are also required; these may also be computed from the element geometry and isoparametric mapping

$$n_x = \frac{1}{\Delta} \frac{\partial y}{\partial s} ; \quad n_y = -\frac{1}{\Delta} \frac{\partial x}{\partial s} \quad (3.10.7)$$

where the Jacobian of the transformation is

$$|\mathbf{J}_s| = \Delta = \left[\left(\frac{\partial x}{\partial \xi} \right)^2 + \left(\frac{\partial y}{\partial \xi} \right)^2 \right]^{\frac{1}{2}} \quad (3.10.8)$$

and ξ is the local coordinate along the edge of the element.

Relations analogous to the above may be derived for the three-dimensional case where the flux normal to an element surface is to be evaluated. The flux component definitions extend naturally with the inclusion of the z component. The normal to the element surface is defined by

$$\hat{\mathbf{n}} = \frac{\mathbf{e}_1 \times \mathbf{e}_2}{|\mathbf{J}_s|} \quad (3.10.9)$$

where $|\mathbf{J}_s| = |\mathbf{e}_1 \times \mathbf{e}_2|$ and the surface vectors \mathbf{e}_i were defined in terms of the surface shape functions in Eq. (3.5.4).

3.10.2 Heat Flow Function

For two-dimensional problems, Kimura and Bejan [51] have proposed the use of a heat flow function to assist in the visualization of energy transport. The heat function is directly analogous to the stream function for incompressible fluid flow and is constructed to satisfy the steady, source free form of the energy equation. In formal terms, the heat function \mathcal{H} is the remaining nonzero component of a vector potential that identically satisfies a form of Eq. (3.1.1). Though the heat flow function can be used to describe both advective and diffusive processes, it will be developed here only for the conduction equation. By definition

$$q_x = -k \frac{\partial T}{\partial x} = \frac{\partial \mathcal{H}}{\partial y}, \quad q_y = -k \frac{\partial T}{\partial y} = -\frac{\partial \mathcal{H}}{\partial x} \quad (3.10.10)$$

For simplicity the definitions in (3.10.10) have also assumed an isotropic conductivity though this is not a required restriction. By limiting attention to the heat conduction process, the heat function reduces to the definition for a heat flux line, i.e., a line that is everywhere tangent to the local flux vector. The change in the heat function is an exact differential such that

$$\delta \mathcal{H} = \int_A^B \mathbf{q} \cdot \hat{\mathbf{n}} d\Gamma \quad (3.10.11)$$

where $\hat{\mathbf{n}}$ is the unit vector normal to the integration path $d\Gamma$ and \mathbf{q} is the heat flux vector on the path which are defined in (3.10.5c) and (3.10.5b), respectively.

The computation of the change in the heat function within a finite element can be accomplished using Eq. (3.10.11) once a suitable integration path AB has been identified. An obvious choice for the integration path is along the two-dimensional element boundaries. Defining an edge interpolation for the flux components

$$q_x = \hat{\Psi}^T \mathbf{q}_x ; \quad q_y = \hat{\Psi}^T \mathbf{q}_y \quad (3.10.12)$$

The relation for the elemental line segment is

$$d\Gamma = \Delta d\xi \quad (3.10.13)$$

where Δ is defined by the parametric mapping of the element edge given in Eq. (3.10.8). The incremental change in the heat function along an element edge can then be formulated as

$$\delta\mathcal{H} = \int_{-1}^{+1} \left[\left(\frac{\partial \hat{\Psi}}{\partial s}^T \mathbf{y} \right) \hat{\Psi}^T \mathbf{q}_x - \left(\frac{\partial \hat{\Psi}}{\partial s}^T \mathbf{x} \right) \hat{\Psi}^T \mathbf{q}_y \right] d\xi \quad (3.10.14)$$

The change in the heat function along any element boundary can be computed from Eq. (3.10.14) once the element geometry and temperature fields are specified; the fluxes needed in (3.10.14) are derived from the definitions in the previous section. Computation of the heat function field for an entire finite element mesh is generated by applying (3.10.14) along successive element boundaries, starting at a node for which a base value of \mathcal{H} has been specified. Observe that by applying (3.10.14) to all the edges of a single element and summing the increments in \mathcal{H} , an estimate of the energy balance for the element can be obtained.

The calculation of the heat function for axisymmetric geometries follows a similar procedure with the appropriate definition for \mathcal{H} being

$$q_r = \frac{\partial \mathcal{H}}{\partial z} ; \quad q_z = -\frac{\partial \mathcal{H}}{\partial r} \quad (3.10.15)$$

and

$$\mathbf{q} = q_r \hat{\mathbf{e}}_r + q_z \hat{\mathbf{e}}_z, \quad \hat{\mathbf{n}} = n_r \hat{\mathbf{e}}_r + n_z \hat{\mathbf{e}}_z \quad (3.10.16)$$

3.11 Advanced Topics in Conduction

3.11.1 Introduction

The methods and algorithms described in the previous sections will allow the vast majority of heat conduction problems to be solved accurately and efficiently. However, as finite element methods are used on problems of increasing complexity, more specialized procedures are sometimes required. In the following sections a few such techniques are described which allow the standard conduction problem to be extended to more complex geometries or more involved physical modeling.

3.11.2 Specialty Elements

The two- and three-dimensional elements described in Section 3.3 are the workhorse elements for describing and solving heat conduction problems in solid bodies or continuum regions. In many applications, special geometric features appear that are not well described by a full continuum element. Two examples of these specialty elements are bars (beams, trusses, wires, cables, etc.) and shells (membranes, panels, fins, etc.) in both two and three dimensions.

The standard three-dimensional bar element may be either a two-node or three-node, isoparametric element as shown in Figure 3.11.1. This element has a variable cross-sectional area with conduction only allowed along the axis of the element. Note that the shape of the cross-section need not be explicitly defined here, though for purposes of boundary condition application, some convention must be established. The simplest assumption is a circular cross-section with flux type boundary conditions then being applied uniformly around the circumference or uniformly over the ends of the bar. Because the surface of the bar is generally not described as a faceted surface, the computation of radiation view factors is generally not possible for this element; inclusion in the enclosure radiation problem is thus prohibited.

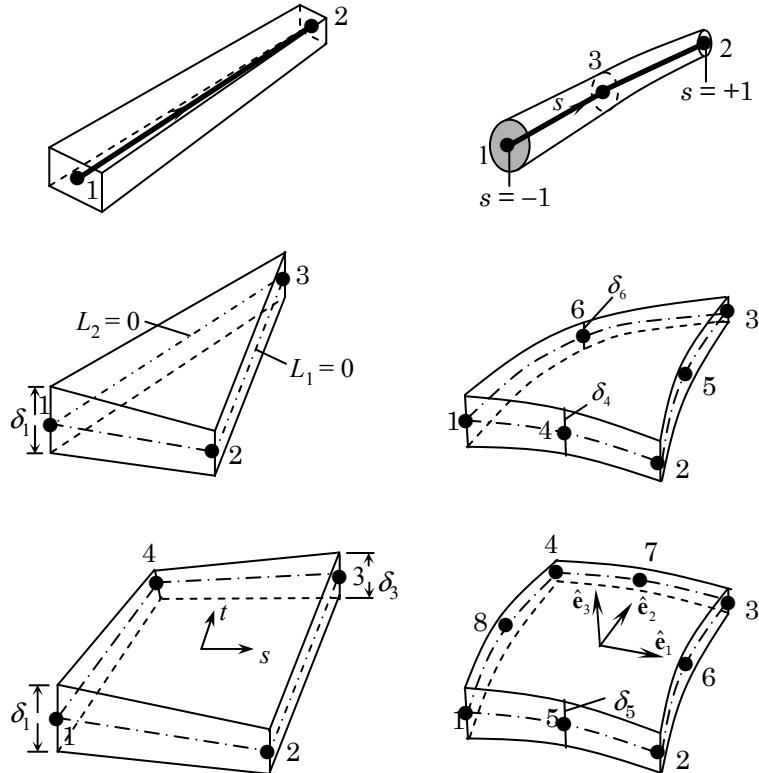


Figure 3.11.1: Three-dimensional bar and shell elements.

The shape function for the two-node element is defined by

$$\{\Psi^e\} = \frac{1}{2} \begin{Bmatrix} (1 - \xi) \\ (1 + \xi) \end{Bmatrix} \quad (3.11.1)$$

and the three-node element is described by

$$\{\Psi^e\} = \frac{1}{2} \begin{Bmatrix} (1 - \xi)\xi \\ 2(1 - \xi^2) \\ (1 + \xi)\xi \end{Bmatrix} \quad (3.11.2)$$

The functions in Eqs. (3.11.1) and (3.11.2) are ordered as shown in the figure and are written in terms of the normalized coordinate ξ that varies from -1 to $+1$. The parametric mapping given in Eq. (3.3.3) still holds and relates the global coordinates (x, y, z) for the element to the local coordinate, ξ .

Bar elements for two-dimensional problems can also be defined by the shape functions in Eqs. (3.11.1) and (3.11.2). In this case, the transformation is carried out from the (x, y) coordinates to the local coordinate ξ . The variable, cross-sectional area for the two-dimensional case reduces to a variable thickness with unit depth. The axisymmetric, two-dimensional bar can be treated in a similar manner, though it is rotated about the z axis. In both cases, these “bar” elements should be thought of as a one-dimensional conduction element in the plane of the problem. These elements are essentially two-dimensional shell elements and may be used as such. Unlike the three-dimensional bar, these two-dimensional elements can be included in a view factor computation since they have “sides” just like the continuum elements. Boundary condition application is straightforward for these element types.

There are a number of different types of shell elements. The main differences between shell definitions are in the planform shape and the temperature approximation through the thickness of the shell. For strictly thermal applications, a standard assumption is that the shell has no temperature variation through the thickness; conduction is only allowed in the plane of the element. For thermal-stress applications it is desirable to have a good representation for the thermal gradient through the thickness of the shell and this requires a higher-order temperature approximation normal to the plane of the element. These higher-order temperature representations are generally difficult to use with continuum elements because of the need to constrain or tie the multiple temperature nodes that occur through the shell thickness to a single node in the adjacent continuum element.

Three-dimensional shell elements with both triangular and quadrilateral planforms are in common use. Four typical element types are shown in Figure 3.11.1. Each element is assumed to have a constant temperature through its thickness. All elements usually allow the shell thickness to vary across the planform. The temperature shape function for the three-node, triangular element is defined by

$$\{\Psi^e\} = \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \end{Bmatrix} \quad (3.11.3)$$

and the six-node, triangular shell element has the following shape functions

$$\{\Psi^e\} = \begin{Bmatrix} L_1(2L_1 - 1) \\ L_2(2L_2 - 1) \\ L_3(2L_3 - 1) \\ 4L_1L_2 \\ 4L_2L_3 \\ 4L_3L_1 \end{Bmatrix} \quad (3.11.4)$$

where the L_i are the standard, in-plane area coordinates that vary from 0 to +1. The four-node, quadrilateral shell element has shape functions of the form

$$\{\Psi^e\} = \frac{1}{4} \begin{Bmatrix} (1 - \xi)(1 - \eta) \\ (1 + \xi)(1 - \eta) \\ (1 + \xi)(1 + \eta) \\ (1 - \xi)(1 + \eta) \end{Bmatrix} \quad (3.11.5)$$

while the eight-node, “serendipity” shell element is defined by

$$\{\Psi^e\} = \frac{1}{4} \begin{Bmatrix} (1 - \xi)(1 - \eta)(-\xi - \eta - 1) \\ (1 + \xi)(1 - \eta)(\xi - \eta - 1) \\ (1 + \xi)(1 + \eta)(\xi + \eta - 1) \\ (1 - \xi)(1 + \eta)(-\xi + \eta - 1) \\ 2(1 - \xi^2)(1 - \eta) \\ 2(1 + \xi)(1 - \eta^2) \\ 2(1 - \xi^2)(1 + \eta) \\ 2(1 - \xi)(1 - \eta^2) \end{Bmatrix} \quad (3.11.6)$$

and the normalized (ξ, η) coordinates vary from -1 to +1. The shape functions defined in Eqs. (3.11.3)–(3.11.6) are recognized as being identical to the interpolation functions for the two-dimensional triangular and quadrilateral continuum elements. Though the interpolation of temperature within the plane of the elements is similar, the geometric representation of the continuum elements and the shell elements is quite different. The parametric mapping for any shell element is accomplished with the following definitions

$$\begin{aligned} x &= \hat{\Psi}^T \mathbf{x} + \frac{1}{2} r \hat{\Psi}^T \delta \hat{\mathbf{e}}_3 \cdot \hat{\mathbf{e}}_x \\ y &= \hat{\Psi}^T \mathbf{y} + \frac{1}{2} r \hat{\Psi}^T \delta \hat{\mathbf{e}}_3 \cdot \hat{\mathbf{e}}_y \\ z &= \hat{\Psi}^T \mathbf{z} + \frac{1}{2} r \hat{\Psi}^T \delta \hat{\mathbf{e}}_3 \cdot \hat{\mathbf{e}}_z \end{aligned} \quad (3.11.7)$$

where $\hat{\Psi}$ is the appropriate linear or quadratic interpolation within the plane. The $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ are vectors of coordinates for the midplane nodes of the element, r is the normalized coordinate along the normal to the element midplane and δ is a vector of thickness values at the nodes. The vectors $\hat{\mathbf{e}}_1$ and $\hat{\mathbf{e}}_2$ are defined as being tangent to the curvilinear coordinates (ξ, η) on the element midplane; $\hat{\mathbf{e}}_3$ is normal to the element midplane and is defined by $\hat{\mathbf{e}}_3 = \hat{\mathbf{e}}_2 \times \hat{\mathbf{e}}_1$. The unit vectors $(\hat{\mathbf{e}}_x, \hat{\mathbf{e}}_y, \hat{\mathbf{e}}_z)$

define the orientation of the global coordinate system. Note that, in general, $\hat{\mathbf{e}}_3$ varies over the planform of the element [$\hat{\mathbf{e}}_3(\xi, \eta)$ or $\hat{\mathbf{e}}_3(L_1, L_2)$] and this variation must be accounted for in the construction of the Jacobian entries for the element mapping procedure. The element matrices for these elements are constructed using the same type of procedures as are used for continuum elements. For the constant temperature through the thickness shells, a numerical quadrature in the plane of the element is sufficient. If a higher-order temperature through the thickness is employed, the shape functions in Eqs. (3.11.3)–(3.11.6) would be altered and a full three-dimensional quadrature would be required.

Allowable boundary conditions for shell elements are basically the same as for the three-dimensional continuum elements since all “sides” of the shell are well defined. Radiation enclosures may also contain shells because each shell face is a polygon that can be easily incorporated in the view factor computation. The edge descriptions that use a midline and thickness distribution are different from the planform faces that use nodes to describe the geometry; both cases could be handled by a general view factor algorithm.

3.11.3 Computational Boundary Conditions

The finite element implementation of all of the standard boundary conditions for a heat conduction problem were covered in Section 3.5. However, it is sometimes necessary to consider other implementations of the standard boundary conditions for special simulation applications. Generally these applications involve material interfaces and require more computational effort to produce a usable physical model for the heat transfer process.

3.11.3.1 Contact boundary conditions

The surface flux that was considered in Section 3.5 was derived from boundary conditions that are applied to the external boundaries of the heat conduction problem. As described in Section 1.10.3 of Chapter 1, it is also appropriate to consider “internal” flux conditions associated with a material interface and in particular, surfaces for which thermal contact resistance is important. The computational form for this internal or contact boundary condition is derived in the same manner as presented previously, though the work to obtain all of the needed data is increased.

From Eq. (1.10.12) the internal or gap heat flux between two surfaces is given by

$$q_{\text{contact}} = h_{\text{contact}}(s_k, T_{\text{contact}}, t)(T_M - T_S) \quad (3.11.8)$$

where h_{contact} is an effective heat transfer coefficient for the contacting surfaces, and T_{contact} is an average temperature between the master surface temperature, T_M , and the slave surface temperature, T_S . This being of the same form as the flux conditions in (3.1.2c) the finite element form (flux vector) for the boundary condition is

$$\mathbf{q}_{\text{cnt}}(T) = \oint_{\Gamma_m} \hat{\Psi} q_{\text{contact}} d\mathbf{s} = \oint_{\Gamma_m} \hat{\Psi} h_{\text{contact}}(T_M - T_S) d\mathbf{s} \quad (3.11.9)$$

or in matrix form (with interpolation for the temperature)

$$\begin{aligned}\mathbf{q}_{\text{cnt}} &= \left(\int_{-1}^1 \int_{-1}^1 h_{\text{contact}}(\xi, \eta) \hat{\Psi}(\xi, \eta) \hat{\Psi}^T(\xi, \eta) |\mathbf{J}_s| d\xi_s d\eta_s \right) \mathbf{T}_M \\ &\quad - \int_{-1}^1 \int_{-1}^1 h_{\text{contact}}(\xi, \eta) \hat{\Psi}(\xi, \eta) T_S(\xi, \eta) |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.11.10a)\end{aligned}$$

$$= \mathbf{G} \mathbf{T}_M - \mathbf{F}_{\text{cnt}} \quad (3.11.10b)$$

where

$$\mathbf{G} = \int_{-1}^1 \int_{-1}^1 h_{\text{contact}} \hat{\Psi} \hat{\Psi}^T |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.11.11a)$$

$$\mathbf{F}_{\text{cnt}} = \int_{-1}^1 \int_{-1}^1 h_{\text{contact}} T_S \hat{\Psi} |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.11.11b)$$

The numerical implementation of this condition requires that master and slave sides of the contact surface be defined. Because unknown temperatures occur on both sides of the interface, each contact surface must be processed in turn as a master surface to satisfy the energy balance; the opposite or slave surface provides the reference temperature for heat transfer across the surface. For generality, the situation shown in the two-dimensional sketch of Figure 3.11.2 is considered, where the nodes and elements on each side of the contact surface are not aligned. If a node on the master surface does not have an image on the slave surface, then h_{contact} is set to zero for that location and the contribution to the contact heat flux for that node is neglected.

In developing Eqs. (3.11.10) and (3.11.11) the contact coefficient h_{contact} is assumed to vary in a known manner over the contact surface. The vector \mathbf{T}_M corresponds to unknown nodal point temperatures on the master surface and the coefficient matrix \mathbf{G} is combined with the diffusion matrix \mathbf{K} during the solution process [see Eq. (3.6.4)]. The temperatures denoted by T_S are not generally nodal point temperatures but rather interpolated temperatures from the slave surface, adjacent to the master surface nodes.

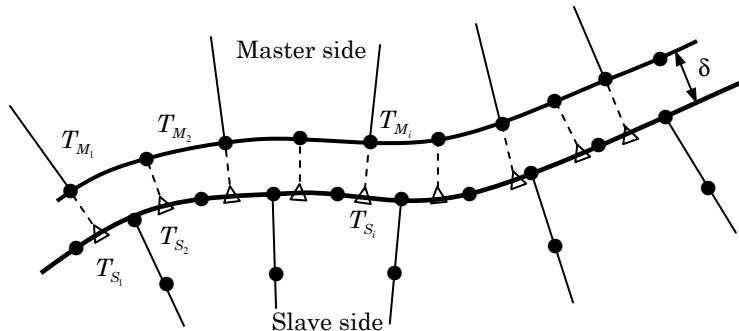


Figure 3.11.2: Sketch and nomenclature for contact boundary condition.

Two choices exist for the evaluation of T_S . When the geometric location of the master node on the slave surface is known, the slave temperature can be expressed in terms of the slave nodal point temperatures via the shape functions for the slave element. That is, $T_S = \hat{\Psi}^T(\xi_m, \eta_m)\mathbf{T}_S$ where (ξ_m, η_m) are the local surface coordinates for the location of the master node on the slave element and \mathbf{T}_S are the (unknown) nodal point temperatures on the slave surface. If this definition for T_S is used in (3.11.11b) then \mathbf{q}_{cnt} in (3.11.10b) can be written as

$$\mathbf{q}_{cnt} = \mathbf{G}\mathbf{T}_M - \mathbf{H}\mathbf{T}_S \quad (3.11.12)$$

where

$$\mathbf{H} = \int_{-1}^1 \int_{-1}^1 h_{contact} \hat{\Psi} \hat{\Psi}^T(\xi_m, \eta_m) |\mathbf{J}_s| d\xi_s d\eta_s \quad (3.11.13)$$

Since the slave term is now written in terms of nodal point temperatures, the matrix \mathbf{H} can also be added to the diffusion matrix during the solution process. This approach provides the most implicit method of formulating the contact condition and allows the direct solution of both the master and slave surface temperatures. The disadvantage of this method is that an additional nodal connectivity is required to properly distribute the coefficient matrix (\mathbf{H}) between the master and slave nodes. For static problems this connectivity can be established once and used for each iteration or time step. In a dynamic situation, where sliding along the interface or opening and closing of a gap occurs, the additional connectivity would have to be reconstructed during the course of the solution.

The slave temperature T_S may also be evaluated by direct shape function interpolation on the slave surface from the current values of the slave nodal point temperatures. In this case, the value of T_S is inserted in the boundary condition and the vector \mathbf{F}_{cnt} is constructed as shown in (3.11.11b). This method circumvents the need for additional nodal connectivity information but does introduce the need for an iterative solution for the temperature unknowns along the interface.

Central to the implementation of the contact condition is the assumed knowledge of the spatial location of the master nodes on the slave surface. Obtaining this information, especially in dynamic problems where contact surfaces may appear and disappear, can be a computationally intensive task that requires specialized algorithms. Generally, the problem specification will nominate a list of potential contact nodes and element surfaces. Various sorting methods (e.g., bin sorting, recursive bisection) can be used to group nodes and surfaces that are geometrically close and reduce the extent of the search. Ultimately each master node is tested against the possible slave surfaces to determine if the node is in contact (within some tolerance) and where on the surface (local coordinates) it is located. Testing a node for contact can be done effectively by using the isoparametric mapping for the element and writing a Newton iteration scheme to find the local coordinates of the node; the local coordinates of the master node are finally tested to determine if they coincide with the element surface.

In addition to providing a generalized surface contact boundary condition, the above formulation can provide a simple method for connecting regions with different mesh spacings. For “large” values of $h_{contact}$, Eq. (3.11.9) forces the temperature distributions on each side of the contact surface to be essentially equal. Though this

method can be made to work in practice, it is not optimal as very large values of $h_{contact}$ can cause ill-conditioning of the matrix problem and difficulties in reaching convergence with an iterative matrix solution method. The constraint boundary conditions discussed next are a better alternative for connecting dissimilar mesh regions.

3.11.3.2 Multipoint constraints

For some applications it is necessary to specify the functional relationship between the temperature at one node and temperature at one or more other nodes. The enforcing of temperature continuity between coincident surfaces with dissimilar meshes and the specification of spatially periodic temperature boundary conditions are two examples of this type of constraint. Constraints between nodes at various spatial locations could also be used in some simulation situations to represent thermal controls. The general multipoint constraint condition is similar in many respects to the contact algorithm with one surface labeled the master surface and the constrained temperature surface labeled the slave surface. The locations of the slave nodes on the master surface are found and recorded using the same types of search procedures as used in the contact algorithm.

Several different methods may be used to enforce the temperature constraint condition on the system of discrete equations. Conceptually, the simplest approach involves the field equation for each slave node being replaced with a constraint equation that relates the temperature at the slave node to some function of the nodal temperatures on the master surface. In the case where temperature continuity is enforced, the slave node value is constrained to be the interpolant of the master node values. In practice this process can be awkward. Constraints must be processed after the matrix is fully assembled and may involve considerable row and column manipulations within the global matrix to process the constraint equation or condense out the constrained nodal temperature. Lagrange multiplier methods, which will be discussed in some detail in a subsequent chapter, could also be used to apply the constraint condition. This scheme has the significant disadvantage of defining additional unknowns for the equation set. The preferred method is a penalty function approach, that begins by writing the constraint equation as

$$\mathbf{f}_{mp} = \mathbf{C}_{mp}\mathbf{T}_{mp} - \mathbf{F}_{mp} = \mathbf{0} \quad (3.11.14)$$

where the common case has $\mathbf{F}_{mp} = \mathbf{0}$, which will be considered here. The matrix \mathbf{C}_{mp} is a constant coefficient matrix with (typically) more columns than rows and \mathbf{T}_{mp} includes both slave and master nodal point temperatures. When $\mathbf{f}_{mp} = \mathbf{0}$, then the constraint is satisfied and the term $\mathbf{C}_{mp}\mathbf{T}_{mp}$ could be added to the global matrix system without altering the energy equation. In general this term is not zero; the penalty method forces this term to be small by multiplying the constraint equation by a large parameter. The penalty function form of Eq. (3.11.14) is

$$\mathbf{C}_{mp}^T \lambda \mathbf{C}_{mp} \mathbf{T}_{mp} = \mathbf{0} \quad (3.11.15)$$

The matrix λ is a diagonal matrix of penalty parameters that are selected to be large enough to force the constraint to be approximately satisfied. Premultiplying Eq. (3.11.14) by \mathbf{C}_{mp}^T makes the constraint system square and symmetric. Equation

(3.11.15) may also be derived by formally adding a penalty function $\mathbf{f}_{mp}^T \lambda \mathbf{f}_{mp}$ to the variational form of the energy equation and taking a variation of the resulting augmented potential. A further discussion of penalty methods will be delayed to the next chapter.

When temperature continuity is enforced, the slave node is set to the interpolant of the master node values. That is, for each k slave node $T_S^k = \hat{\Psi}^T(\xi_s, \eta_s)\mathbf{T}_M$, where (ξ_s, η_s) are the local surface coordinates for the location of the slave node on the master element and \mathbf{T}_M are the (unknown) nodal point temperatures on the master surface. The coefficients of \mathbf{C}_{mp} are the values of $\hat{\Psi}^T(\xi_s, \eta_s)$. Once the coefficients for \mathbf{C}_{mp} have been evaluated for each constraint, the matrix in (3.11.15) can be constructed and added to the global system. A connectivity between the slave node and the master nodes must be constructed; this data is usually generated as part of the search procedure.

For periodic conditions, the master/slave search procedure is complicated by the slave and master surfaces being in different spatial locations. Typically, a coordinate transformation is specified that translates and rotates the constrained surface into the master surface, after which the search procedure may be carried out in the usual manner. Also, the slave node temperature would generally be set to the master surface temperature plus some temperature increment representing the heat flux across the periodic geometry.

3.11.3.3 Partially covered surfaces

Contact boundary conditions permit a wide variety of material interface situations to be simulated effectively, while multipoint constraints can ease the modeling of complex geometries with little loss in solution accuracy. In both cases it is possible to have a dramatic mismatch of the element faces and edges along each side of the interface. One major effect of this mismatch is to alter the ability to consistently apply flux type boundary conditions to the exposed portions of the master or slave surfaces. Shown in Figure 3.11.3 is a typical three-dimensional case of partially covered elements for which a flux type boundary condition might be required on all exposed surfaces. The question is how to numerically integrate the boundary condition specification over the arbitrary polygonal surface of the underlying element.

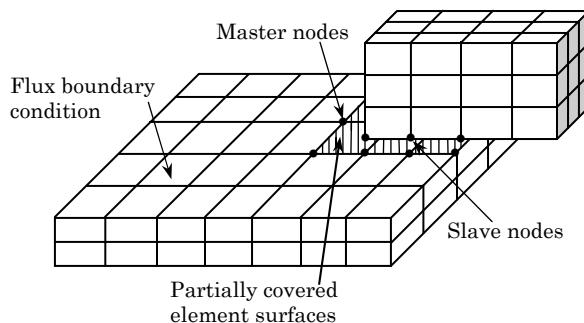


Figure 3.11.3: Sketch for partially covered element surfaces.

A method advocated by Rashid [52] begins by finding the intersection of the slave surface with the master. As a result of the intersection algorithm a series of lines is produced that describes the outline of the uncovered portion of the master surface. The surface integral that describes the flux boundary condition to the master surface can then be integrated with a special type of quadrature rule that locates evaluation points on the boundary of the exposed surface. Rashid [52] describes how the quadrature weights are derived to integrate polynomial functions to a given order of accuracy. The work involved in this algorithm is somewhat excessive for large, complex geometries. Further advances in this approach have led to the variable-element-topology finite element method (VETFEM) [53,54] where an element may contain any number of nodes in any reasonable geometric arrangement. This approach could be used in the partially covered element problem though it has yet to be explored. A simpler method would involve the conformal refinement of the polygonal surface into standard two-dimensional elements that could be easily integrated over the surface. To avoid the complexity of area refinement, an even simpler and more approximate treatment is considered here. Because the flux boundary condition is evaluated or sampled at the integration points on the master surface, it might be sufficient to determine if any integration points are covered by the slave surface and then omit them from the surface quadrature. It is a relatively easy task to determine if a master element quadrature point is within the boundary of a slave surface and flag it for exclusion from the surface integration. This can produce a very rough approximation to the surface integral, especially for cases where the slave surface covers a portion of the master surface but avoids a quadrature location. However, in many cases it provides an adequate approximation to the boundary condition and has the added advantage of being computationally simple. This last point is very important for dynamic contact problems and problems with element removal or addition.

3.11.4 Bulk Nodes

The environment external to the continuum region modeled by a finite element model influences the thermal diffusion process through the flux boundary conditions given in Eqs. (3.1.2c) and (3.6.2) and in particular through the specification of the reference temperatures, T_c and T_r . In some cases, a zero-dimensional model for the external region is useful in accounting for changes in the reference temperature. Such a model is often termed a bulk node.

A bulk node is characterized by a single temperature, $T_b(t)$, and pressure, $P_b(t)$ and is defined as a general control volume, CV, bounded by a control surface, CS. Mass and energy may flow across the control surface and the control volume may be time-dependent. Processes occurring within the bulk node are assumed to be in quasi-equilibrium and be uniformly distributed within the CV. The statement of mass conservation for the CV is

$$\begin{aligned}\frac{dM}{dt} &= \frac{d}{dt} \int_{\text{CV}} \rho dV \\ &= \sum_{\text{CS}} \delta \dot{m}_{\text{in}} - \sum_{\text{CS}} \delta \dot{m}_{\text{out}}\end{aligned}\quad (3.11.16)$$

where the summations are over the segments of the control surface where an

incremental mass flux, $\delta\dot{m}$, is defined. The energy conservation for the CV is

$$\begin{aligned}\frac{dE}{dt} &= \frac{d}{dt} \int_{CV} \rho \mathcal{E} dV \\ &= \sum_{CS} (h_0 \delta\dot{m} + \delta q + \delta\mathcal{P})_{in} - \sum_{CS} (h_0 \delta\dot{m} + \delta q + \delta\mathcal{P})_{out}\end{aligned}\quad (3.11.17)$$

where $h_0 \delta\dot{m}$ is the mass transfer energy rate, q is the thermal energy rate and \mathcal{P} is the mechanical energy rate. For the bulk node of interest here, the kinetic and potential energy changes in the mass transfer rate may be neglected. Also, shaft work and shear forces are neglected in the mechanical energy rate and conduction within the control volume is neglected in the thermal rate. The bulk node energy equation then is

$$\frac{dU}{dt} = P_b \dot{V} + \sum_{CS} (h_0 \delta\dot{m} + \delta q)_{in} - \sum_{CS} (h_0 \delta\dot{m} + \delta q)_{out}\quad (3.11.18)$$

Here P_b is the uniform pressure for the bulk node and \dot{V} is the time rate of change of the bulk node volume. Also, for an ideal substance, the internal energy is $U = MC_v T_b$ which allows the recovery of the bulk node temperature from the energy equation. For use with a finite element model, the summation over the control surface is replaced by a summation over the element surfaces bounding the bulk node. The thermal energy rates in (3.11.18) are replaced by the finite element convection and radiation boundary conditions (3.1.2c) as appropriate, where the reference temperatures are now T_b . The mass flow terms in (3.11.16) and (3.11.18) may be imposed on the bulk node to model vents or orifices associated with the geometry. These terms may also be associated with the chemical reaction of materials surrounding the bulk node. Decomposition reactions may add mass and energy to the bulk node during the reaction and/or contribute to the bulk node through the removal of elements (see Section 3.11.6). Element removal also leads to changes in the volume of the bulk node as does the Lagrangian motion of material surrounding the bulk node. The bulk node material is generally modeled as either an ideal gas or a constant pressure liquid.

The bulk node mass and energy Eqs. (3.11.16) and (3.11.18) must be solved in conjunction with the finite element equations since nodal point temperatures appear in the right-hand side of Eq. (3.11.18). Steady solutions do not usually require a bulk node since a constant state can be represented by a known reference temperature in the convective or radiative boundary condition. A completely implicit (fully coupled) formulation for the bulk node requires that Eqs. (3.11.18) and (3.11.16) be discretized in time, according to the chosen finite element integration method, and added to the finite element equations (matrix) as two additional unknowns. To establish the correct relationship between the bulk node variables and the nodal point temperatures on the surface of the bulk node volume, a special connectivity must be generated which relates one bulk node to many surface nodes.

In the standard case where the geometry surrounding the bulk node is fixed, the fully coupled method is efficient and robust. Iterative matrix solvers are particularly adept at handling the large matrix bandwidth associated with the one to many connectivity of the bulk node. Direct methods for the matrix problem may need to

be modified to effectively treat the increased bandwidth. Finally, when the geometry around the bulk node varies with time (e.g., element removal), the continued rebuilding of the connectivity and altering of the matrix structure makes the fully coupled approach less appealing. The bulk node equations may be decoupled from the finite element equations and integrated in time with any suitable integration method. Note that the forcing functions in the ordinary differential equations in (3.11.16) and (3.11.18) may often contain rough (incremental) data that is not well suited to sophisticated integration methods. Coupling of the bulk node variables with the finite element temperatures could occur at either the beginning or end of the common integration interval.

3.11.5 Reactive Materials

The presence of reactive materials in the heat conduction problem requires that a number of nonlinear conservation equations be solved for the chemical species in conjunction with the temperature field. The general formulation for the chemistry problem was outlined in Section 1.9. The mathematical nature (stiffness) of the kinetic equations dictates that for computational efficiency, the chemistry and thermal diffusion equations be solved independently. The solution process is formally based on an operator splitting technique as defined, for example, in [55].

Operator splitting is particularly effective in the case of condensed materials due to the form of the kinetic equations. Because diffusion of the species is neglected, the kinetic equations have no spatial gradients and reduce to ordinary differential equations that can be defined locally on each finite element. In essence, the chemical species can be viewed as state variables for each element and can be solved on an element-by-element basis. In most situations it is convenient to define all species and species equations at the integration points for each element. During a time step, the chemistry solution is advanced first using a fixed (frozen) temperature field; the temperature field is subsequently advanced over the same time interval using the recently evaluated (frozen) chemistry result. If a predictor/corrector time integration method is employed, the frozen temperature field used for the chemistry solution is the temperature produced from the predictor step. When a predictor equation is not employed for time integration, the last available temperature field is used for the chemistry solution.

The inherent stiffness of the kinetic equations requires that special integration methods be used to advance the chemistry solution in time. Stiff, ordinary differential equation (ODE) methods have been extensively studied and a number of library packages are available. The package CHEMEQ [56] developed by T. R. Young was designed specifically for chemical reaction systems and is easily incorporated into a finite element solution method. An updated version of the kinetics package, CHEMEQ2 [57], may also be used in finite element codes. The techniques used in CHEMEQ are based on a combination of classical predictor/corrector methods and asymptotic methods for the stiff components of the system. The rate equations for each reactive finite element may be solved using their own integration time step over the global time interval of interest. The most restrictive chemistry time step for all of the reactive elements may then be used to influence the choice of the next thermal diffusion time step. A typical relation for thermal time step choice is [see Eq. (3.7.20)]

$$\Delta t_{n+1} = \min\{\Delta t_{\text{diff}}, X_{\text{chem}} \times \Delta t_{\text{chem}}\} \quad (3.11.19)$$

where Δt_{diff} is the estimated time step for the heat conduction equation and Δt_{chem} is the minimum time step estimated for the chemistry solution. The parameter X_{chem} is a user-defined scale factor that typically has a value between 10 and 100. When reactive processes are unimportant, the adaptive time integration in the chemistry integrator will produce a chemistry time step that is relatively large and Eq. (3.11.19) will allow the conduction solution to dictate the problem time scale. As the reactive process accelerates, the chemistry time step will decrease significantly and ultimately control the time step formula in (3.11.19). The transition point for control of the global time step can be dictated by the user through the X_{chem} parameter.

As noted in Section 1.11, a change of phase may be viewed as a reversible chemical reaction with material 1 going to material 2 with an appropriate change in energy. Writing the phase change reaction as two one-way reactions with reaction rates k_1 and k_2 produces



where \mathcal{M}_1 is assumed to be the low temperature phase. Using the chemical reaction formulation from Section 1.9, the ordinary differential equation (1.9.12) describing the above reaction are

$$\frac{d\mathcal{M}_1}{dt} = R_1 = k_1\mathcal{M}_1 - k_2\mathcal{M}_2 = A_1 e^{(-E_1/RT)} \mathcal{M}_1 - A_2 e^{(-E_2/RT)} \mathcal{M}_2 \quad (3.11.21a)$$

$$\frac{d\mathcal{M}_2}{dt} = R_2 = -k_1\mathcal{M}_1 + k_2\mathcal{M}_2 = -A_1 e^{(-E_1/RT)} \mathcal{M}_1 + A_2 e^{(-E_2/RT)} \mathcal{M}_2 \quad (3.11.21b)$$

where β_1 and β_2 are assumed to be zero. Employing the substitution $\mathcal{M}_2 = 1 - \mathcal{M}_1$ in (3.11.21a), this equation may be solved analytically for $\mathcal{M}_1(T, t)$ yielding

$$\mathcal{M}_1 = \frac{k_2}{k_1 + k_2} + \frac{k_1}{k_1 + k_2} e^{-(k_1 + k_2)t} \quad (3.11.22)$$

where it has been assumed that \mathcal{M}_1 and \mathcal{M}_2 range between 0 and 1 with initial conditions of $\mathcal{M}_1 = 1$ and $\mathcal{M}_2 = 0$. This is a linear first-order reaction. The constants in Eq. (3.11.22) are the pre-exponentials, A_1, A_2 and the activation energies, E_1, E_2 , which may be adjusted to set the temperature and temperature interval over which the reaction or phase change occurs. The latent heat release is described by Eq. (1.9.9) and for the pair of reactions is

$$Q_r = q_1 r_1 + q_2 r_2 = q_1 k_1 \mathcal{M}_1 + q_2 k_2 \mathcal{M}_2 \quad (3.11.23)$$

The equations in (3.11.21) can be solved by the methods outlined above for the general reaction kinetics problem. When coupled with the heat release equation in (3.11.23) and the heat conduction equations for the material, an efficient method for phase change problems is produced.

3.11.6 Material Motion

The heat conduction equation in (3.1.1) was written for a stationary region, in which case the Lagrangian and Eulerian descriptions of the boundary value problem are identical (see Section 1.4.1). If motion of the conducting regions is to be considered, then a particular reference frame must be selected. The Eulerian description is usually associated with fluid-like motions that involve very large material deformations, unsteadiness, and a need to fully consider the equations of motion for the material. However, there are heat conduction problems that can be well described in an Eulerian reference frame. Generally, if the motion is continuous (steady), the kinematics are simple and essentially unidirectional and the geometry of the moving material is not complicated, then an Eulerian description may be appropriate. Problems of this type are usually related to some type of continuous processing operation on a simply shaped (cylinder, bar, slab, etc.) material region. The Eulerian form of the energy equation is altered from (3.1.1) to the advection-diffusion form of (1.4.13) or (1.5.3). That is,

$$\rho C \left(\frac{\partial T}{\partial t} + u_j \frac{\partial T}{\partial x_j} \right) = \frac{\partial}{\partial x_i} \left(k_{ij} \frac{\partial T}{\partial x_j} \right) + Q \quad (3.11.24)$$

Since a momentum equation is not considered in the heat conduction problem, it is required that the velocity field in (3.11.24) be completely prescribed as a function of time and space; the nonconservative form of the energy equation also requires that the velocity field be divergence free. The additional advective term present in (3.11.24) adds minimal complexity to the heat conduction formulation and can be easily incorporated in a finite element formulation as an alternate energy equation. The finite element equations related to (3.11.24) will not be discussed here since they form a major element of the next two chapters on fluid mechanics and convective heat transfer. Note that Eq. (3.11.24) may be used with the standard conduction equation in different regions of the same problem since they are both referenced to the same coordinate system. Mixtures of Eulerian and Lagrangian descriptions are also permissible.

Heat conduction problems are usually described in a Lagrangian frame since the focus is on solid bodies. The incorporation of material motion in the Lagrangian description is easily accomplished as long as the material deformations are not too large; distortion of the individual finite elements may become a problem under large deformations. Again, since no equations of motion are included in the conduction problem, the kinematics for the material region must be completely specified. For rigid body motions such a prescription is straightforward; new material coordinates are directly defined by translation and rotation of the region. Material deformation is generally more complex and usually requires the solution of a solid mechanics problem for all but the simplest motions. Deformation is normally accompanied by changes in density which must be accounted for in the conduction problem. Mass conservation issues are not usually associated with heat conduction problems but must be considered when deformations are present.

Related to material motion is the situation where material is added to or removed from the original heat conduction region. This type of physical situation can be represented by the simple addition or deletion of finite elements from the original

element discretization of the problem. The physical process will be well modeled if the time constant for the addition or deletion of material is short compared to the time constant for the thermal process and the details of the material alteration are unimportant. Element birth and death may be implemented by either of two methods. In the first case, elements are added to or deleted from the global element list, new connectivities are established and entries in the global matrix are reorganized to reflect the new list of elements. This scheme has the benefit of always solving a problem of minimal size, though it suffers from the additional work needed to reorganize the element data. The second approach always builds an equation set (matrix) that corresponds to all the elements that could possibly be in the problem and uses a fixed connectivity. If an element is not currently active (i.e., has not been born or has already died) then the equations for that element are zeroed out and replaced by a constant temperature constraint. The matrix problem for this case is larger than the first scheme, but there is no work to activate or deactivate an element other than the setting of an element flag.

A common problem in element birth and death is the updating of boundary conditions with changes in the element topology. If a three-dimensional element with a flux boundary condition is removed from the problem, some type of algorithm must be present to determine which newly exposed faces of the surrounding elements now have a boundary condition applied to them. One method for treating this problem of boundary condition inheritance is to tag every face of every element with information about its adjacent element face. Upon activating or deactivating an element, the face data of the adjacent elements is switched to reflect the new state of the face (either exposed or covered). For this type of method to be effective, boundary conditions must be applied to the entire region of a problem. This algorithm must also be altered, if contact or multipoint constraint conditions exist and partially covered element faces occur.

3.12 Example Problems

3.12.1 Introduction

In this section a number of heat conduction and radiation problems are presented to illustrate the various aspects of finite element analysis procedures. The first example is a simple two-dimensional/three-dimensional problem designed to illustrate convergence and accuracy of various basic finite elements. The second problem demonstrates the performance of the conduction/radiation solution algorithms. A second group of examples demonstrates the use of anisotropic and nonlinear material properties as well as the phase change algorithm. The remaining examples are more complex and represent typical engineering problems encountered in practice. All of the problems described here were analyzed using the finite element code, COYOTE [58].

3.12.2 Element Convergence

The spatial accuracy and order of convergence of various finite elements can be evaluated by any of several methods and is usually performed as part of a code verification process [59,60]. In this example, an analytic solution is available for comparison with a series of numerical solutions produced on various mesh

refinements. The physical problem involves a two-dimensional section of a 2:1 aspect ratio bar. The section is volumetrically heated with the vertical faces having an applied flux (left face) and a specified temperature (right face). The horizontal faces are adiabatic (bottom face) and cooled with a convective heat transfer coefficient (top face). The steady solution was developed analytically in terms of Green's functions [61,62] but requires the numerical evaluation of a series. This particular solution was included as part of a general solution procedure for parallelepiped developed by Beck [63] and used for code verification purposes.

The test problem has a horizontal length of 0.1 m and vertical height of 0.05 m. The problem can be extruded in the third dimension to test three-dimensional elements. The thermal conductivity was set to $k = 0.4\text{W/m}\cdot^\circ\text{C}$ and the volume heating was constant at $Q = 1.353 \times 10^5\text{W/m}^3$. For the boundary conditions, the applied flux is $q = 3,500\text{W/m}^2$, the specified temperature is $T = 25^\circ\text{C}$ and the convective parameters are $h = 60\text{W/m}^2\cdot^\circ\text{C}$ and $T_c = 25^\circ\text{C}$. A series of mesh refinements ranging from 5×5 to 320×320 were used with both linear and quadratic quadrilateral elements; similar meshes with three-dimensional hexahedral elements produced the same results.

The steady temperature field for the above test problem is shown in Figure 3.12.1 as an isotherm plot. Table 3.1 lists the computed temperatures and heat fluxes at three points in the domain for each of the mesh refinements and each element type. The points used for reporting temperatures and components of the heat flux are: Point 1-lower left corner, Point 2-midpoint of top surface and Point 3-upper left corner. The analytic solution is also listed and it can be seen that for even modest refinements and the linear element type, the temperature predictions are quite accurate. The convergence of the flux is substantially slower, especially for the lower-order element. Local, point-wise convergence rates (slope of the error versus refinement curve) in the L_{\inf} norm for the temperature are listed in the table and are seen to follow the second-order rate predicted by interpolation theory for the linear element; the quadratic element is superconvergent in temperature at Points 1 and 2. The flux singularity at Point 3 compromises the accuracy at that point.

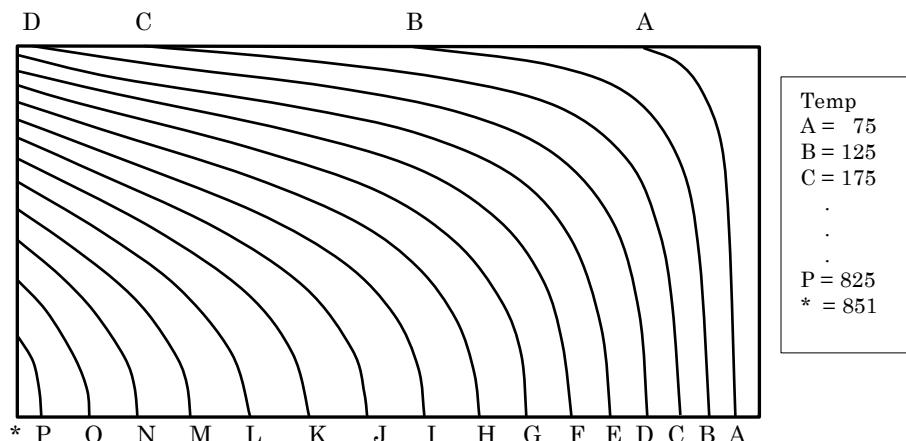


Figure 3.12.1: Temperature contours for 2:1 bar verification problem.

Table 3.12.1: Finite element convergence to an analytical solution.

Mesh $n \times n$	Point 1		Point 2		Point 3
	Temp	Flux, q_x	Temp	Flux, q_y	Temp
Quad 4 elements					
5 × 5	855.375	3,121.24	—	—	226.550
10 × 10	854.342	3,300.73	128.220	5,863.18	231.069
20 × 20	854.087	3,398.88	128.212	6,027.17	233.175
40 × 40	854.023	3,449.25	128.210	6,109.74	234.000
80 × 80	854.007	3,474.60	128.210	6,151.13	234.289
160 × 160	854.003	3,487.30	128.210	6,171.86	234.382
320 × 320	854.002	3,493.65	128.210	6,182.22	234.411
Conv. Rate	2.016	1.00	2.229	1.00	1.574
Quad 8 elements					
5 × 5	853.958	3,483.80	128.209	6,186.16	233.380
10 × 10	853.999	3,497.81	128.210	6,194.84	234.126
20 × 20	854.002	3,499.72	128.210	6,193.19	234.343
40 × 40	854.002	3,499.96	128.210	6,192.74	234.402
80 × 80	854.002	3,499.99	128.210	6,192.63	234.417
160 × 160	854.002	3,500.00	128.210	6,192.61	234.421
320 × 320	854.002	3,500.00	128.210	6,192.60	234.422
Conv. Rate	4.327	2.99	4.403	1.92	1.915
Exact	854.002	3,500.00	128.210	6,192.60	234.423

3.12.3 Conduction/Radiation Solution

A useful geometry for the testing of coupled conduction and radiation problems consists of a series of circular concentric conducting rings separated by annular radiation enclosures. Shown in Figure 3.12.2 is a five ring example of this geometry. In this case the five rings have uniform thickness and are composed of the same material, a mild steel. The enclosures are uniformly spaced. The inside of the inner ring and the outside of the outer ring are subjected to convection boundary conditions. The higher reference temperature for the inner ring produces an overall energy transfer across the geometry to the lower external temperature. This problem is easily modified to make the coupled problem easier or more difficult by changing the properties of one or more of the rings, changing the enclosure gaps and/or modifying the boundary conditions. This problem has been studied for both transient and steady solution methods by Hogan and Gartling [38].

To illustrate the differing performance of some of the coupled solution methods defined in Section 3.8 only a single, steady state example will be considered here. The inner radii of the rings are 0.01 to 0.05 m with a thickness of 0.002 m. The thermal conductivity is $k = 35.0\text{W/m}\cdot\text{^\circ K}$ and the surface emissivity of all surfaces is 0.8. The convection conditions on the inner ring are $h = 100\text{W/m}^2\cdot\text{^\circ K}$ and $T_c = 3,000\text{^\circ K}$ while the outer ring is subjected to $h = 50\text{W/m}^2\cdot\text{^\circ K}$ and $T_c = 300\text{^\circ K}$. For each steady state solution method all the rings were initialized to a uniform temperature of $T_i = 1,600\text{^\circ K}$. Convergence was accepted at a tolerance of 1.0×10^{-5} on the L_2 -norm of the successive iterations.

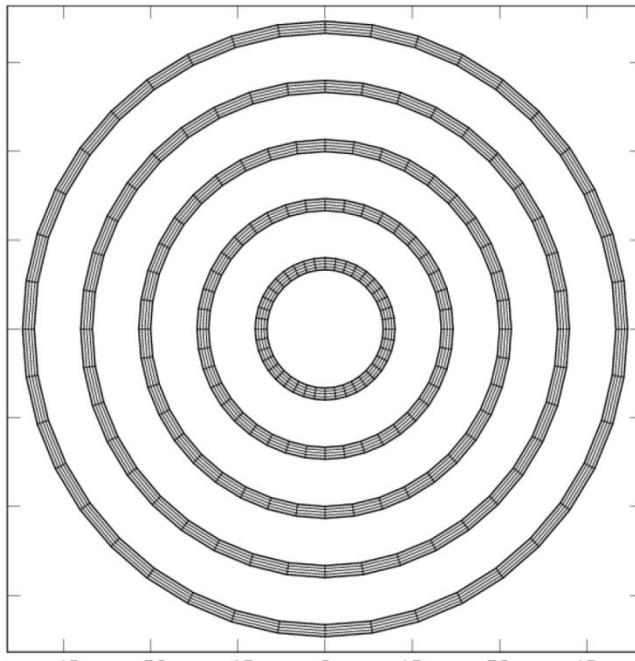


Figure 3.12.2: Schematic and mesh for conduction/radiation test problem.

The solution methods used for this problem included the fully coupled (Newton) method (FC), the segregated explicit method with Picard (SEP) and the segregated semi-implicit Newton method (SSN). Some performance data for the three methods is presented in Table 3.12.2. The fully coupled method is clearly superior, though it does have the major drawback of increased matrix size. The semi-implicit scheme is quite good, especially considering its applicability to very large problems. The explicit method is really not robust enough for general problems. An analytic solution can be derived for this problem, and for the typical (linear element) discretization shown in Figure 3.12.2 and the stated convergence tolerance, the temperature errors were less than 0.10%.

Table 3.12.2: Performance of coupled conduction/radiation solution algorithms.

Algorithm	Initial Temp	Relaxation Factor	Number of Iterations	CPU Time (sec)
FC	Uniform 800	0.0	7	1.35
FC	Uniform 1,600	0.0	6	1.08
SEP	Uniform 800	0.0	Divergence	—
SEP	Uniform 1,600	Varied	Divergence	—
SSN	Uniform 800	0.0	Divergence	—
SSN	Uniform 800	0.2	73	2.24
SSN	Uniform 1,600	0.0	133	3.47

3.12.4 Temperature-Dependent Conductivity

Many materials of engineering interest have thermophysical properties, such as conductivity, that vary with temperature. This example illustrates the difference in results that can be expected when conductivity variations are included in the model. The domain, material property variations, boundary conditions, and mesh of four-node bilinear elements are shown in Figure 3.12.3. Each half of the domain is occupied by a different isotropic material; Material 1 has constant conductivity, $k_1 = k_0$, while Material 2 has a conductivity that varies linearly with temperature as $k_2 = k_0 + aT$. No heat flow across the boundary between the two materials is assumed. All other surfaces are convectively cooled with a constant h_c and T_c .

Due to the variable conductivity, this problem is nonlinear and the steady-state solution was obtained using a Picard iteration. A converged solution was obtained in four iterations. Figure 3.12.4 contains a plot of the isotherms. Since the plotted isotherm bands are the same for the two materials, the difference in the temperature fields is readily observed. At the high temperature end (applied flux surface) of the region, the conductivity for the variable material is high and the temperature gradient (isotherm spacing) is therefore lower than for the constant conductivity case. At the low temperature end of the region where the conductivities are comparable, the temperature gradients are similar.

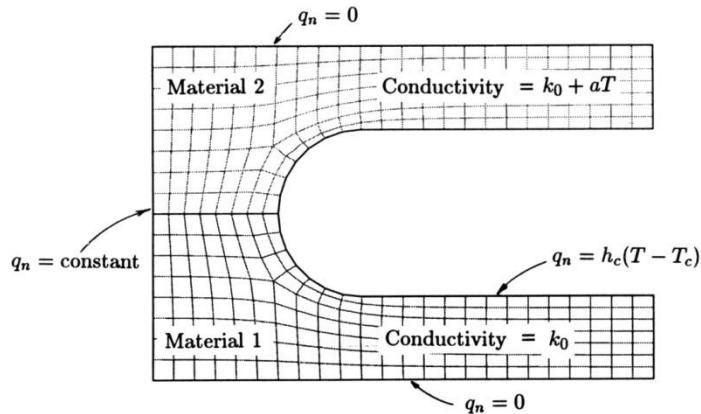


Figure 3.12.3: Schematic and mesh for the temperature-dependent conductivity problem.

3.12.5 Anisotropic Conductivity

As explained in Section 3.9, the inclusion of an anisotropic conductivity into a finite element model is relatively straightforward and may be important for the simulation of laminated, fibrous, or wire-wrapped materials. A planar square with a tilted square insert is used to demonstrate the effects of an orthotropic conductivity. Referring to Figure 3.12.5, the outer square is an isotropic material with conductivity, k . Constant temperature boundary conditions are imposed on the vertical edges of the square while the horizontal edges are insulated. The insert material is orthotropic with $k_{11} = k$ and $k_{22} = k/10$; the orientation of the material axes with respect to the global coordinate axes will be specified for each simulation.

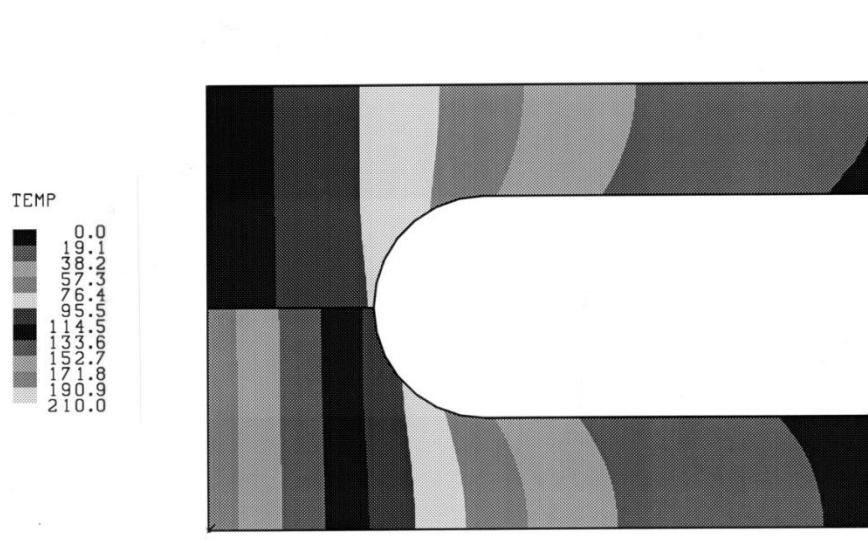


Figure 3.12.4: Isotherms for the temperature-dependent conductivity problem.

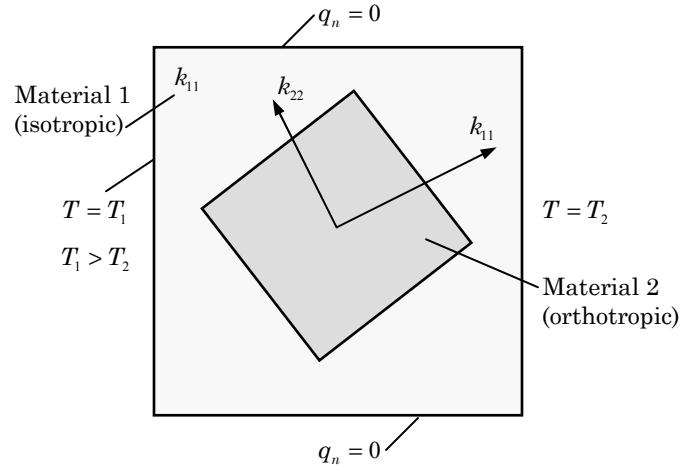


Figure 3.12.5: Geometry of the anisotropic conductivity problem.

Since the problem is linear, it may be solved directly without iteration. The first solution (see Figure 3.12.6a) corresponds to the case where the material axes for the insert material are aligned with the coordinate axes. As expected, the solution is one-dimensional since the k_{11} component for the insert is the same magnitude as the surrounding isotropic material; the k_{22} component plays no role in the solution since there is no temperature gradient imposed in the transverse direction. Figure 3.12.6b shows the temperature field for the case where the material axes of the insert are inclined upward 45° with respect to the global axes (the material axes are aligned with the mesh lines in the insert). The distortion of the temperature field due to the anisotropy is very obvious. Finally, Figure 3.12.6c illustrates the solution for the case as in (b) but with both components of the insert conductivity reduced by a factor of 10. This accentuates the difference between the two materials and leads to higher gradients and more distortion of the isotherms within the insert.

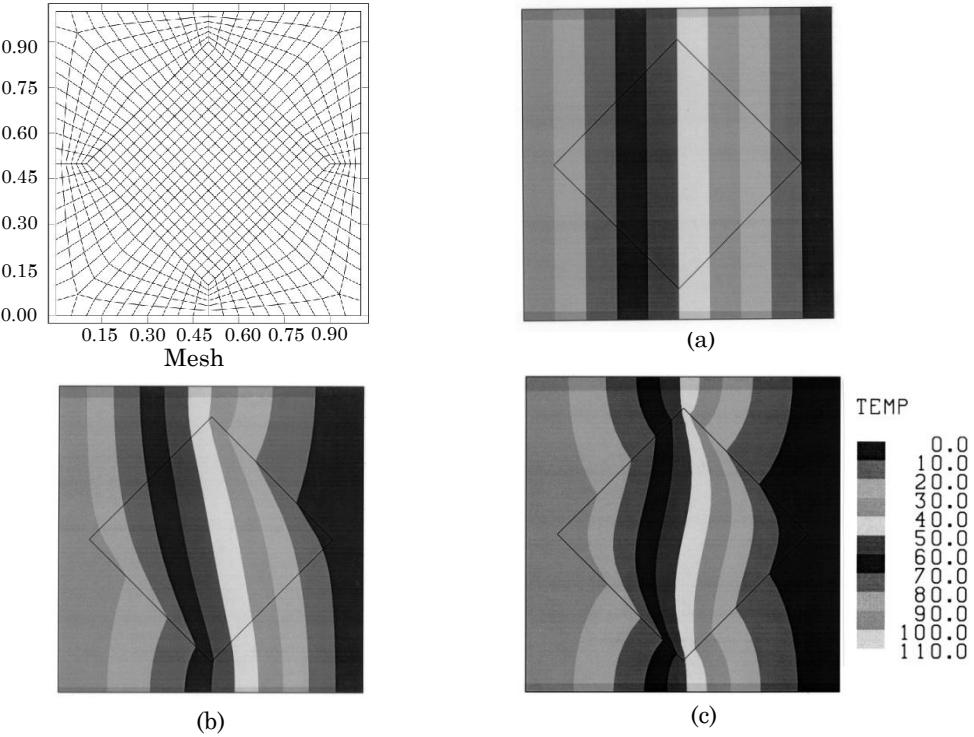


Figure 3.12.6: Temperature fields for the anisotropic conductivity problem.

3.12.6 One-Dimensional Stefan Problem

A standard test problem for phase change algorithms is the one-dimensional Stefan problem (see Carslaw and Jaeger [64]). The problem consists of a material region $0 \leq x \leq 4$, held initially at a uniform temperature greater than the liquidus temperature. At time zero, one face of the region ($x = 0$) is lowered to a temperature below the solidus temperature, causing a solidification front to propagate into the material. The present example is solved using a series of meshes with eight-node (biquadratic) elements and the trapezoid rule, time integration procedure. The schematic of the one-dimensional problem is shown in Figure 3.12.7, which also contains a finite element mesh and boundary conditions; mesh densities from 16 to 256 elements were used in the simulations. The following material properties are used: $\rho = 1.0$, $C = 1.0$, $k = 1.08$, $L = 70.26$, $\Delta T = 2.0, 0.5$, $T_s = -0.15$, and $T_l = -2.15, -0.65$.

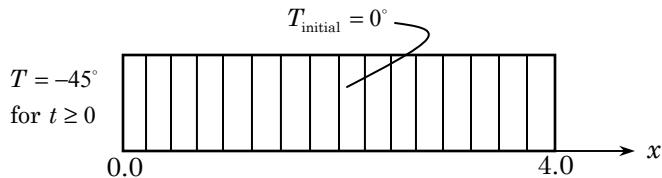


Figure 3.12.7: Schematic of the one-dimensional Stefan problem.

Figures 3.12.8a and 3.12.8b show the computed temperature fields at a point one unit from the cooled face. Computations were made for two values of the phase change temperature interval, ΔT , and a range of integration time steps. Note that the analytical solution was developed for a single phase transition temperature while the numerical solution requires a finite transition interval, ΔT . Figure 3.12.8a shows the temperature response under mesh refinement for a $\Delta T = 2.0$ case and a time step of $\Delta t = 0.01$; the computed response for $\Delta T = 0.5$ is essentially the same with the indicated time step and the same mesh refinements. Figure 3.12.8b shows the temperature response when the time step is decreased and the latent heat release is more accurately represented. These simulations are for a fixed mesh refinement of 256 elements. Note that the size of the phase transition interval, ΔT , is relatively unimportant in this case with the time step having the predominate effect. Figure 3.12.9 contains a plot of the location of the phase boundary as a function of time for the last cases studied on a fixed 256 element mesh. The larger time step case overpredicts the location of the phase boundary early in time and retains the error throughout the simulation. In the smaller time step simulations an effect of a larger ΔT can be seen as the solution progresses. The use of $\Delta T = 0.5$ and a small time step provides a reasonably accurate solution to the phase change problem and a good approximation to the analytic solution.

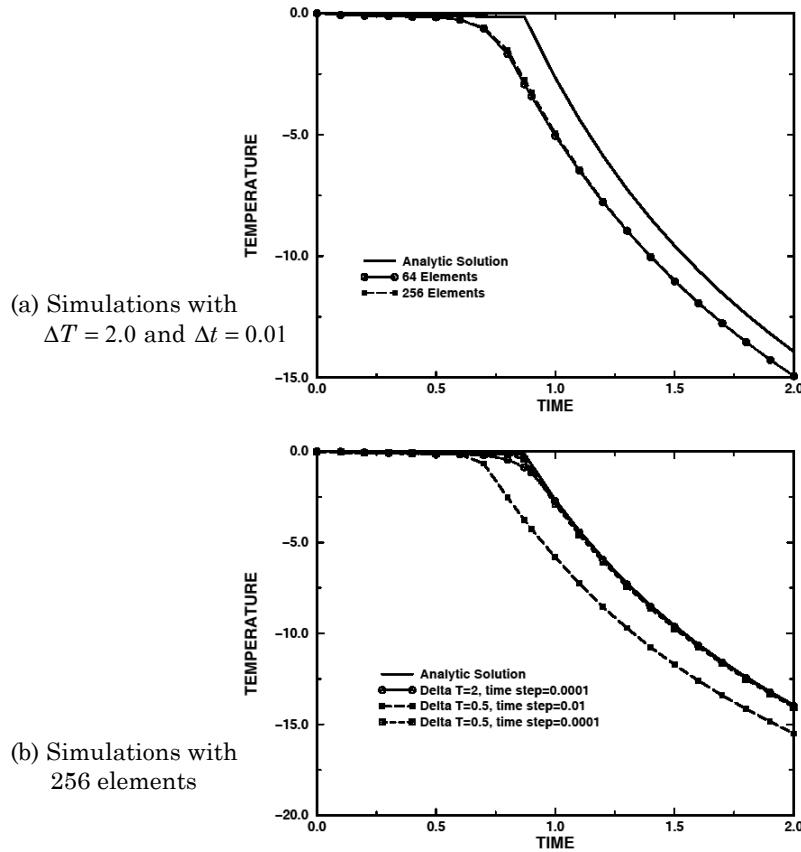


Figure 3.12.8: Temperature vs. time at $x = 1.0$ for the Stefan problem.

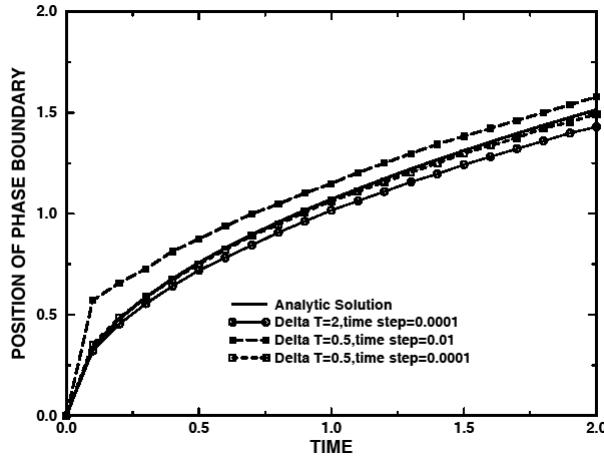


Figure 3.12.9: Location of the phase boundary vs. time for the Stefan problem. Corresponds to simulations with a 256 element mesh.

3.12.7 Drag Bit Analysis

The next example deals with the thermal (and stress) analysis of a polycrystalline diamond compact (PDC) drag bit that is commonly used in the oil and gas drilling industry. A planar, two-dimensional model of a single stud in the drag bit body is considered (see Figure 3.12.10). The thermal boundary conditions for the model are also indicated in the sketch. Heat is input to the stud at the wearflat due to frictional heating; the heating function is given as an empirical function of cutter, rock, and operating conditions. The remaining surfaces of the stud are convectively cooled by the drilling fluid. Boundary conditions on the bit body are taken to be adiabatic since they represent planes of symmetry with adjoining cutters. Figure 3.12.11 contains two finite element meshes of eight-node quadrilateral elements used for this problem; the second mesh shows the location of a cooling nozzle and a convection boundary condition used in some of the parametric studies.

A wide variety of boundary and geometric conditions in both the steady-state and transient regimes [65] have been examined for the problem. Figures 3.12.12a and 3.12.12b contain plots of isotherms for two steady-state cases, which differ in the assumed convective heat transfer coefficient for the drilling fluid. The high temperature gradients caused by the high cooling rate are clearly seen in the figures. As is quite common in many applications, a finite element stress analysis followed the heat transfer study. Figures 3.12.13a and 3.12.13b show contour plots of the predicted plastic strains in the cutter corresponding to the temperature fields of Figures 3.12.12a and 3.12.12b, respectively, and an appropriate set of mechanical loads. Studies such as this have been used to predict the life of drill bits, improve cutter design, and provide operational limits for improved drilling performance.

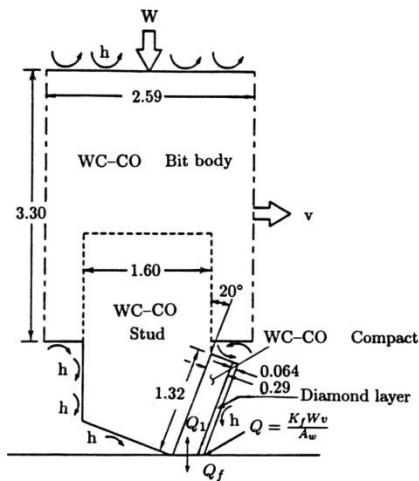


Figure 3.12.10: Schematic of the polycrystalline diamond compact (PDC) drag bit.

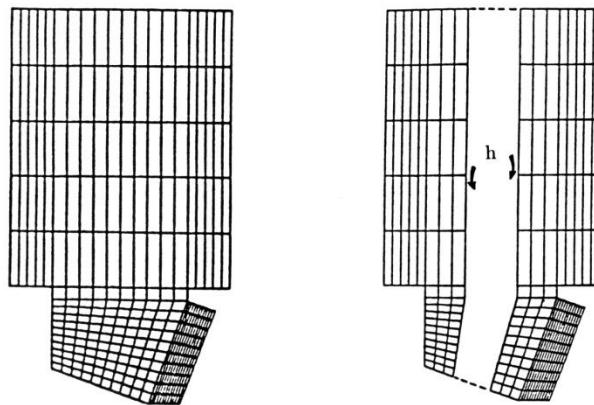


Figure 3.12.11: Finite element meshes used for the analysis of the PDC drag bit.

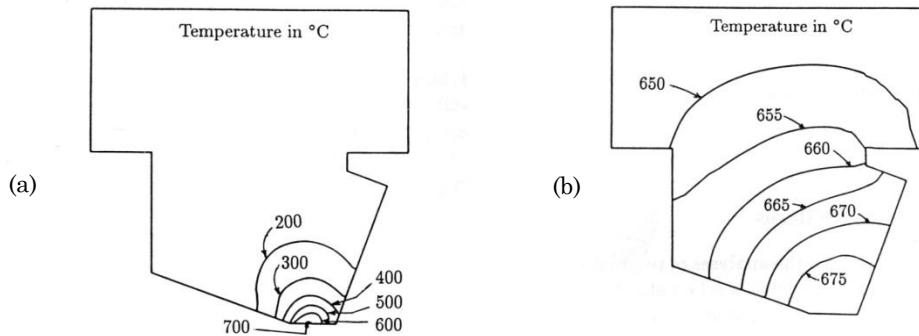


Figure 3.12.12: Steady-state isotherms for the drag bit. (a) $h_c = 5 \text{ W}/(\text{cm}^2 \cdot {}^\circ\text{C})$. (b) $h_c = 0.01 \text{ W}/(\text{cm}^2 \cdot {}^\circ\text{C})$.

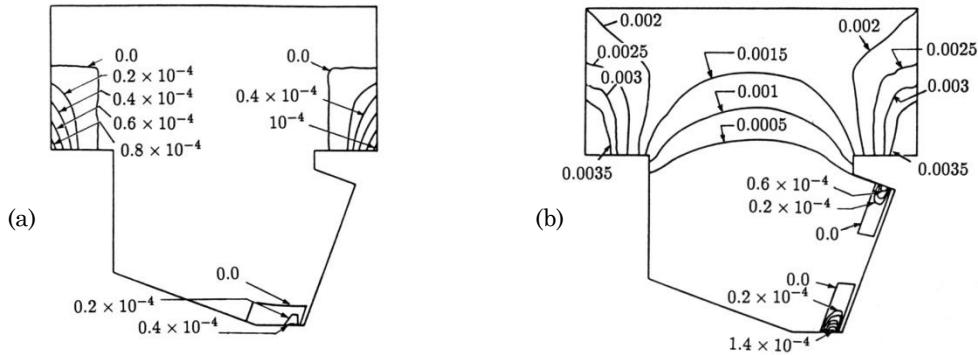


Figure 3.12.13: Plastic strain contours for the drag bit. (a) Corresponds to thermal field in Figure 3.12.12a. (b) Corresponds to thermal field in Figure 3.12.12b.

3.12.8 Brazing and Welding Analysis

The next two examples come from the analysis of two manufacturing processes for electrical components. Figure 3.12.14 shows a schematic and finite element mesh for one-half (upper half) of the axisymmetric geometry of interest; a three-dimensional mesh used in a subsequent analysis to verify the two-dimensional results is also shown. In manufacturing this component, two halves (upper and lower) are brought

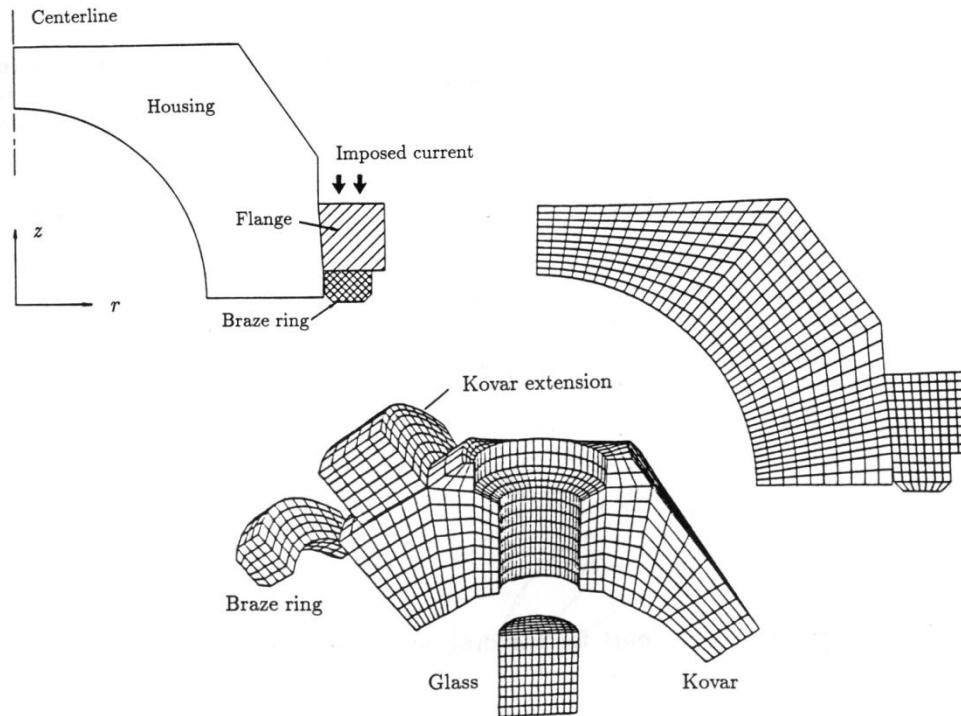


Figure 3.12.14: Schematic and finite element mesh for the brazing problem.

together under a modest pressure. An electrical current is passed through the outer flange to resistively heat the braze rings and bond the two halves of the device. In the simplified thermal model considered here, the brazing process is modeled using a time-dependent (pulsed) volumetric heating of the flange and braze ring. All exterior boundaries are assumed to be adiabatic since the entire process takes place on a very short time scale. This problem was solved using four-node, bilinear elements and a modified Crank–Nicolson procedure.

Figure 3.12.15 shows a series of isotherm plots for two times during a one-pulse brazing cycle. Heating occurs over the first 0.5 seconds, during which time the temperature on the flat of the braze ring reaches $\sim 2100^{\circ}\text{F}$. Later in time, the temperature in the braze ring decreases as energy is conducted toward the centerline of the component. A companion stress analysis was used to identify high stress regions and potential failure (cracking) sites in the component.

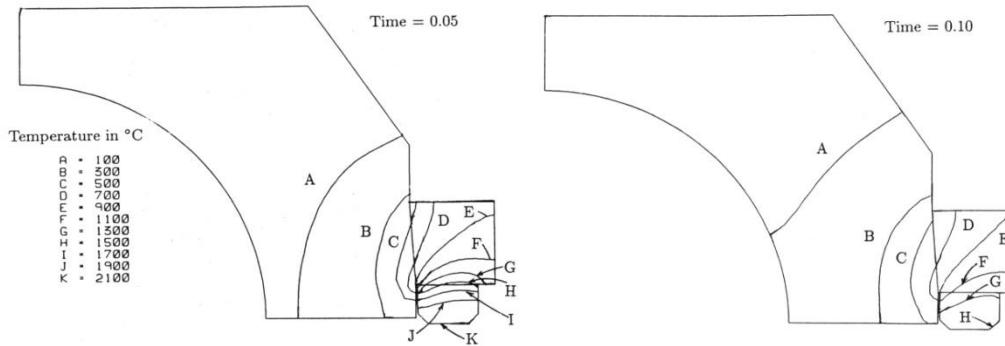


Figure 3.12.15: Isotherms at two times during the brazing analysis.

Like brazing, welding is a process that is often used in the assembly of electrical components. Figure 3.12.16 shows the finite element representation of a battery case and attached header; the header and case are welded using a gas-tungsten arc (GTA) weld. The metallic connecting pins in the battery header are fixed to the header with glass-to-metal seals and must retain a hermetic seal during the GTA process. The figure shows an enlargement of the connecting pins, seals and header assembly. A study was performed to evaluate the benefits of providing various types of heat sinks to the header and case during the weld process. Details of the parametric study and weld parameters and modeling are available in [66]. Figure 3.12.17 contains one heat sink configuration; the mesh for this problem contained approximately 27,000 eight-node, brick elements.

Figures 3.12.18 and 3.12.19 show temperature fields in the header at the conclusion of one revolution of the GTA welder. It is clear from the large differences in temperature at the pins that heat sinking has a strong positive benefit. The temperature gradients in the glass for the no heat sink case are sufficient to cause a tension failure (cracking) of the glass.

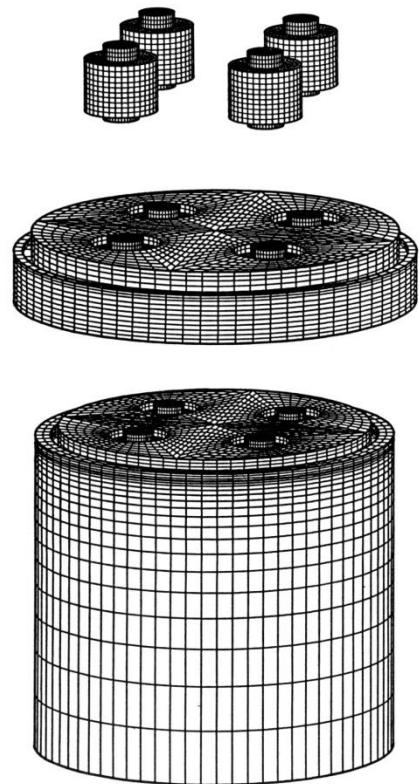


Figure 3.12.16: Finite element mesh for the battery header welding problem.

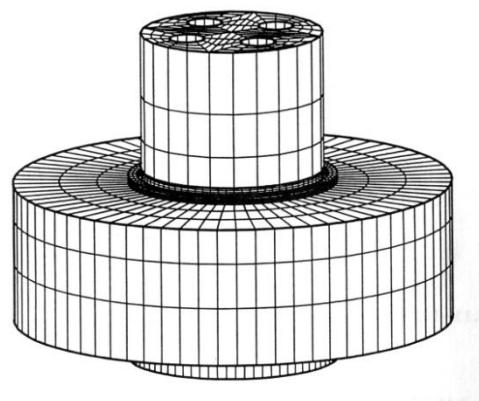


Figure 3.12.17: Finite element mesh for the heat sink used in the battery header weld.

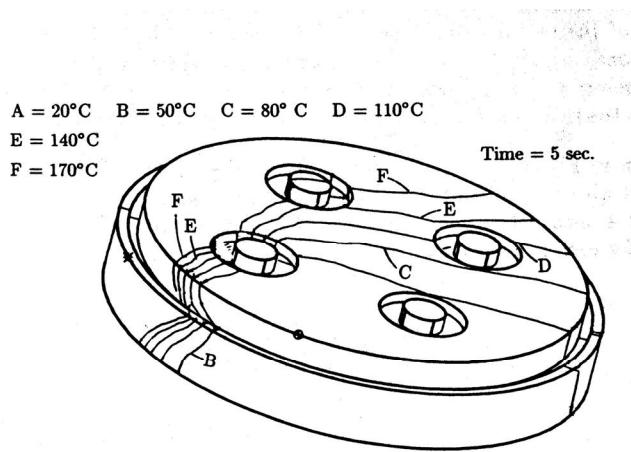


Figure 3.12.18: Temperature field in the header following weld for the case without heat sink (Time, $t = 5$ sec).

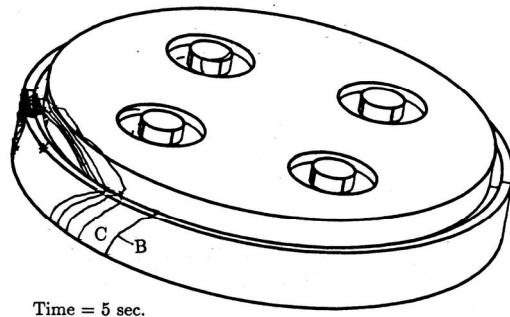


Figure 3.12.19: Temperature field in the header following weld for the case with heat sink (Time, $t = 5$ sec).

3.12.9 Investment Casting

A heat transfer analysis of the cooling phase of an investment casting can provide important guidance to the design of the metal distribution system (gates, runners, etc.) for the casting and indicate potential casting problems (e.g., shrinkage). Shown in Figure 3.12.20 is a finite element mesh of a structural part to be produced in stainless steel by the investment casting process. The mesh contains ~ 2400 eight-node brick elements. For purposes of this preliminary analysis, the mold for the casting is not considered and the metal distribution system is also ignored. The casting model is initially assumed to be at the metal pour temperature and loses energy to the furnace walls through radiation; investment castings are poured in a vacuum. The number of element surfaces participating in radiative transfer is ~ 4500 , which leads to approximately 20 million view factors. The view factors were computed using both the double area integration and the hemicube method. A further review of some of the numerical and modeling problems encountered in this type of simulation is available in [67].

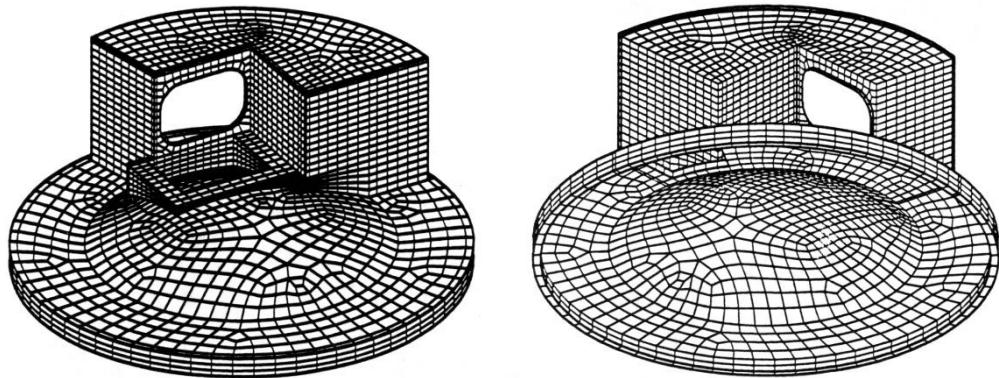


Figure 3.12.20: Finite element mesh for a cast structural part.

Shown in Figure 3.12.21 are surface temperature plots at two times during the cooling process. The temperatures in Figure 3.12.21 were computed using the full radiation model; a simplified radiation boundary condition that did not use view factors but employed a uniform σT^4 heat loss on the surface produced a significantly different temperature field.

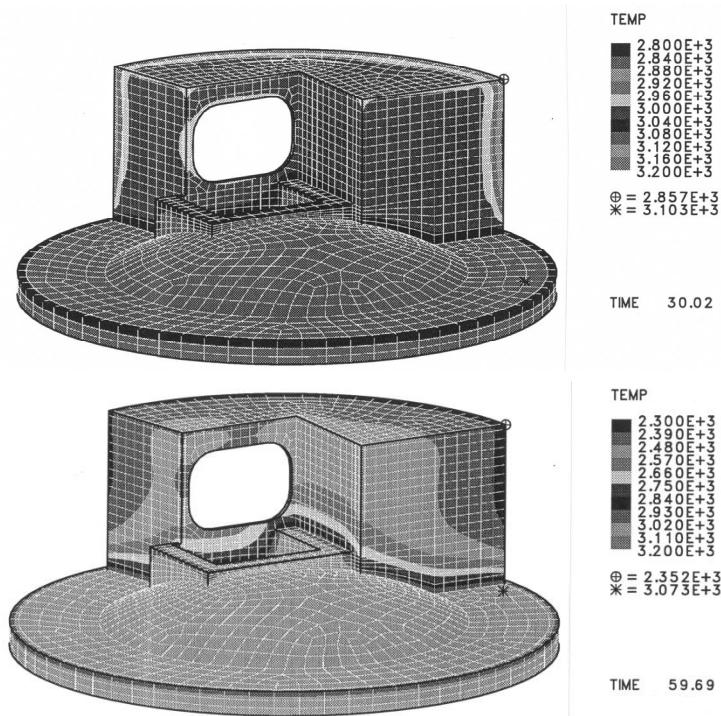


Figure 3.12.21: Temperature field (in $^{\circ}\text{F}$) during cooling of a casting using an enclosure radiation model.

Figure 3.12.22 contains plots showing the extent of the mushy zone (liquidus to solidus temperature interval) at two times during cooling. Such information, coupled with cooling rate data, can provide an indication of the metal microstructure and part quality.

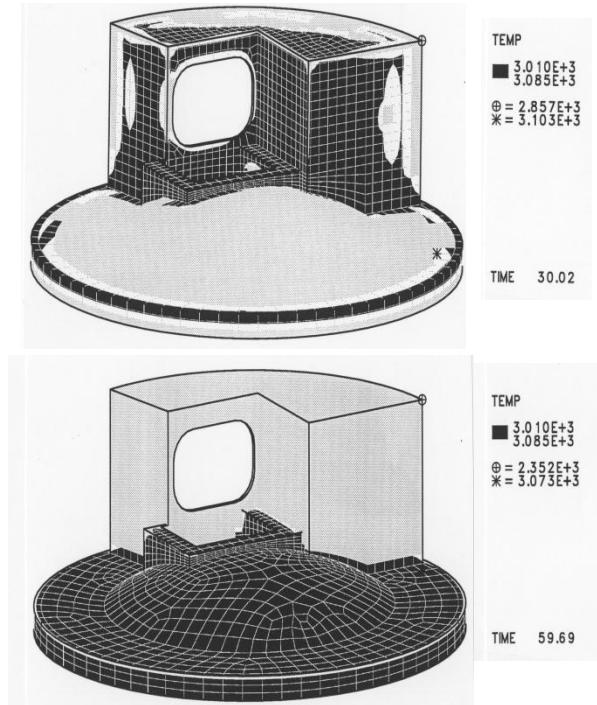


Figure 3.12.22: Liquidus to solidus temperature interval (in °F) (mushy zone) during cooling of a casting.

Problems

- 3.1** Show that the weak form of Eq. (3.1.1) is given by Eq. (3.2.1). Use the following component form of the Green–Gauss theorem to integrate-by-parts,

$$\int_{\Omega^e} \frac{\partial}{\partial x_i} (w F_i) \, d\mathbf{x} = \oint_{\Gamma^e} (w F_i) n_i \, ds, \quad (i = 1, 2, 3)$$

- 3.2** Using the α –family of approximation,

$$\mathbf{T}^{n+1} = \mathbf{T}^n + \Delta t[(1 - \alpha)\dot{\mathbf{T}}^n + \alpha\dot{\mathbf{T}}^{n+1}], \quad 0 \leq \alpha \leq 1$$

in Eq. (3.2.4), derive the fully discretized set of equations in the form

$$\hat{\mathbf{K}}(\mathbf{T}^{n+1})\mathbf{T}^{n+1} = \hat{\mathbf{F}}(\mathbf{T}^n)$$

Define $\hat{\mathbf{K}}(\mathbf{T}^{n+1})$ and $\hat{\mathbf{F}}(\mathbf{T}^n)$ in terms of the original matrices, time increment Δt , and the parameter α .

3.3 Specialize the results of Problem 3.2 to the case of $\alpha = 0$ and verify Eq. (3.7.10).

3.4 Specialize the results of Problem 3.2 to the case of $\alpha = 1$ and verify Eq. (3.7.12).

3.5 (*Newton's Method*) Rewrite Eq. (3.7.1) as

$$\mathbf{R}(\mathbf{T}) \equiv \hat{\mathbf{K}}(\mathbf{T})\mathbf{T} - \hat{\mathbf{F}}(\mathbf{T}) = \mathbf{0} \quad (1)$$

Newton's method is based on a truncated Taylor's series expansion of the residual $\mathbf{R}(\mathbf{T})$ about the known solution \mathbf{T}^r , where r denotes the iteration number:

$$\mathbf{0} = \mathbf{R}(\mathbf{T}^r) + \frac{\partial \mathbf{R}}{\partial \mathbf{T}} \Big|_{\mathbf{T}^r} \delta \mathbf{T} + O(\delta \mathbf{T})^2 \quad (2)$$

Here $\delta \mathbf{T} = (\mathbf{T}^{r+1} - \mathbf{T}^r)$. Omitting the terms of order two and higher, write

$$\mathbf{R}(\mathbf{T}^r) = -\frac{\partial \mathbf{R}}{\partial \mathbf{T}} \Big|_{\mathbf{T}^r} (\mathbf{T}^{r+1} - \mathbf{T}^r) \equiv -\mathbf{J}(\mathbf{T}^r)(\mathbf{T}^{r+1} - \mathbf{T}^r) \quad (3)$$

where \mathbf{J} is the Jacobian matrix (also known as the tangent matrix),

$$\mathbf{J} = \frac{\partial \mathbf{R}}{\partial \mathbf{T}} \Big|_{\mathbf{T}^r} \quad (4)$$

Show that

$$\mathbf{T}^{r+1} = \mathbf{T}^r - \mathbf{J}^{-1}(\mathbf{T}^r)\mathbf{R}(\mathbf{T}^r) \quad (5)$$

3.6 Derive the tangent matrix for the case where $\hat{\mathbf{K}}$ is defined by Eq. (3.2.5a) with k_{mn} given by

$$k_{mn} = \delta_{mn}(k_0 + k_1 T^p)$$

where k_0 and k_1 are constants, and p is an integer.

3.7 The energy equation for simultaneous conduction and radiation in a participating medium can be expressed by

$$-\nabla \cdot [k_e(T)\nabla T] = Q \quad (1)$$

where

$$k_e(T) = k + \frac{16\sigma n^2 T^3}{3\beta} \quad (2)$$

Here T is the temperature, Q is the internal heat generation, n denotes the refractive index of the medium, σ is the Stefan–Boltzmann constant, and β is the Roseland mean extinction coefficient. Develop the finite element model of the equation and determine the tangent coefficient matrix for a planar (two-dimensional) domain.

3.8 Repeat Problem 3.7 for a radially axisymmetric domain.

References for Additional Reading

1. E. B. Becker, G. F. Carey, and J. T. Oden, *Finite Elements, an Introduction*, Vol. I, Prentice Hall, Englewood Cliffs, New Jersey (1981).
2. T. J. R. Hughes, *The Finite Element Method, Linear Static and Dynamic Finite Element Analysis*, Dover Publications, New Jersey (2000).
3. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw–Hill, New York (2006).

4. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis*, Oxford University Press, Oxford, UK (2004).
5. O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, *The Finite Element Method: Its Basis and Fundamentals*, 6th ed., Elsevier, New York (2005).
6. M. J. Bluck and S. P. Walker, “Polynomial Basis Functions on Pyramidal Elements,” *Communications in Numerical Methods in Engineering*, **24**, 1827–1837 (2008).
7. I. Ergatoudis, B. M. Irons, and O. C. Zienkiewicz, “Curved, Isoparametric, ‘Quadrilateral’, Elements for Finite Element Analysis,” *International Journal of Solids and Structures*, **4**, 31–42 (1968).
8. B. M. Irons, “Engineering Applications of Numerical Integration in Stiffness Methods,” *AIAA Journal*, **14**, 2035–2037 (1966).
9. B. M. Irons, “Quadrature Rules for Brick-Based Finite Elements,” *International Journal for Numerical Methods in Engineering*, **3**, 293–294 (1971).
10. G. R. Cowper, “Gaussian Quadrature Formulas for Triangles,” *International Journal for Numerical Methods in Engineering*, **7**, 405–408 (1973).
11. P. C. Hammer, O. P. Marlowe, and A. H. Stroud, “Numerical Integration over Simplexes and Cones,” *Mathematics Tables Aids Computation*, National Research Council, Washington, D. C., **10**, 130–137 (1956).
12. A. N. Loxan, N. Davids, and A. Levenson, “Table of the Zeros of the Legendre Polynomials of Order 1–16 and the Weight Coefficients for Gauss’ Mechanical Quadrature Formula,” *Bulletin of the American Mathematical Society*, **48**, 739–743 (1942).
13. C. T. Reddy and D. J. Shippy, “Alternative Integration Formulae for Triangular Finite Elements,” *International Journal for Numerical Methods in Engineering*, **17**, 133–139 (1981).
14. B. Carnahan, H. A. Luther, and J. O. Wilkes, *Applied Numerical Methods*, John Wiley & Sons, New York (1969).
15. B. M. Irons, “A Frontal Solution Program for Finite Element Analysis,” *International Journal for Numerical Methods in Engineering*, **2**, 5–32 (1970).
16. P. Hood, “Frontal Solution Program for Unsymmetric Matrices,” *International Journal for Numerical Methods in Engineering*, **10**, 379–399 (1976).
17. Y. Hasbani and M. Engelman, “Out-of-Core Solution of Linear Equations with Non-Symmetric Coefficient Matrix,” *Computers and Fluids*, **7**, 13–31 (1979).
18. R. L. Taylor, E. L. Wilson, and S. J. Sackett, “Direct Solution of Equations by Frontal and Variable Band, Active Column Methods,” in *Proceedings of the U. S.-European Workshop on Nonlinear Finite Element Analysis in Structural Mechanics*, Bochum, W. Germany (1980).
19. G. E. Forsythe and C. B. Moler, *Computer Solution of Linear Algebraic Systems*, Prentice Hall, Englewood Cliffs, New Jersey (1967).
20. G. Strang, *Linear Algebra and Its Applications*, 2nd ed., Academic Press, New York (1980).
21. P. Gresho, R. Lee, R. Sani, and T. Stullich, “On the Time-Dependent FEM Solution of the Incompressible Navier–Stokes Equations in Two and Three Dimensions,” in *Recent Advances in Numerical Methods in Fluids*, Vol 1, Pineridge Press, Swansea (1980).
22. P. Gresho, D. Griffiths and D. Silvester, “Adaptive Time-Stepping for Incompressible Flow, Part I: Scalar Advection-Diffusion,” *SIAM Journal on Scientific Computing*, **30**, 2018–2054 (2008).
23. K. J. Bathe, *Numerical Methods in Finite Element Analysis*, Prentice Hall, Englewood Cliffs, New Jersey (1976).
24. T. Belytschko and T. J. R. Hughes (Eds.), *Computational Methods for Transient Analysis*, North-Holland, Amsterdam (1983).

25. J. H. Argyris and O. W. Scharpf, "Finite Elements in Time and Space," *The Aeronautical Journal of the Royal Society*, **73**, 1041–1044 (1969).
26. J. Donea, "On the Accuracy of Finite Element Solutions to the Transient Heat Conduction Equation," *International Journal for Numerical Methods in Engineering*, **8**, 103–110 (1974).
27. J. M. Bettencourt, O. C. Zienkiewicz, and G. Cantin, "Consistent Use of Finite Elements in Time and the Performance of Various Recurrence Schemes for the Heat Diffusion Equation," *International Journal for Numerical Methods in Engineering*, **17**, 931–938 (1981).
28. T. J. R. Hughes, "Unconditionally Stable Algorithms for Nonlinear Heat Conduction," *Computer Methods in Applied Mechanics and Engineering*, **10**, 135–139 (1977).
29. W. L. Wood and R. W. Lewis, "A Comparison of Time Marching Schemes for the Transient Heat Conduction Equation," *International Journal for Numerical Methods in Engineering*, **9**, 679–689 (1975).
30. W. L. Wood, "Control of Crank–Nicolson Noise in the Numerical Solution of the Heat Conduction Equation," *International Journal for Numerical Methods in Engineering*, **11**, 1059–1065 (1977).
31. 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," *Communications in Numerical Methods in Engineering*, **18**, 131–139 (2002).
32. A. M. P. Valli, G. F. Carey and A. L. G. A. Coutinho, "Control Strategies for Time Step Selection in Finite Element Simulation of Incompressible Flows and Coupled Reaction-Convection-Diffusion Processes," *International Journal for Numerical Methods in Fluids*, **47**, 201–231 (2005).
33. C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice Hall, Englewood Cliffs, New Jersey (1971).
34. P. D. Lax and R. D. Richtmyer, "Survey of the Stability of Linear Finite Difference Equations," *Communications in Pure and Applied Mathematics*, **9**, 267–293 (1956).
35. R. A. Bialecki, A. J. Kassab and A. Fic, "Proper Orthogonal Decomposition and Modal Analysis for Acceleration of Transient FEM Thermal Analysis," *International Journal for Numerical Methods in Engineering*, **62**, 774–797 (2005).
36. T. Bechtold, E. Rudnyi and J. Korvink, "Error Indicators for Fully Automatic Extraction of Heat-Transfer Macromodels for MEMS," *Journal of Micromechanics and Microengineering*, **15**, 430–440 (2005).
37. D. Botto, S. Zucca and M. Gola, "Reduced-Order Models for the Calculation of Thermal Transients of Heat Conduction/Convection FE Models," *Journal of Thermal Stresses*, **30**, 819–839 (2007).
38. R. E. Hogan and D. K. Gartling, "Solution Strategies for Coupled Conduction/Radiation Problems," *Communications in Numerical Methods in Engineering*, **24**, 523–542 (2008).
39. J. D. Osnes, "A Method for Efficiently Incorporating Radiative Boundaries in Finite Element Programs," in *Proceedings of the Third International Conference on Numerical Methods in Thermal Problems*, Seattle, Washington (1983).
40. A. Soria and P. Pegon, "Quasi-Newton Iterative Strategies Applied to the Heat Diffusion Equation," *International Journal for Numerical Methods in Engineering*, **30**, 661–677 (1990).
41. R. O. Stafford, A. B. Rice and D. F. Pinella, "Investment Casting Process Design - Part 2: Solidification Simulations," in *Proceedings of the 1987 ASME International Computers in Engineering Conference*, New York, NY (1987).
42. M. Engelman and M. A. Jamnia, "Grey-Body Surface Radiation Coupled with Conduction and Convection for General Geometries," *International Journal for Numerical Methods in Fluids*, **13**, 1029–1053 (1991).

43. A. B. Shapiro, "FACET – A Radiation View Factor Computer Code for Axisymmetric, 2D Planar and 3D geometries with Shadowing," Lawrence Livermore National Laboratory, UCID-19887, Livermore, CA (1983).
44. A. F. Emery, "VIEWC User's Manual," University of Washington, Seattle, WA (1988).
45. M. W. Glass, "Chaparral – A Library Package for Solving Large Enclosure Radiation Heat Transfer Problems," Sandia National Laboratories Report, SAND95-2049, Albuquerque, New Mexico (1995).
46. R. Siegel and J. R. Howell, *Thermal Radiation Heat Transfer*, McGraw–Hill, New York (1972).
47. A. F. Emery, D. Johansson, M. Lobo, and A. Abrous, "A Comparative Study of Methods for Computing the Diffuse Radiation Viewfactors for Complex Structures," *Journal of Heat Transfer*, **113**, 413–422 (1991).
48. K. Morgan, R. W. Lewis, and O. C. Zienkiewicz, "An Improved Algorithm for Heat Conduction Problems with Phase Change," *International Journal for Numerical Methods in Engineering*, **12**, 1191–1195 (1978).
49. W. Randolph and K. J. Bathe, "An Efficient Algorithm for Analysis of Nonlinear Heat Transfer with Phase Changes," *International Journal for Numerical Methods in Engineering*, **18**, 119–134 (1982).
50. J. N. Reddy, *Mechanics of Laminated Composite Plates and Shells. Theory and Analysis*, 2nd ed., CRC Press, Boca Raton, Florida (2004).
51. S. Kimura and A. Bejan, "The 'Heatline' Visualization of Convective Heat Transfer," *Journal of Heat Transfer*, **105**, 916–919 (1983).
52. M. M. Rashid, "The Arbitrary Local Mesh Replacement Method: An Alternative to Remeshing for Crack Propagation Analysis," *Computer Methods in Applied Mechanics and Engineering*, **154**, 133–150 (1998).
53. M. M. Rashid and P.M. Gullett, "On a Finite Element Method with Variable Element Topology," *Computer Methods in Appl. Mech. and Engineering*, **190**, 1509–1527 (2000).
54. C. R. Dohrmann and M. M. Rashid, "Polynomial Approximation of Shape Function Gradients from Element Geometries," *International Journal for Numerical Methods in Engineering*, **53**, 945–958 (2002).
55. I. S. Wichman, "On the Use of Operator-Splitting Methods for the Equations of Combustion," *Combustion and Flame*, **83**, 240–252 (1991).
56. T. R. Young, "CHEMEOQ — A Subroutine for Solving Stiff Ordinary Differential Equations," NRL Memorandum Report 4091, Naval Research Laboratory, Washington, DC (1980).
57. D. R. Mott, E. S. Oran and B. van Leer, "A Quasi-Steady-State Solver for the Stiff Ordinary Differential Equations of Reaction Kinetics," *Journal of Computational Physics*, **164**, 407–428 (2000).
58. D. K. Gartling, R. E. Hogan and M. W. Glass, "COYOTE — A Finite Element Computer Program for Nonlinear Heat Conduction Problems," Sandia National Laboratories Report, 2009-4926, Albuquerque, New Mexico (2009).
59. P. Roach, *Verification and Validation in Computational Science and Engineering*, Hermosa Publishers, Albuquerque, New Mexico (1998).
60. P. Knupp and K. Salari, *Verification of Computer Codes in Computational Science and Engineering*, Chapman and Hall/CRC, Boca Raton, Florida (2003).
61. J. Beck, K. Cole, A. Haji-Sheikh and B. Litkouhi, *Heat Conduction Using Green's Functions*, Hemisphere Press, Washington, DC (1992).
62. R. L. McMasters, K. J. Dowding, J. V. Beck and D. H. Yen, "Methodology to Generate Accurate Solutions for Verification in Transient Three-Dimensional Heat Conduction," *Numerical Heat Transfer*, **41**, 521–541 (2002).

63. J. Beck, "COND3D Users Manual for Three-Dimensional Subroutine for Exact Transient Heat Conduction Solutions in Parallelepipeds," Beck Engineering Consultants Report, Okemos, Michigan (2000).
64. H. S. Carslaw and J. C. Jaeger, *Conduction of Heat in Solids*, Oxford University Press, Oxford (1959).
65. D. A. Glowka and C. M. Stone, "Thermal Response of Polycrystalline Diamond Compact Cutters Under Simulated Downhole Conditions," *Society of Petroleum Engineering Journal*, 143–156 (1985).
66. G. A. Knorovsky and S. N. Burchett, "The Effect of Heat Sinks in GTA Microwelding," in *Proceedings of the Second International Conference on Trends in Welding*, American Welding Society, Gatlinburg, Tennessee (1989).
67. R. E. Hogan, M. W. Glass, P. R. Schunk, and D. K. Gartling, "Thermal Analysis of the Cooling and Solidification of Investment Castings," in *Proceedings of the First International Conference on Transport Phenomena in Processing*, Pacific Institute for Thermal Engineering, Honolulu, Hawaii (1992).

Flows of Viscous Incompressible Fluids

4.1 Introduction

4.1.1 Background

The field of fluid mechanics is concerned with the motion of fluids and their effect on the surroundings. A fluid state of matter is often characterized by the relative mobility of the molecules that constitute the matter. Fluids exist either in the form of a gas or a liquid; more complex materials, such as mixtures, may also have a fluid behavior. In this chapter, finite element models are developed based on the weak formulation of the equations of viscous, incompressible fluids under isothermal conditions.

The motion of a fluid is governed by the global laws of conservation of mass, momenta, and energy (see Chapter 1). These equations consist of a set of coupled, nonlinear, partial differential equations in terms of the velocity components, temperature, and pressure. When temperature effects are not important, the energy equation is uncoupled from the momentum (i.e., the Navier–Stokes) equations. Thus for isothermal flows, we need to solve only the Navier–Stokes equations and continuity equation. When the Reynolds number for the flow is very low, the nonlinear terms due to inertial effects can be neglected, resulting in a linear boundary value problem. Such a flow is termed *Stokes flow*.

4.1.2 Governing Equations

The laws governing the flow of Newtonian fluids were reviewed in Chapter 1. The equations are specialized here for viscous fluids that are subject to the assumption of incompressibility. We will use the phrases *incompressible fluid* and *incompressible flow* interchangeably, while recognizing the subtle difference in meaning. The equations are written here for a fluid region Ω both in vector as well as Cartesian component form in an Eulerian reference frame, $\mathbf{x} = (x_1, x_2, x_3)$.

Conservation of Mass

$$\nabla \cdot \mathbf{v} = 0 \quad (4.1.1a)$$

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (4.1.1b)$$

Conservation of Momentum

$$\rho_0 \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} \right) - \nabla \cdot \boldsymbol{\sigma} + \rho_0 \mathbf{f} = 0 \quad (4.1.2a)$$

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) - \frac{\partial \sigma_{ij}}{\partial x_j} + \rho_0 f_i = 0 \quad (4.1.2b)$$

Constitutive Equations

$$\begin{aligned}\sigma &= \tau - P\mathbf{I} ; \quad \tau = 2\mu\mathbf{D} \\ \mathbf{D} &= \frac{1}{2} [(\nabla\mathbf{v}) + (\nabla\mathbf{v})^T]\end{aligned}\quad (4.1.3a)$$

$$\begin{aligned}\sigma_{ij} &= \tau_{ij} - P\delta_{ij} ; \quad \tau_{ij} = \mu D_{ij} \\ D_{ij} &= \frac{1}{2} \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right)\end{aligned}\quad (4.1.3b)$$

In Eqs. (4.1.1)–(4.1.3), \mathbf{v} denotes the velocity vector, σ is the total stress tensor, τ is the viscous stress tensor, P is the pressure, \mathbf{f} is the body force vector (per unit mass), ρ_0 is the density, and μ is the shear viscosity of the fluid. The Cartesian components of the dependent variables are obvious from the above equations. In writing the Cartesian component form of the equations, the index notation with summation on repeated indices is used. The operator D/Dt denotes the material (or convective) time derivative,

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + \mathbf{v} \cdot \nabla, \quad \frac{D}{Dt} \equiv \frac{\partial}{\partial t} + v_j \frac{\partial}{\partial x_j} \quad (4.1.4)$$

Equation (4.1.1a) or (4.1.1b) is also known as the *continuity equation*, incompressibility constraint, or divergence-free condition on the velocity field. Equations in (4.1.2), with the constitutive relation indicated in Eqs. (4.1.3a,b), are known as the *Navier–Stokes equations*.

The boundary conditions are given by

$$\mathbf{v} = \mathbf{f}^v \quad \text{or} \quad v_i = f_i^v(s_j, t) \quad \text{on } \Gamma_v \quad (4.1.5a)$$

$$\mathcal{T} \equiv \sigma \cdot \hat{\mathbf{n}} = \mathbf{f}^T \quad \text{or} \quad \mathcal{T}_i \equiv \sigma_{ij}(s_j, t)n_j(s_j) = f_i^T(s_j, t) \quad \text{on } \Gamma_T \quad (4.1.5b)$$

where $\hat{\mathbf{n}}$ is the unit normal to the boundary and Γ_v and Γ_T are defined in Section 1.10.

The vector form of the equations in (4.1.1)–(4.1.3) can be written in any coordinate system that is suitable for the description and solution of the problem under consideration (see Section 1.5). Using the constitutive equations and deformation rate-velocity relations (4.1.3), Eqs. (4.1.1) and (4.1.2) can be expressed solely in terms of the velocity components and pressure. These are summarized below:

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (4.1.6)$$

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) - \frac{\partial}{\partial x_j} \left[-P\delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 f_i = 0 \quad (4.1.7)$$

It is possible to express these equations in terms of other variables, such as the stream function and vorticity. For example, in the two-dimensional case, we could write the momentum Eq. (4.1.7) in terms of the velocity \mathbf{v} and *vorticity* ω as

$$\frac{\partial \omega}{\partial t} + \mathbf{v} \cdot \nabla \omega = \nu \nabla^2 \omega \quad (4.1.8)$$

where ν is the kinematic viscosity. In two dimensions, ω is a scalar function defined by

$$\omega \equiv \frac{\partial v_1}{\partial x_2} - \frac{\partial v_2}{\partial x_1} \quad (4.1.9)$$

Notice that in deriving (4.1.8) (by taking the curl of the momentum equations and using several vector identities) the pressure has been eliminated from the momentum equation. This is a situation that is highly sought after in some computational schemes, because of the difficulties associated with the pressure variable in an incompressible flow. A three-dimensional version of the vorticity transport equation is possible, although it has seen relatively little use in computation due to the complexity of the vorticity boundary conditions. Again, in two dimensions, a stream function, ψ , may be defined such that the continuity equation (4.1.6) is identically satisfied. Using both the vorticity and stream function definitions, the Navier–Stokes equations can be expressed as the pair consisting of Eq. (4.1.8) and

$$\nabla^2 \psi = -\omega \quad (4.1.10)$$

with the stream function defined by

$$v_2 \equiv -\frac{\partial \psi}{\partial x_1}, \quad v_1 \equiv \frac{\partial \psi}{\partial x_2}, \quad \omega \equiv \frac{\partial v_1}{\partial x_2} - \frac{\partial v_2}{\partial x_1} \quad (4.1.11)$$

The symbol ψ should not be confused with ψ_i used for interpolation functions; it should be clear from the context of the discussion. The introduction of these variables in the two-dimensional case reduces the number of equations that have to be solved from three to two. For the three-dimensional case, the stream function and vorticity are vector functions with three components each, and thus there is no reduction in the number of variables. The pressure, if required, may be computed from a Poisson equation of the form

$$\nabla^2 P = \nabla \cdot (\mathbf{f} + \nu \nabla^2 \mathbf{v} - \mathbf{v} \cdot \nabla \mathbf{v}) \quad (4.1.12)$$

which is obtained by taking the divergence of the momentum equation. It is also possible to rewrite the momentum equation entirely in terms of the stream function and reduce the flow problem to a single equation. Unfortunately, this approach leads to a fourth-order equation that increases the complexity of any computational method. Since these alternative formulations are not as computationally convenient as the original equations, especially for the imposition of boundary conditions, we will confine our attention here to the development of finite element models for Eqs. (4.1.6) and (4.1.7).

In the present study we shall consider two different finite element models of Eqs. (4.1.6) and (4.1.7), both of which may be considered *weak form Galerkin finite element methods* or *WKGDEM*. The first one is a natural formulation in which the weak forms of Eqs. (4.1.6) and (4.1.7) are used to construct the finite element model. The resulting finite element model is termed the *velocity-pressure model* or *mixed model*. The phrase “mixed” is used because velocity variables are mixed with the force-like variable, pressure, and both types of variables are retained in a single formulation. The second model is based on the interpretation that the

continuity equation (4.1.6) is an additional relation among the velocity components (i.e., a constraint among the v_i), and the constraint is satisfied in a least-squares (i.e., approximate) sense. This particular method of including the constraint in the formulation is known as the *penalty function method*, and the model is termed the *penalty-finite element model* (see [1–4]). In this case, the pressure variable is effectively eliminated from the formulation. It is informative to note that the velocity-pressure (or mixed) formulation is the same as the Lagrange multiplier formulation, wherein the constraint is included by means of the Lagrange multiplier method. The Lagrange multiplier turns out to be the negative of the pressure.

Before proceeding, it should be noted that there are other types of finite element models that have been developed in recent years. Of growing interest, especially for compressible applications, are *discontinuous Galerkin methods*. This type of finite element models is well described in the book by Li [5] and will not be discussed here. Variational Multiscale methods (VMS) have also become a recognized approach in finite element analysis of fluid flow problems and these will be considered in a later section that deals with stabilization. Finally, finite element models based on least-squares formulations have been developed by various researchers [6–15], including the *k-version finite element models* introduced and advanced by Surana and Reddy [7,9,14], and they will be discussed briefly later in this chapter.

4.2 Mixed Finite Element Model

4.2.1 Weak Form

The starting point for the development of the finite element models of Eqs. (4.1.6) and (4.1.7) is their weak forms. As discussed in Section 2.4, the weak forms are developed using a three-step procedure. These steps are briefly reviewed here. For convenience, we denote expressions on the left side of Eqs. (4.1.6) and (4.1.7) by f_1 and \mathbf{f}_2 , respectively. The weighted-integral statements of the two equations over a typical element Ω^e are given by

$$\int_{\Omega^e} Q f_1 \, d\mathbf{x} = 0 \quad (4.2.1)$$

$$\int_{\Omega^e} \mathbf{w} \cdot \mathbf{f}_2 \, d\mathbf{x} = 0 \quad (4.2.2)$$

where (Q, \mathbf{w}) are weight functions, which will be equated, in the weak form Galerkin finite element models, to the interpolation functions used for (P, \mathbf{v}) , respectively (see Reddy [1,2] for details).

To obtain the weak form of Eq. (4.1.7), we use integration-by-parts to equally distribute integration between the dependent variables and the weight functions. However, in any problem, such trading of differentiability is subjected to the restriction that the resulting boundary expressions are physically meaningful. Otherwise, the secondary variables of the formulation may not be the quantities the physical problem admits as the boundary conditions. An examination of the boundary stress components T_i in Eq. (4.1.5) shows that the pressure term occurs as a part of the expression [see Eq. (4.1.3)]. For example, T_1 is the x_1 -component of the total boundary stress, which is the sum of the viscous boundary stress and

the hydrostatic boundary stress,

$$\mathcal{T}_1 = \mu \left[2 \frac{\partial v_1}{\partial x_1} n_1 + \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) n_2 + \left(\frac{\partial v_1}{\partial x_3} + \frac{\partial v_3}{\partial x_1} \right) n_3 \right] + (-P n_1) \quad (4.2.3)$$

Here (n_1, n_2, n_3) denote the components of the unit normal vector on the boundary. It is clear from the above expression that the entire expression in the square brackets of Eq. (4.1.7) must be integrated by parts in the momentum equations to obtain \mathcal{T}_i in the boundary expression. No integration-by-parts is used in the continuity equation (4.1.6) because no relaxation of differentiability on v_i can be accomplished; further, the resulting boundary conditions will not be physically justifiable.

Keeping the above comments in mind, we carry out the remaining two steps of the weak formulation, and obtain the following integral statements:

$$0 = \int_{\Omega^e} Q \frac{\partial v_i}{\partial x_i} d\mathbf{x} \quad (4.2.4)$$

$$0 = \int_{\Omega^e} \left\{ \rho_0 \left(w_i \frac{\partial v_i}{\partial t} + w_i v_j \frac{\partial v_i}{\partial x_j} \right) + \frac{\partial w_i}{\partial x_j} \left[-P \delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] - \rho_0 w_i f_i \right\} d\mathbf{x} \\ - \oint_{\Gamma^e} w_i \mathcal{T}_i ds \quad (4.2.5)$$

This completes the weak form development.

4.2.2 Finite Element Model

Since we are developing the weak form Galerkin finite element models, the choice of the weight functions is restricted to the spaces of approximation functions used for the pressure and velocity fields. Suppose that the dependent variables (v_i , P) are approximated by expansions of the form

$$v_i(\mathbf{x}, t) = \sum_{m=1}^M \psi_m(\mathbf{x}) v_i^m(t) = \boldsymbol{\Psi}^T \mathbf{v}_i \quad (4.2.6a)$$

$$P(\mathbf{x}, t) = \sum_{l=1}^L \phi_l(\mathbf{x}) P_l(t) = \boldsymbol{\Phi}^T \mathbf{P} \quad (4.2.6b)$$

where $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ are (column) vectors of interpolation (or shape) functions, \mathbf{v}_i and \mathbf{P} are vectors of nodal values of velocity components and pressure, respectively, and the superscript $(\cdot)^T$ denotes a transpose of the enclosed vector or matrix. The weight functions (Q , \mathbf{w}) have the following correspondence (see [1] for further details),

$$Q \approx \phi_l, \mathbf{w} \approx \boldsymbol{\Psi} \quad (4.2.7)$$

Substitution of Eqs. (4.2.6) and (4.2.7) into Eqs. (4.2.4) and (4.2.5) results in the following finite element equations:

Continuity

$$-\left[\int_{\Omega^e} \boldsymbol{\Phi} \frac{\partial \boldsymbol{\Psi}^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_i = \mathbf{0} \quad (4.2.8)$$

i-th Momentum

$$\begin{aligned} & \left[\int_{\Omega^e} \rho_0 \Psi \Psi^T d\mathbf{x} \right] \dot{\mathbf{v}}_i + \left[\int_{\Omega^e} \rho_0 \Psi (\Psi^T \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \right] \mathbf{v}_i + \left[\int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \right] \mathbf{v}_i \\ & + \left[\int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_j - \left[\int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x} \right] \mathbf{P} = \left\{ \int_{\Omega^e} \rho_0 \Psi f_i d\mathbf{x} \right\} + \left\{ \oint_{\Gamma^e} \mathcal{T}_i \Psi ds \right\} \end{aligned} \quad (4.2.9)$$

where the superposed dot represents a time derivative. The above equations can be written symbolically in matrix form as

Continuity

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (4.2.10)$$

Momentum

$$\mathbf{M} \dot{\mathbf{v}} + \mathbf{C}(\mathbf{v}) \mathbf{v} + \mathbf{K} \mathbf{v} - \mathbf{Q} \mathbf{P} = \mathbf{F} \quad (4.2.11)$$

where $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}^T$. For the three-dimensional case, Eqs. (4.2.10) and (4.2.11) have the following explicit matrix form [the continuity equation (4.2.10) and momentum equation (4.2.11) are combined into one]:

$$\begin{aligned} \begin{Bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \mathbf{F}_3 \\ \mathbf{0} \end{Bmatrix} &= \begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{u}}_1 \\ \dot{\mathbf{u}}_2 \\ \dot{\mathbf{u}}_3 \\ \dot{\mathbf{P}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P} \end{Bmatrix} + \\ & \begin{bmatrix} 2\mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{12} & \mathbf{K}_{31} & -\mathbf{Q}_1 \\ \mathbf{K}_{21} & \mathbf{K}_{11} + 2\mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{23} & -\mathbf{Q}_2 \\ \mathbf{K}_{31} & \mathbf{K}_{32} & \mathbf{K}_{11} + \mathbf{K}_{22} + 2\mathbf{K}_{33} & -\mathbf{Q}_3 \\ -\mathbf{Q}_1^T & -\mathbf{Q}_2^T & -\mathbf{Q}_3^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P} \end{Bmatrix} \end{aligned} \quad (4.2.12)$$

The coefficient matrices shown in Eq. (4.2.12) are defined by

$$\begin{aligned} \mathbf{M} &= \int_{\Omega^e} \rho_0 \Psi \Psi^T d\mathbf{x}; \quad \mathbf{C}(\mathbf{v}) = \int_{\Omega^e} \rho_0 \Psi (\Psi^T \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \\ \mathbf{K}_{ij} &= \int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x}; \quad \mathbf{Q}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x} \\ \mathbf{F}_i &= \int_{\Omega^e} \rho_0 \Psi f_i d\mathbf{x} + \oint_{\Gamma^e} \Psi \mathcal{T}_i ds \end{aligned} \quad (4.2.13)$$

where a sum on repeated indices is implied in the definition of \mathbf{C} .

The two sets of interpolation functions used in Eq. (4.2.6) should be of the Lagrange type, i.e., derived by interpolating only the values of the functions, and not their derivatives. There are two different finite elements associated with the two field variables (v_i, P), hence there are two different finite element meshes corresponding to the two variables over the same domain, Ω . If one of the meshes contains the other mesh as a subset, then we choose to display the first mesh and indicate the nodal degrees of freedom associated with the nodes of a typical element of the mesh.

The interpolation used for the pressure variable should be different from that used for the velocities, because the weak form in Eqs. (4.2.4) and (4.2.5) contain the first derivatives of the velocities v_i and no derivatives of the pressure P . In addition, the essential boundary conditions of the formulation do not include specification of the pressure; it enters the boundary conditions as a part of the natural boundary conditions. This implies that the pressure variable need not be carried as a variable that is continuous across interelement boundaries. These observations lead to the conclusion that the pressure variable should be interpolated with functions that are one order less than those used for the velocity field and that the approximation may be discontinuous (i.e., not continuous from one element to other). Thus, quadratic interpolation of v_i and discontinuous linear interpolation of P are admissible. Models that use equal interpolation of the velocities and pressure with this formulation are known to give inaccurate results for the pressure (see [6–16]). This heuristic argument for unequal interpolation anticipates the formal theory, which will be cited in Section 4.5.2 with the discussion on appropriate interpolation functions.

Returning to Eqs. (4.2.10) and (4.2.11), they can be combined into one,

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}} \\ \mathbf{P} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{u}) + \mathbf{K} & -\mathbf{Q} \\ -\mathbf{Q}^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F} \\ \mathbf{0} \end{Bmatrix} \quad (4.2.14)$$

or in a more symbolic format as

$$\bar{\mathbf{M}}\dot{\mathbf{U}} + \bar{\mathbf{K}}\mathbf{U} = \bar{\mathbf{F}} \quad (4.2.15a)$$

where

$$\mathbf{U} = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{P}\}^T, \quad \bar{\mathbf{M}} = \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \bar{\mathbf{K}} = \begin{bmatrix} \mathbf{C}(\mathbf{u}) + \mathbf{K} & -\mathbf{Q} \\ -\mathbf{Q}^T & \mathbf{0} \end{bmatrix} \quad (4.2.15b)$$

The general form of Eq. (4.2.15a) is the same as the nonlinear diffusion equation (3.6.3). Therefore, the time-approximation schemes discussed in Section 3.7.3 are readily applicable to the ordinary differential equations in (4.2.14) or (4.2.15a).

4.3 Penalty Finite Element Models

4.3.1 Introduction

The penalty function method, like the Lagrange multiplier method, allows us to reformulate a problem with constraints as one without constraints (see [1–4, 23–32]). The problem at hand, namely, the flow of a viscous incompressible fluid, must first be restated as a variational problem subjected to a constraint. For the purpose of describing the penalty function method, we consider the steady Stokes flow problem (i.e., without time-dependent and nonlinear terms) in two dimensions:

Conservation of mass

$$\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} = 0 \quad (4.3.1)$$

Conservation of momentum

$$-\frac{\partial}{\partial x_1} \left(2\mu \frac{\partial v_1}{\partial x_1} \right) - \frac{\partial}{\partial x_2} \left[\mu \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) \right] + \frac{\partial P}{\partial x_1} - \rho_0 f_1 = 0 \quad (4.3.2a)$$

$$-\frac{\partial}{\partial x_1} \left[\mu \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) \right] - \frac{\partial}{\partial x_2} \left(2\mu \frac{\partial v_2}{\partial x_2} \right) + \frac{\partial P}{\partial x_2} - \rho_0 f_2 = 0 \quad (4.3.2b)$$

The weak form of these equations, also known as the *variational problem*, is given by (see Reddy [1,2]):

$$B((w_1, w_2, Q), (v_1, v_2, P)) = \ell(w_1, w_2, Q) \quad (4.3.3)$$

where (w_1, w_2, Q) are the weight functions used for the momentum and continuity equations, respectively. Also, $B(\cdot, \cdot)$ is a *bilinear form* [i.e., an expression that is linear in (w_1, w_2, Q) and (v_1, v_2, P)] and $\ell(\cdot)$ is a *linear form*, that in the present case are defined by

$$\begin{aligned} B(\cdot, \cdot) &= \mu \int_{\Omega^e} \left[2 \left(\frac{\partial w_1}{\partial x_1} \frac{\partial v_1}{\partial x_1} + \frac{\partial w_2}{\partial x_2} \frac{\partial v_2}{\partial x_2} \right) + \left(\frac{\partial w_1}{\partial x_2} + \frac{\partial w_2}{\partial x_1} \right) \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) \right] d\mathbf{x} \\ &\quad - \int_{\Omega^e} \left[\left(\frac{\partial w_1}{\partial x_1} + \frac{\partial w_2}{\partial x_2} \right) P + \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} \right) Q \right] d\mathbf{x} \end{aligned} \quad (4.3.4a)$$

$$\ell(\cdot) = \int_{\Omega^e} \rho_0 (f_1 w_1 + f_2 w_2) d\mathbf{x} + \oint_{\Gamma^e} (\mathcal{T}_1 w_1 + \mathcal{T}_2 w_2) ds \quad (4.3.4b)$$

and $(\mathcal{T}_1, \mathcal{T}_2)$ are the boundary stress components,

$$\begin{aligned} \mathcal{T}_1 &= \left(2\mu \frac{\partial v_1}{\partial x_1} - P \right) n_1 + \mu \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) n_2 \\ \mathcal{T}_2 &= \mu \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) n_1 + \left(2\mu \frac{\partial v_2}{\partial x_2} - P \right) n_2 \end{aligned} \quad (4.3.5)$$

The finite element model based on the weak form (4.3.3) is a special case of the mixed finite element model given in Eq. (4.2.14). Equation (4.2.14) is more general than the problem at hand in that Eq. (4.2.14) is valid for time-dependent Navier–Stokes equations. In the interest of completeness, the penalty function method is described first using the steady Stokes problem. The results will then be generalized to the time-dependent Navier–Stokes equations. We describe two different penalty finite element models of flows of viscous, incompressible fluids, namely, the reduced integration penalty model and the consistent penalty model.

4.3.2 Penalty Function Method

Suppose that the velocity field (v_1, v_2) is constrained to satisfy the continuity equation (4.3.1) identically. Then the weight functions (w_1, w_2) (being virtual variations of the velocity components) also satisfy the constraint condition (continuity equation)

$$\frac{\partial w_1}{\partial x_1} + \frac{\partial w_2}{\partial x_2} = 0 \quad (4.3.6)$$

As a result, the second integral expression in the bilinear form (4.3.4a) drops out, and the pressure (and hence the weight function Q) does not appear explicitly in the variational problem (4.3.3). The resulting variational problem now can be stated as

follows: among all vectors $\mathbf{v} = (v_1, v_2)$ which satisfy the continuity equation (4.3.1), find the one that satisfies the variational problem

$$B_0((w_1, w_2), (v_1, v_2)) = \ell_0(w_1, w_2) \quad (4.3.7)$$

for all admissible weight functions $\mathbf{w} = (w_1, w_2)$ [i.e., those which satisfy the condition in Eq. (4.3.6)]. The bilinear and linear forms are defined by

$$B_0(\mathbf{w}, \mathbf{v}) = \mu \int_{\Omega^e} \left[2 \left(\frac{\partial w_1}{\partial x_1} \frac{\partial v_1}{\partial x_1} + \frac{\partial w_2}{\partial x_2} \frac{\partial v_2}{\partial x_2} \right) + \left(\frac{\partial w_1}{\partial x_2} + \frac{\partial w_2}{\partial x_1} \right) \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) \right] d\mathbf{x} \quad (4.3.8a)$$

$$\ell_0(\mathbf{w}) = \int_{\Omega^e} \rho_0(f_1 w_1 + f_2 w_2) d\mathbf{x} + \oint_{\Gamma^e} (\mathcal{T}_1 w_1 + \mathcal{T}_2 w_2) ds \quad (4.3.8b)$$

Equation (4.3.7) is called a constrained variational problem, because the solution vector \mathbf{v} is constrained to satisfy the continuity equation. We note that the bilinear form in Eq. (4.3.8a) is symmetric: $B_0(\mathbf{w}, \mathbf{v}) = B_0(\mathbf{v}, \mathbf{w})$,

$$B_0((w_1, w_2), (v_1, v_2)) = B_0((v_1, v_2), (w_1, w_2)) \quad (4.3.9)$$

Whenever the bilinear form of a variational problem is symmetric in its arguments, it is possible to construct a quadratic functional such that the minimum of the quadratic functional is equivalent to the variational problem (see Reddy [1,2]). The quadratic functional for the problem at hand is given by the expression

$$I_0(\mathbf{v}) = \frac{1}{2} B_0(\mathbf{v}, \mathbf{v}) - \ell_0(\mathbf{v}) \quad (4.3.10)$$

where the velocity vector \mathbf{v} satisfies the continuity equation (4.3.1). Now we can state that Eqs. (4.3.1) and (4.3.2) governing the steady flow of viscous incompressible fluids are equivalent to minimizing the quadratic functional $I_0(\mathbf{v})$ subjected to the constraint

$$G(\mathbf{v}) \equiv \frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} = 0 \quad (4.3.11)$$

It should be remembered that the discussion presented in this section is directed toward the reformulation of the problem as one of a constrained problem so that the penalty function method can be used. As can be seen from the above development, the advantage of the formulation is that the pressure variable does not appear in the model. However, there are certain computational complications associated with the penalty finite element model. These will be discussed shortly.

In the penalty function method, the constrained problem is reformulated as an unconstrained problem as follows: minimize the modified functional

$$I_P(\mathbf{v}) \equiv I_0(\mathbf{v}) + \frac{\gamma_e}{2} \int_{\Omega^e} [G(\mathbf{v})]^2 d\mathbf{x} \quad (4.3.12)$$

where γ_e is called the *penalty parameter*. Note that the constraint is included in a least-squares sense into the functional. Seeking the minimum of the modified

functional $I_P(\mathbf{v})$ is equivalent to seeking the minimum of both $I_0(\mathbf{v})$ and $G(\mathbf{v})$, the latter with respect to the weight γ_e . The larger the value of γ_e , the more exactly the constraint is satisfied. The necessary condition for the minimum of I_P is

$$\delta I_P = 0 \quad (4.3.13)$$

We have

$$\begin{aligned} 0 &= \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_1}{\partial x_1} \frac{\partial v_1}{\partial x_1} + \mu \frac{\partial \delta v_1}{\partial x_2} \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) \right] d\mathbf{x} - \int_{\Omega^e} \rho_0 f_1 \delta v_1 d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_1 T_1 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_1}{\partial x_1} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} \right) d\mathbf{x} \end{aligned} \quad (4.3.14a)$$

$$\begin{aligned} 0 &= \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_2}{\partial x_2} \frac{\partial v_2}{\partial x_2} + \mu \frac{\partial \delta v_2}{\partial x_1} \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) \right] d\mathbf{x} - \int_{\Omega^e} \rho_0 f_2 \delta v_2 d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_2 T_2 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_2}{\partial x_2} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} \right) d\mathbf{x} \end{aligned} \quad (4.3.14b)$$

These two statements provide the weak forms for the penalty finite element model with $\delta v_1 = w_1$ and $\delta v_2 = w_2$. We note that the pressure does not appear explicitly in the weak form (4.3.14), although it is a part of the boundary stresses (4.3.5). An approximation for the pressure can be post-computed from the relation (see [1,2,4,16,33,34])

$$P = -\gamma_e \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} \right) \quad (4.3.15)$$

where $\mathbf{v} = \mathbf{v}(\gamma_e)$ is the solution of Eq. (4.3.14).

The time derivative terms and nonlinear terms can be added to Eq. (4.3.14) without affecting the above discussion. For the general three-dimensional case, the weak forms are given by

$$\begin{aligned} 0 &= \int_{\Omega^e} \rho_0 \delta v_1 \left[\frac{\partial v_1}{\partial t} + \left(v_1 \frac{\partial v_1}{\partial x_1} + v_2 \frac{\partial v_1}{\partial x_2} + v_3 \frac{\partial v_1}{\partial x_3} \right) - f_1 \right] d\mathbf{x} \\ &\quad + \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_1}{\partial x_1} \frac{\partial v_1}{\partial x_1} + \mu \frac{\partial \delta v_1}{\partial x_2} \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) + \mu \frac{\partial \delta v_1}{\partial x_3} \left(\frac{\partial v_1}{\partial x_3} + \frac{\partial v_3}{\partial x_1} \right) \right] d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_1 T_1 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_1}{\partial x_1} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) d\mathbf{x} \end{aligned} \quad (4.3.16a)$$

$$\begin{aligned} 0 &= \int_{\Omega^e} \rho_0 \delta v_2 \left[\frac{\partial v_2}{\partial t} + \left(v_1 \frac{\partial v_2}{\partial x_1} + v_2 \frac{\partial v_2}{\partial x_2} + v_3 \frac{\partial v_2}{\partial x_3} \right) - f_2 \right] d\mathbf{x} \\ &\quad + \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_2}{\partial x_2} \frac{\partial v_2}{\partial x_2} + \mu \frac{\partial \delta v_2}{\partial x_1} \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) + \mu \frac{\partial \delta v_2}{\partial x_3} \left(\frac{\partial v_2}{\partial x_3} + \frac{\partial v_3}{\partial x_2} \right) \right] d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_2 T_2 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_2}{\partial x_2} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) d\mathbf{x} \end{aligned} \quad (4.3.16b)$$

$$\begin{aligned} 0 &= \int_{\Omega^e} \rho_0 \delta v_3 \left[\frac{\partial v_3}{\partial t} + \left(v_1 \frac{\partial v_3}{\partial x_1} + v_2 \frac{\partial v_3}{\partial x_2} + v_3 \frac{\partial v_3}{\partial x_3} \right) - f_3 \right] d\mathbf{x} \\ &\quad + \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_3}{\partial x_3} \frac{\partial v_3}{\partial x_3} + \mu \frac{\partial \delta v_3}{\partial x_1} \left(\frac{\partial v_1}{\partial x_3} + \frac{\partial v_3}{\partial x_1} \right) + \mu \frac{\partial \delta v_3}{\partial x_2} \left(\frac{\partial v_2}{\partial x_3} + \frac{\partial v_3}{\partial x_2} \right) \right] d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_3 T_3 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_3}{\partial x_3} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) d\mathbf{x} \end{aligned} \quad (4.3.16c)$$

This completes the description of the penalty function method as applied to viscous incompressible flows.

4.3.3 Reduced Integration Penalty Model

The basic penalty method may be implemented by either of two procedures. Historically, the first computations were carried out by directly using the previous variational statement and some numerical techniques to make the resulting computational problem tractable. The so-called reduced integration penalty (RIP) finite element model is obtained from Eqs. (4.3.16a-c) by substituting finite element interpolation (4.2.6a) for the velocity field, and setting $\delta v_i = \Psi$:

$$\begin{aligned} & \left[\begin{array}{ccc} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M} \end{array} \right] \left\{ \begin{array}{c} \dot{\mathbf{v}}_1 \\ \dot{\mathbf{v}}_2 \\ \dot{\mathbf{v}}_3 \end{array} \right\} + \left[\begin{array}{ccc} \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}(\mathbf{v}) \end{array} \right] \left\{ \begin{array}{c} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{array} \right\} + \\ & \left[\begin{array}{ccc} 2\mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{21} & \mathbf{K}_{31} \\ \mathbf{K}_{12} & \mathbf{K}_{11} + 2\mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{32} \\ \mathbf{K}_{31} & \mathbf{K}_{32} & \mathbf{K}_{11} + \mathbf{K}_{22} + 2\mathbf{K}_{33} \end{array} \right] \left\{ \begin{array}{c} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{array} \right\} \\ & + \left[\begin{array}{ccc} \hat{\mathbf{K}}_{11} & \hat{\mathbf{K}}_{12} & \hat{\mathbf{K}}_{13} \\ \hat{\mathbf{K}}_{21} & \hat{\mathbf{K}}_{22} & \hat{\mathbf{K}}_{23} \\ \hat{\mathbf{K}}_{31} & \hat{\mathbf{K}}_{32} & \hat{\mathbf{K}}_{33} \end{array} \right] \left\{ \begin{array}{c} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{array} \right\} = \left\{ \begin{array}{c} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \mathbf{F}_3 \end{array} \right\} \end{aligned} \quad (4.3.17)$$

where \mathbf{M} , $\mathbf{C}(\mathbf{v})$, \mathbf{K}_{ij} , and \mathbf{F}_i are the same as those defined in Eq. (4.2.13), and

$$\hat{\mathbf{K}}_{ij} = \int_{\Omega^e} \gamma_e \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \quad (4.3.18)$$

Equation (4.3.17) can be expressed symbolically as

$$\bar{\mathbf{M}}\dot{\mathbf{v}} + [\bar{\mathbf{C}}(\mathbf{v}) + \bar{\mathbf{K}}^1(\mu) + \bar{\mathbf{K}}^2(\gamma)] \mathbf{v} = \bar{\mathbf{F}} \quad (4.3.19)$$

where $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}^T$. If the matrices indicated above are all constructed with the usual quadrature techniques, the penalty function equations are not solvable. The name for the method comes from the requirement that the penalty terms in $\bar{\mathbf{K}}^2$ be constructed with a reduced quadrature rule. Further details of the implementation are given in Section 4.5.3 and in [2,4,16].

4.3.4 Consistent Penalty Model

An alternative implementation of the penalty method is known in the literature as the consistent penalty method, and deals more directly with the finite element model than the variational statement. In this formulation, the pressure in the momentum Eq. (4.3.2) is replaced with [three-dimensional version of Eq. (4.3.15)]

$$P = -\gamma_e \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) = -\gamma_e \frac{\partial v_i}{\partial x_i} \quad (4.3.20)$$

and Eq. (4.3.1) is replaced with Eq. (4.3.20). The finite element model of Eq. (4.3.20) is given by

$$\left[\int_{\Omega^e} \Phi \Phi^T d\mathbf{x} \right] \mathbf{P} = - \left[\int_{\Omega^e} \gamma_e \Phi \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_i \quad (4.3.21)$$

or in terms of a matrix notation

$$\mathbf{M}_p \mathbf{P} = -\gamma_e \mathbf{Q}^T \mathbf{v} \quad (4.3.22)$$

or

$$\mathbf{P} = -\gamma_e \mathbf{M}_p^{-1} \mathbf{Q}^T \mathbf{v} \quad (4.3.23)$$

where $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}^T$, $\mathbf{Q}^T = (\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3)$, and \mathbf{Q}_i and \mathbf{M}_p are given by

$$\mathbf{Q}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x}; \quad \mathbf{M}_p = \int_{\Omega^e} \Phi \Phi^T d\mathbf{x} \quad (4.3.24)$$

When Eq. (4.3.23) is substituted for the pressure into the finite element model of the momentum equation (4.2.11), we obtain

$$\mathbf{M} \dot{\mathbf{v}} + (\mathbf{C}(\mathbf{v}) + \mathbf{K} + \mathbf{K}_p) \mathbf{v} = \mathbf{F} \quad (4.3.25)$$

with

$$\mathbf{K}_p = \gamma_e \mathbf{Q} \mathbf{M}_p^{-1} \mathbf{Q}^T \quad (4.3.26)$$

Equation (4.2.10) is not used in conjunction with Eq. (4.3.25) because the continuity equation is included in Eq. (4.3.25) through Eq. (4.3.23).

Equation (4.3.25) has the same general form as Eq. (4.3.19). The overall size of the penalty finite element model in Eq. (4.3.19) or (4.3.25) is reduced in comparison to the mixed finite element model in Eq. (4.2.15). To recover the pressure, the inverted form of (4.3.23) is used with a known velocity field. The particular form of the penalty finite element model described here is termed a *consistent penalty model* because it is derived from the consistent, mixed finite element approximation to the momentum and penalized continuity equations. This is in contrast to the reduced integration penalty (RIP) model described earlier, where the shape functions for the pressure do not enter the penalty matrix (4.3.18). Although the equations of the two models may be mathematically identical for certain choices of the pressure approximation and reduced integration, there are other differences that affect the numerical implementation. Numerical implementation of the consistent method described here relies on the ability to efficiently construct the \mathbf{K}_p matrix, i.e., invert \mathbf{M}_p at the element level. This requirement restricts the choices for the basis functions, Φ , used to represent the pressure [4,16–21]. Further comments on the basis functions will be given in Section 4.5.

4.4 Finite Element Models of Porous Flow

The derivation of the finite element model for the porous flow problems of Section 1.6 follows the development of previous sections in a completely analogous manner. For completeness, we present the mixed finite element model of the porous flow equations under isothermal conditions. The penalty finite element model of the equations can also be developed, following the discussion presented in Section 4.3.

The dependent variables are again approximated by the relations in Eqs. (4.2.6a,b). Substituting them into the weak statements of Eqs. (1.6.1) and (1.6.2) and omitting the temperature terms, results in the following equations:

Continuity

$$-\left[\int_{\Omega^e} \Phi \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_i = \mathbf{0} \quad (4.4.1)$$

Momentum

$$\begin{aligned} & \left[\int_{\Omega^e} \frac{\rho_0}{\phi} \Psi \Psi^T d\mathbf{x} \right] \dot{\mathbf{u}}_i + \left[\int_{\Omega^e} \frac{\rho_0 \hat{c}}{\sqrt{\kappa}} \Psi (\Psi^T \|\mathbf{v}\|) \Psi^T d\mathbf{x} \right] \mathbf{v}_i + \left[\int_{\Omega^e} \frac{\mu_e}{\kappa} \Psi \Psi^T d\mathbf{x} \right] \mathbf{v}_i \\ & + \left[\int_{\Omega^e} \mu_e \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \right] \mathbf{v}_i + \left[\int_{\Omega^e} \mu_e \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_j - \left[\int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x} \right] \mathbf{P} \\ & = \left[\int_{\Omega^e} \rho_0 f_i \Psi d\mathbf{x} \right] + \left\{ \oint_{\Gamma^e} \mathbf{T}_i \Psi ds \right\} \end{aligned} \quad (4.4.2)$$

where f_i denote the body force components. The acceleration tensor is assumed to be equal to $1/\phi$. Equations (4.4.1) and (4.4.2) can be written symbolically in the matrix form as

Continuity

$$-\tilde{\mathbf{Q}}^T \mathbf{v} = \mathbf{0} \quad (4.4.3)$$

Momentum

$$\tilde{\mathbf{M}} \dot{\mathbf{v}} + \tilde{\mathbf{C}}(\mathbf{u}) \mathbf{v} + \tilde{\mathbf{A}} \mathbf{v} + \tilde{\mathbf{K}} \mathbf{v} - \tilde{\mathbf{Q}} \mathbf{P} = \tilde{\mathbf{F}} \quad (4.4.4)$$

where $\mathbf{v} = \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}^T$ and

$$\tilde{\mathbf{M}} = \int_{\Omega^e} \frac{\rho_0}{\phi} \Psi \Psi^T d\mathbf{x}; \quad \tilde{\mathbf{C}}(\mathbf{v}) = \int_{\Omega^e} \frac{\rho_0 \hat{c}}{\sqrt{\kappa}} \Psi (\Psi^T \|\mathbf{v}\|) \Psi^T d\mathbf{x} \quad (4.4.5)$$

$$\tilde{\mathbf{A}} = \int_{\Omega^e} \frac{\mu}{\kappa} \Psi \Psi^T d\mathbf{x}; \quad \tilde{\mathbf{K}}_{ij} = \int_{\Omega^e} \mu_e \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \quad (4.4.6)$$

$$\tilde{\mathbf{Q}}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x}; \quad \tilde{\mathbf{F}}_i = \int_{\Omega^e} \rho_0 \Psi f_i d\mathbf{x} + \oint_{\Gamma^e} \Psi \mathbf{T}_i ds \quad (4.4.7)$$

where summation on repeated indices is implied. Equations (4.4.3) and (4.4.4) can be arranged into a single matrix equation of the same form as given previously for the Navier–Stokes equations [see Eq. (4.2.14)]:

$$\begin{bmatrix} \tilde{\mathbf{M}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{u}} \\ \dot{\mathbf{P}} \end{Bmatrix} + \begin{bmatrix} \tilde{\mathbf{C}}(\mathbf{u}) + \tilde{\mathbf{A}} + \tilde{\mathbf{K}} & -\tilde{\mathbf{Q}} \\ -\tilde{\mathbf{Q}}^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \end{Bmatrix} = \begin{Bmatrix} \tilde{\mathbf{F}} \\ \mathbf{0} \end{Bmatrix} \quad (4.4.8)$$

The above equation may take on a somewhat different structure when the various porous flow models are invoked. When the inertia terms are unimportant, then $\tilde{\mathbf{C}} = 0$. For the familiar Darcy flow case ($\hat{c} = 0, \mu_e = 0$), the above equation simplifies to

$$\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{u}} \\ \dot{\mathbf{P}} \end{Bmatrix} + \begin{bmatrix} \tilde{\mathbf{A}} & -\tilde{\mathbf{Q}} \\ -\tilde{\mathbf{Q}}^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \end{Bmatrix} = \begin{Bmatrix} \tilde{\mathbf{F}} \\ \mathbf{0} \end{Bmatrix} \quad (4.4.9)$$

The full porous model in (4.4.8) is very comparable to the Navier–Stokes equations of the previous section and shares many of the computational characteristics of the viscous flow problem. In general, these mixed and penalty finite element equations are stable and convergent with the proper choice of interpolation function. The reduced system in (4.4.9) is more questionable, since the Darcy formulation is really a potential flow and does not require a mixed finite element formulation. Notice that the two equations represented in (4.4.9) can be combined to form a single scalar equation for the pressure. That is, solving the first equation for \mathbf{v} (which is the Darcy equation)

$$\mathbf{v} = \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{Q}}\mathbf{P} + \tilde{\mathbf{A}}^{-1}\tilde{\mathbf{F}} \quad (4.4.10)$$

and substituting into the continuity (second) equation produces

$$\tilde{\mathbf{Q}}^T\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{Q}}\mathbf{P} = \tilde{\mathbf{Q}}^T\tilde{\mathbf{A}}^{-1}\tilde{\mathbf{F}} \quad (4.4.11)$$

which is a Poisson equation for the pressure. If the mixed method is used here, an inconsistency arises since the pressure shape function was of lower order than the velocity interpolation. A low order pressure solution from (4.4.11), when substituted into (4.4.10) would not be consistent with the generation of an accurate higher order velocity solution. Darcy type flows, when approached directly (i.e., not through reduction of a Brinkman type equation) are usually formulated as a continuum Poisson equation that resembles the heat conduction problem. The pressure would then be the primary variable to be interpolated and the resulting velocity field would be of lower order and consistent. The mixed method can be made to work with (4.4.9) though its convergence is suboptimal. Further details on this problem and a demonstration of the porous flow model can be found in [35]. This completes the development of the mixed finite element model of the porous flow equations.

4.5 Computational Considerations

4.5.1 Properties of the Matrix Equations

It is important to recognize some of the basic features of the matrix equations in (4.2.12), (4.3.17), and (4.4.8) because these characteristics will heavily influence the choice of a solution procedure for the various types of problems studied here. The given matrix equations represent the discrete analogs of the basic conservation equations with each term representing a particular physical process: \mathbf{M} represents the mass and \mathbf{C} represents the velocity dependent convective transport term. The viscous diffusion term is represented by \mathbf{K} , and \mathbf{Q} represents the pressure gradient operator and \mathbf{Q}^T is the divergence operator. The term \mathbf{F} contains body forces and surface forces.

An inspection of the structure of the individual matrices in (4.2.13) shows that \mathbf{M} and \mathbf{K} are symmetric, while \mathbf{C} is unsymmetric. This makes $\bar{\mathbf{K}}$ in Eq. (4.2.15a) unsymmetric. Thus, for general flows, the solution procedure must deal with an unsymmetric system. Conversely, when material properties are constant and flow velocities (e.g., Reynolds number) are sufficiently small so that a Stokes approximation is valid, the equations are symmetric and linear.

An additional feature of the mixed finite element model is the presence of zeroes on the matrix diagonals corresponding to the pressure [see Eq. (4.2.12)]. Equations (4.2.14), (4.3.17), (4.3.25), and Eq. (4.4.8) represent a set of ordinary differential equations in time. The fact that the pressure does not appear explicitly in the continuity equation imparts a number of characteristics to the discrete problem. For example, the lack of a time derivative for the pressure [see Eqs. (4.2.14) and (4.4.8)] makes the system time-singular in the pressure and precludes the use of purely explicit time-integration methods. The zero entries on the diagonal cause difficulties in the solution of linear equations resulting from (4.2.14) or (4.4.8). Direct equation solvers must use some type of pivoting strategy, while the use of iterative solvers is severely handicapped by poor convergence attributable mainly to the divergence-free condition. There are a number of other technical problems associated with the incompressibility condition that directly influence the computational problem. Questions regarding the impossibility of impulsively started flows, proper boundary conditions for derived equations (e.g., vorticity transport equation, pressure Poisson equation), and suitable initial conditions for time-dependent computations are all derived from the special form of the continuity equation. An explanation of many of these topics can be found in the works of Reddy [1–3,23], Oden [4], Oden and Carey [16], Gresho [36–38], Gresho and Sani [39] and others [17–21].

4.5.2 Choice of Interpolation Functions

In the previous sections we have presented finite element formulations whereby boundary value problems of interest are reduced to discrete systems of algebraic equations. In the present section, a discussion of the choice of interpolation functions used in various models is presented. We will limit our attention to the mixed and penalty finite element models. In a later section, some discussion of methods that are reformulated to reduce various computational difficulties will be provided.

Of central importance to the development of a finite element model is the choice of particular elements (i.e., interpolation functions) to be included in the element library for the models used. For the diffusion problem considered in Section 3.3, a wide variety of elements was presented. The finite element models of heat transfer as well as viscous incompressible flows (see Section 4.2.2) require only the C^0 -continuous functions to approximate the field variables. Thus, any of the Lagrange and serendipity family of interpolation functions are admissible for the interpolation of the velocity field and temperature in mixed and penalty finite element models.

The choice of interpolation functions used for the pressure variable in the mixed finite element model is further constrained by the special role the pressure plays in incompressible flows. Recall that the pressure can be interpreted as a Lagrange multiplier that serves to enforce the incompressibility constraint on the velocity field. From Eqs. (4.2.4) and (4.2.7) it is seen that the approximation function Φ used for pressure is the weighting function for the continuity equation. It has been shown, both numerically (see Reddy [3] and Taylor and Hood [40]) and theoretically (see Sani, et al. [25,26]) that, in order to prevent an overconstrained system of discrete equations, the interpolation used for pressure must be at least one order lower than that used for the velocity field. Further, pressure need not be made continuous across elements because the pressure is not a primary variable of the weak form presented in Eqs. (4.2.4) and (4.2.5).

Convergent finite element approximations of problems with constraints are governed by the ellipticity requirement and the *Ladyzhenskaya–Babuska–Brezzi (LBB) condition* (see Oden [4], Oden and Carey [16], and others [17–21]). The mixed and penalty finite elements used for viscous incompressible fluids must satisfy the LBB condition in order that they yield accurate, stable and convergent solutions. It is by no means a simple task to rigorously prove whether every new element developed for viscous incompressible flows satisfies the LBB condition. The discussion of the LBB theory and its application to various elements is beyond the scope of the present study. An extensive description of the LBB condition and its relation to practical incompressible finite elements can be found in the text by Gresho and Sani [39]. Before proceeding, we should note several points about finite element approximations that do not satisfy the LBB condition for mixed and penalty GFEM. First, while it is possible to solve mixed models with equal order velocity and pressure elements, the pressure solution will be highly oscillatory (checker-board). The velocity field from equal order elements will generally be accurate; the pressure field can be recovered from the oscillatory solution by a post-processing filter. Equal order elements that circumvent the LBB restrictions have been proposed and demonstrated. These techniques invariably lead to other types of weighted residual formulations, such as augmented Galerkin or “stabilized” methods (see, for example, Hughes, et al. [41], Tezduyar [42] and Franca, et al. [43]) or least-squares approaches (see Jiang and coworkers [6,44,45], Surana and Reddy and coworkers [7–14], and Bochev and Gunzburger [15]). Other approaches stabilize the pressure approximation directly by locally employing a projection method [46]. Some types of stabilization will be discussed briefly in later sections.

Several of the most standard finite elements for use in the mixed finite element models are selected here for discussion. Some of these elements are also valid (efficient) for use with the penalty method. The interpolation functions used for the velocity field, as well as various choices for the pressure approximation will be identified. Additional elements are catalogued and discussed in [39].

4.5.2.1 Quadrilateral elements (2-D)

One of the most commonly used elements for two-dimensional flows of viscous incompressible fluids is the nine-node rectangular element shown in Figure 4.5.1. The velocity components and any auxiliary variables that are present are approximated using the biquadratic Lagrange functions [also see Eq. (2.10.7)], which are given in Eq. (4.5.1). The functions are expressed in terms of the normalized coordinates for the element, (ξ, η) , which vary from -1 to $+1$.

$$\{\Psi^e\} = \frac{1}{4} \left\{ \begin{array}{l} (1 - \xi)(1 - \eta)(-\xi - \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ (1 + \xi)(1 - \eta)(\xi - \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ (1 + \xi)(1 + \eta)(\xi + \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ (1 - \xi)(1 + \eta)(-\xi + \eta - 1) + (1 - \xi^2)(1 - \eta^2) \\ 2(1 - \xi^2)(1 - \eta) - (1 - \xi^2)(1 - \eta^2) \\ 2(1 + \xi)(1 - \eta^2) - (1 - \xi^2)(1 - \eta^2) \\ 2(1 - \xi^2)(1 + \eta) - (1 - \xi^2)(1 - \eta^2) \\ 2(1 - \xi)(1 - \eta^2) - (1 - \xi^2)(1 - \eta^2) \\ 4(1 - \xi^2)(1 - \eta^2) \end{array} \right\} \quad (4.5.1)$$

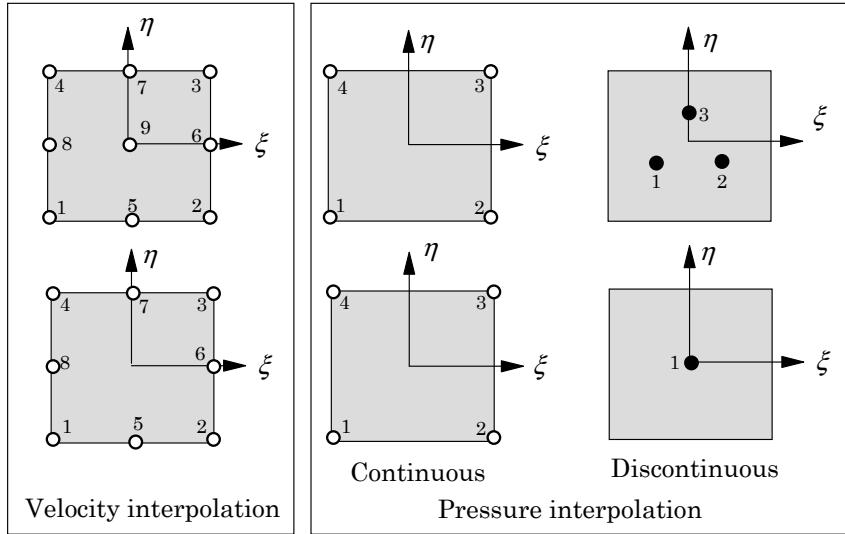


Figure 4.5.1: The quadrilateral elements used for the mixed model.

Several different pressure approximations are available when the velocities are approximated by quadratic Lagrange functions. The first is a continuous-bilinear approximation, in which the pressure is defined at the corner nodes of the element and is made continuous across element boundaries. The bilinear interpolation functions are defined in Eq. (4.5.2).

$$\{\Phi^e\} = \frac{1}{4} \begin{Bmatrix} (1-\xi)(1-\eta) \\ (1+\xi)(1-\eta) \\ (1+\xi)(1+\eta) \\ (1-\xi)(1+\eta) \end{Bmatrix} \quad (4.5.2)$$

With this combination of velocity and pressure interpolation the element is usually designated as a Q_2Q_1 element, where the first letter and subscript indicates a quadratic (second order) polynomial interpolation for each component of velocity on the quadrilateral and the second letter and subscript refers to the linear (first order) polynomial pressure approximation on the quadrilateral.

A second pressure approximation involves a discontinuous, linear variation defined on the element by

$$\Phi = \begin{Bmatrix} 1 \\ x \\ y \end{Bmatrix} \quad (4.5.3)$$

Here, the unknowns are not nodal point values of the pressure but correspond to the coefficients in $P = a + bx + cy$. In Eq. (4.5.3) the interpolation functions are written in terms of the global coordinates (x, y) for the problem; these functions could also be expressed in terms of the element coordinates. An element with the velocity approximation in Eq. (4.5.1) and the pressure approximation in Eq. (4.5.3) is labeled as Q_2P_{-1} element, where the P designation indicates a complete polynomial

on the element and the negative subscript signifies the function is discontinuous at element boundaries. Note that the P and Q designations differ in that P is a complete polynomial of degree $m = 1$ in the spatial coordinates and Q is a polynomial (not necessarily complete) of degree $n = 2$ in each spatial coordinate. The P designation is also used to label interpolations on simplex elements; Eq. (4.5.3) can be viewed as being defined on a triangle within the quadrilateral.

Two other pressure approximations also involve discontinuous functions on the element. If the pressure is defined at four points within the element and is not continuous at the element boundaries then a Q_2Q_{-1} element is produced. This element is not LBB compliant but has been used with some success. Also, a Q_2Q_0 element is possible where the pressure is a simple constant on each element. An eight-node quadratic velocity element may also be defined with some of the same types of pressure interpolation. The designation $Q_2^{(8)}$ is typically used to indicate the “serendipity” shape functions [see Eq. (2.10.8)]. Eight-node elements that have been used include the continuous pressure element $Q_2^{(8)}Q_1$, and the discontinuous pressure elements $Q_2^{(8)}Q_{-1}$ and $Q_2^{(8)}P_{-1}$. The most popular and most accurate of these two-dimensional quadrilaterals is the Q_2P_{-1} element which has become a standard for incompressible flow problems. Though some of the eight-node elements mentioned above are fairly good, they have been supplanted by the nine-node elements.

The elements shown in Figure 4.5.1 are known to satisfy the LBB condition and thus give reliable and convergent solutions for velocity and pressure fields (see Oden [4], Le Tallec and Ruas [19], and Chapelle and Bathe [20]). Other elements that are not LBB stable may yield acceptable solutions for some problems but are not always reliable for general applications. The unstable, equal order Q_1Q_1 and Q_2Q_2 elements will often produce usable velocity fields but with checkerboard pressure fields. A macro element constructed from a four patch of Q_1Q_1 elements is LBB stable but has found little use in practice. A widely used element is the Q_1P_0 element which employs a continuous bilinear velocity approximation and a discontinuous, constant on the element pressure field. This element is generally unstable with its LBB stability being mesh dependent [17,47,48]. However, it has found extensive use in practice, always returning good velocity accuracy and reasonable pressures after filtering. This element continues to attract the attention of mathematicians as it has proven a formidable challenge for stability and convergence proofs [17,39,48].

When the element interpolation functions are written in terms of the normalized local coordinates (ξ, η) , the relationship between the global coordinates (x, y) and the element coordinates (ξ, η) is defined by the transformation (see Sections 2.11 and 3.3)

$$x = \Upsilon^T \mathbf{x} ; \quad y = \Upsilon^T \mathbf{y} \quad (4.5.4)$$

where Υ is a vector of interpolation functions on the quadrilateral and the (\mathbf{x}, \mathbf{y}) are vectors of coordinates, which describe the geometry of the element. This transformation is quite general, and for the nine-node quadrilateral element it allows accurate representation of straight or curved boundaries of a domain. For an isoparametric formulation, we have $\Upsilon = \Psi$, where Ψ are the interpolation functions used to approximate the velocity field. When $\Upsilon = \Psi$ is quadratic, a quadratic interpolation of the element boundary is possible. In a subparametric formulation, we set $\Upsilon = \Phi$, and only a linear interpolation of the element boundary is possible.

4.5.2.2 Triangular elements (2-D)

With regard to two-dimensional triangular elements for mixed and penalty applications, the choices of velocity and pressure combinations are very similar to the quadrilaterals. The P_2P_1 and P_2P_{-1} elements correspond to their quadrilateral counterparts, with quadratic velocities and linear continuous and discontinuous pressures, respectively. The P_2P_1 element is LBB stable; the basic P_2P_{-1} element is not LBB stable. To achieve stability the P_2P_{-1} element must be enhanced to $P_2^+P_{-1}$, which indicates the addition of a cubic bubble function to the velocity approximation.

Lower order velocity triangles have also been tested with P_1P_0 being the simplest. This element is not LBB stable and is rarely used, unlike the similar quadrilateral. A useful, stable, low order triangle is $P_1^+P_1$ where again a cubic bubble is added to the velocity approximation. The isoparametric concept outlined for the quadrilaterals is equally applicable to the various triangular elements. Despite the ease with which triangular meshes can be created, quadrilateral elements generally seem to be preferred for viscous flow computations.

4.5.2.3 Hexahedral elements (3-D)

A standard element used in the analysis of three-dimensional viscous flow problems is the eight-node, hexahedron (brick) shown in Figure 4.5.2. The velocity components are approximated using the following trilinear Lagrange functions [also see Eq. (3.3.1)]:

$$\{\Psi^e\} = \frac{1}{8} \begin{Bmatrix} (1-\xi)(1-\eta)(1-\zeta) \\ (1+\xi)(1-\eta)(1-\zeta) \\ (1+\xi)(1+\eta)(1-\zeta) \\ (1-\xi)(1+\eta)(1-\zeta) \\ (1-\xi)(1-\eta)(1+\zeta) \\ (1+\xi)(1-\eta)(1+\zeta) \\ (1+\xi)(1+\eta)(1+\zeta) \\ (1-\xi)(1+\eta)(1+\zeta) \end{Bmatrix} \quad (4.5.5)$$

The functions are written in terms of the element natural coordinates (ξ, η, ζ) .

The pressure approximation for this element, being one order lower than Ψ , is a constant (and obviously discontinuous between elements),

$$\Phi = \{c\} \quad (4.5.6)$$

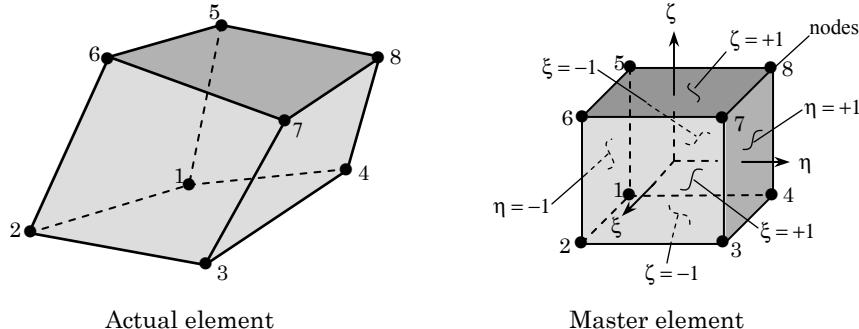


Figure 4.5.2: The eight-node brick element.

This element is denoted by Q_1Q_0 and is not LBB stable. Higher-order elements are possible and follow the construction pattern of the quadrilateral elements. Most higher-order velocity elements have not found widespread use because of the computational expense involved for an element with a large number of unknowns. This bias against higher order elements may change with the continued improvement in fast iterative matrix solvers.

The simplest LBB stable, higher order hexahedrons are the Q_2Q_1 and $Q_2^{20}Q_1$ elements, which are triquadratic in the velocity and continuous trilinear in the pressure. The Q_2Q_1 hex has 27 nodes for velocity and 8 nodes for pressure. The $Q_2^{20}Q_1$ element uses the “serendipity” functions for velocity that omit the mid-face and centroidal nodes from the Lagrange functions. The discontinuous pressure versions of these elements are denoted Q_2P_1 and $Q_2^{20}P_{-1}$ and are LBB stable. These elements have an element level mass balance unlike the continuous pressure versions.

Since the element shape functions are expressed in the normalized coordinates, the standard parametric transformation is needed

$$x = \boldsymbol{\Upsilon}^T \mathbf{x} ; \quad y = \boldsymbol{\Upsilon}^T \mathbf{y} ; \quad z = \boldsymbol{\Upsilon}^T \mathbf{z} \quad (4.5.7)$$

For an isoparametric formulation, we take $\boldsymbol{\Upsilon} = \boldsymbol{\Psi}$. Note that the eight-node brick element can only have straight edges unless a superparametric representation (i.e., $\boldsymbol{\Upsilon}$ is set to a triquadratic function) is employed. Superparametric descriptions are rarely used. For the higher order hexahedrons, the isoparametric formulation allows curved (quadratic) element edges.

4.5.2.4 Tetrahedral elements (3-D)

Tetrahedral elements for mixed and penalty formulations follow the two-dimensional triangular constructs for velocity and pressure combinations. The low order $P_1^+P_1$ element is LBB stable and adds a quartic bubble function to the linear velocity field. For higher order tetrahedrons, the continuous P_2P_1 element is LBB stable and the easiest to construct. The $P_2^+P_{-1}$ is also LBB stable with the quartic bubble. The isoparametric formulation is readily available for these elements.

Note that when heterogeneous meshes are constructed using hexahedrons and tetrahedrons, transition elements, such as prisms, must be used. Though easily constructed, LBB stability criteria for prisms is generally unknown.

4.5.3 Evaluation of Element Matrices in Penalty Models

The numerical evaluation of the coefficient matrices appearing in Eq. (4.3.18) for the RIP method requires special consideration. This aspect is discussed here by considering the Stokes model for the steady-state case.

For the steady case, Eq. (4.3.19) is of the form

$$(\mu \bar{\mathbf{K}}^1 + \gamma \bar{\mathbf{K}}^2) \{ \mathbf{v} \} = \{ \bar{\mathbf{F}} \} \quad (4.5.8)$$

where $\bar{\mathbf{K}}^1$ is the contribution from the viscous terms and $\bar{\mathbf{K}}^2$ is from the penalty terms, which come from the incompressibility constraint. In theory, as we increase the value of γ , the conservation of mass is satisfied more exactly. However, in practice, for some large value of γ , the contribution from the viscous terms would

be negligibly small compared to the penalty terms in the computer. Thus, if $\bar{\mathbf{K}}^2$ is a nonsingular (i.e., invertible) matrix, the solution of Eq. (4.3.21) for a large value of γ is trivial, $\{\mathbf{v}\} = \{\mathbf{0}\}$. While the solution satisfies the continuity equation, it does not satisfy the momentum equations. In this case the discrete problem (4.3.21) is said to be over-constrained or “locked.” If $\bar{\mathbf{K}}^2$ is singular, then the sum $(\mu\bar{\mathbf{K}}^1 + \gamma\bar{\mathbf{K}}^2)$ is nonsingular (because $\bar{\mathbf{K}}^1$ is nonsingular), and a nontrivial solution to the problem can be obtained.

The numerical problem described above is eliminated by proper evaluation of the integrals defining $\bar{\mathbf{K}}^1$ and $\bar{\mathbf{K}}^2$. It is found that if the coefficients of $\bar{\mathbf{K}}^2$ (i.e., penalty terms) are evaluated using a numerical integration (see Section 2.11) rule of an order less than that required to integrate them exactly, the finite element equations (4.3.21) give acceptable solutions for the velocity field. This technique of under-integrating the penalty terms is known in the literature as *reduced (order) integration*. For example, if a linear rectangular element is used to approximate the velocity field in a two-dimensional problem, the matrix coefficients $\bar{\mathbf{K}}^1$, as well as \mathbf{M} and $\mathbf{C}(\mathbf{v})$, are evaluated using the 2×2 Gauss quadrature, and $\bar{\mathbf{K}}^2$ are evaluated using the one-point (1×1) Gauss quadrature. The one-point quadrature yields a singular $\bar{\mathbf{K}}^2$. Therefore, Eq. (4.3.21) cannot be inverted, whereas $(\mu\bar{\mathbf{K}}^1 + \gamma\bar{\mathbf{K}}^2)$ is nonsingular and can be inverted (after assembly and imposition of boundary conditions) to obtain a good finite element solution of the original problem. When a quadratic rectangular element is used, the 3×3 Gauss quadrature is used to evaluate $\bar{\mathbf{K}}^1$, \mathbf{M} , and $\mathbf{C}(\mathbf{v})$, and the 2×2 Gauss quadrature is used to evaluate $\bar{\mathbf{K}}^2$. Similar comments apply to three-dimensional elements.

Unlike the RIP, the successful numerical implementation of the consistent penalty method described by Eq. (4.3.26) does not require special quadrature rules. However, the method does depend on the ability to efficiently invert \mathbf{M}_p at the element level during construction of the \mathbf{K}_p matrix. This requirement restricts the choices for the basis functions, Φ , used to represent the pressure [4–10]. If the pressure interpolation is continuous between elements then \mathbf{M}_p cannot be inverted at the element level; it is a global matrix whose inverse is a full matrix. When the pressure is approximated with a discontinuous interpolant, the \mathbf{M}_p can be inverted for each element and the rest of terms in \mathbf{K}_p easily constructed. Elements such as Q_2P_{-1} , Q_2Q_{-1} and Q_1Q_0 are candidates for use with a consistent penalty method.

The choice of the penalty parameter is largely dictated by the ratio of the magnitude of penalty terms to the viscous and convective terms (or compared to the Reynolds number, R_e), the mesh, and the word length of the computer. The following range of γ is used in computations

$$\gamma = 10^4 R_e \text{ to } \gamma = 10^{12} R_e \quad (4.5.9)$$

The upper bound is often a function of the word length in the computer.

4.5.4 Pressure Calculation

The pressure is a variable of some importance in flow problems and is usually required as part of a computational solution. In the penalty function methods this requires some post-processing operations. For the consistent penalty scheme, the pressure recovery is relatively straightforward and relies on the element level

equation given in (4.3.22). That is,

$$\mathbf{M}_p \mathbf{P} = -\gamma_e \mathbf{Q}^T \mathbf{v} \quad (4.5.10)$$

which can be solved for \mathbf{P} when the velocity field on the element is known. For the discontinuous interpolation functions used to construct \mathbf{M}_p , the solution to Eq. (4.5.10) yields a set of coefficients that allow the pressure to be evaluated at any point within the element. Typically, the pressure is evaluated at the nodes of the element and then averaged at common nodes between elements to arrive at a continuous pressure field.

The pressure recovery for the RIP method is more involved. The pressure should be computed, using Eq. (4.3.15) (or its three-dimensional equivalent), at quadrature points corresponding to the reduced integration rule. This is equivalent to using an interpolation for pressure that is one order less than the one used for the velocity field. However, the pressure computed using Eq. (4.3.15) at the reduced integration points is not always reliable and accurate. Pressures predicted using linear elements, especially for coarse meshes, are seldom acceptable. Quadratic elements are known to yield more reliable results and, in general, triangular elements perform poorly. Various techniques have been proposed in the literature to improve the accuracy of the recovered pressure fields (see [24,25,32,33]).

One alternative method for computing pressure is to use the pressure Poisson equation (obtained from the momentum equations)

$$\nabla^2 P = \nabla \cdot (\mathbf{f} + \nu \nabla^2 \mathbf{v} - \mathbf{v} \cdot \nabla \mathbf{v}) \quad (4.5.11)$$

Since Eq. (4.5.11) is similar to the heat conduction problem [with Q replaced by the right-hand side expression in Eq. (4.5.11)], it is straightforward to solve it using the finite element method. We will not discuss this method further.

Another alternative for pressure recovery was first proposed by Salonen and Aalto [49] and later used for different problems by Shiojima and Shimazaki [50]. By substituting the known velocity field and/or the shear stresses, the momentum equations can be expressed in terms of the pressure. These equations are solved using a least-squares finite element model (see [33]). Following the developments in [33], a derivation of the pressure computation scheme is presented here.

Since the velocity components v_i are known from the solution of Eq. (4.3.19), their substitution into Eq. (4.1.7) results in the residual,

$$\mathbf{R} \equiv M_1 \hat{\mathbf{e}}_1 + M_2 \hat{\mathbf{e}}_2 + M_3 \hat{\mathbf{e}}_3 \quad (4.5.12)$$

where M_i is the i -th component of the momentum equation and $\hat{\mathbf{e}}_i$ is the unit vector along the x_i coordinate direction:

$$M_i = \rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) - \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 f_i \quad (4.5.13)$$

The only unknown in Eq. (4.5.12) is the pressure P . We determine P such that

$$\mathbf{R} = \mathbf{0} \quad (4.5.14)$$

Equation (4.5.14) is a vector equation (i.e., there are three equations) in a single unknown, P . Hence, we require the square of the residual vector in Eq. (4.5.14) to be a minimum over an element with respect to P :

$$\frac{\partial}{\partial P} \left(\int_{\Omega^e} \mathbf{R} \cdot \mathbf{R} \, d\mathbf{x} \right) = 0 \quad (4.5.15)$$

This method is known as the *least-squares* method. Equation (4.5.15) provides the necessary weak statement for the determination of pressure:

$$\int_{\Omega^e} \mathbf{R} \cdot \frac{\partial \mathbf{R}}{\partial P} \, d\mathbf{x} = 0 \quad (4.5.16a)$$

or

$$\int_{\Omega^e} M_i \frac{\partial M_i}{\partial P} \, d\mathbf{x} = 0 \quad (4.5.16b)$$

Substituting $P = \sum_{i=1}^L P_i \phi_i$ into Eq. (4.5.16b), we obtain

$$\int_{\Omega^e} M_k \frac{\partial \phi_i}{\partial x_k} \, d\mathbf{x} = 0 \quad (4.5.17)$$

where in Eq. (4.5.17), and in the following equations, sum on $k = 1, 2, 3$ is implied. Equation (4.5.17) gives the i -th equation of the system of L equations for the nodal values (P_1, P_2, \dots, P_L). In matrix form, Eq. (4.5.17) can be expressed as

$$\mathbf{K}^* \mathbf{P} = \mathbf{F}^* \quad (4.5.18)$$

where

$$K_{ij}^* = \int_{\Omega^e} \frac{\partial \phi_i}{\partial x_k} \frac{\partial \phi_j}{\partial x_k} \, d\mathbf{x}, \quad F_i^* = \int_{\Omega^e} \frac{\partial \phi_i}{\partial x_k} M_k \, d\mathbf{x} \quad (4.5.19)$$

The right-hand side vector $\{\mathbf{F}^*\}$ consists of the contributions due to body forces, inertial forces, and viscous forces. The body force terms and inertial force terms are calculated at the nodes in the same manner as they were computed for velocity calculations. The viscous terms need some special treatment in their calculation. When linear elements are used, the viscous stresses are constant within the element and their derivatives vanish. The same problem arises in the finite element implementation of Eq. (4.5.11) because it involves second-order derivatives of the velocity field. To overcome this difficulty, the stresses are computed using the standard $2 \times 2 \times 2$ Gauss quadrature for the eight-noded (trilinear) brick elements. The stresses at the Gauss points are extrapolated to obtain the contribution of the stresses at the nodes. Such contributions from all surrounding elements are averaged to determine the nodal stresses. The main drawback of this method is that the stresses for the boundary nodes are not as accurate as those inside the domain. However, improvements are possible if the nodal stresses are calculated using an appropriate extrapolation scheme.

4.5.5 Traction Boundary Conditions

Equation (4.2.13) contains the contribution due to the boundary stresses applied to an element (sum on j)

$$\mathbf{F}_i^s = \oint_{\Gamma_e} \Psi \mathcal{T}_i ds = \oint_{\Gamma_e} \Psi \sigma_{ij} n_j ds \quad (4.5.20)$$

For example, for two-dimensional problems, Eq. (4.5.20) takes the explicit form

$$\mathbf{F}_1^s = \mathbf{F}_x^s = \oint_{\Gamma_e} \Psi \mathcal{T}_x ds = \oint_{\Gamma_e} \Psi \sigma_{xx} n_x ds + \oint_{\Gamma_e} \Psi \sigma_{xy} n_y ds \quad (4.5.21a)$$

$$\mathbf{F}_2^s = \mathbf{F}_y^s = \oint_{\Gamma_e} \Psi \mathcal{T}_y ds = \oint_{\Gamma_e} \Psi \sigma_{yx} n_x ds + \oint_{\Gamma_e} \Psi \sigma_{yy} n_y ds \quad (4.5.21b)$$

In general, the force vector for each equation contains contributions from both an applied normal stress and a shear stress. Boundary conditions could be specified directly in terms of the quantities shown in Eqs. (4.5.21a,b), i.e., the x and y components of the traction vector or the three independent components of the stress tensor. The pressure enters the calculation through the normal stress components:

$$\sigma_{xx} = -P + \tau_{xx}, \quad \sigma_{yy} = -P + \tau_{yy}, \quad \sigma_{zz} = -P + \tau_{zz}$$

As discussed in Section 3.5, the surface integrals shown in Eqs. (4.5.21a,b) can be evaluated in terms of the normalized coordinates for an element edge. Using the appropriate definition for the surface element ds

$$ds = |\mathbf{J}_s| d\xi_s = \left[\left(\frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{x} \right)^2 + \left(\frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{y} \right)^2 \right]^{\frac{1}{2}} d\xi_s = \Delta d\xi_s \quad (4.5.22a)$$

and the components of the normal

$$n_x = \frac{1}{\Delta} \frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{y} \quad ; \quad n_y = -\frac{1}{\Delta} \frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{x} \quad (4.5.22b)$$

the expressions in Eqs. (4.5.21a,b) can be written as

$$\mathbf{F}_x^s = \int_{-1}^{+1} \Psi \sigma_{xx} \frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{y} d\xi_s - \int_{-1}^{+1} \Psi \sigma_{xy} \frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{x} d\xi_s \quad (4.5.23a)$$

$$\mathbf{F}_y^s = \int_{-1}^{+1} \Psi \sigma_{yx} \frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{y} d\xi_s - \int_{-1}^{+1} \Psi \sigma_{yy} \frac{\partial \hat{\mathbf{Y}}^T}{\partial \xi_s} \mathbf{x} d\xi_s \quad (4.5.23b)$$

Recall that $\hat{\mathbf{Y}}$ is a vector of edge functions for the element, and they may be either linear or quadratic depending on the type of mapping used to describe the geometry of the element. The stress components σ_{ij} in Eq. (4.5.23) are specified along the element boundary. The above derivation may be repeated in a completely analogous manner for the three-dimensional case using the appropriate relations for the surface element ds .

It should be noted that the traction boundary conditions are written in terms of the *total* stress components. The total normal stress components contain pressure as a part. For example, the normal stress $\sigma_{11} = \sigma_{xx}$ is given by [see Eq. (4.1.3b)]

$$\sigma_{11} = -P + 2\mu \left(\frac{\partial v_1}{\partial x_1} \right) \quad (4.5.24)$$

In many practical cases, the viscous part of the normal stress is negligibly small and the normal stress is essentially equal to the pressure. When the viscous part is not negligible, the application of a normal stress boundary condition does not distinguish between contributions from the pressure and viscous parts but simply reflects their net effect.

The traction boundary conditions as written in Eqs. (4.5.23a,b) imply that the stress components σ_{ij} are known functions of position along the boundary. Though this is the most usual situation, other types of boundary conditions are possible. For example, a slip or “friction” boundary condition relates the shear stress to a change in velocity at the boundary [see Eq. (1.10.3)]. Along a boundary with $y = \text{constant}$, this condition is written as

$$\sigma_{xy} = \frac{1}{\alpha} (v_x - v_b) \quad (4.5.25)$$

where v_x is the fluid velocity at the wall, v_x^s is the velocity of the boundary, and α is the slip coefficient. If this condition (or its multidimensional generalization) is substituted into Eqs. (4.5.23a,b), then a boundary matrix and vector combination is produced since the unknown fluid velocity appears in the equation. This is analogous to the convective heat transfer boundary condition in Eq. (2.2.6) or (3.5.12), and is treated in the same manner. Other types of boundary conditions, such as the surface tension condition, may involve other variables, but the basic method of implementation is similar in all cases.

In the discussion presented above, we have only considered the case where the applied tractions or stress components are specified directly in the global (x, y, z) coordinate frame. In many situations of practical interest, the stresses will only be known as components normal and tangent to the boundary surface that is not a coordinate surface. It is a fairly straightforward process to rewrite Eqs. (4.5.23a,b) so that this case is properly implemented. The situation is slightly more complicated if velocity components normal and tangent to an arbitrary non-coordinate surface are to be specified. The standard procedure in this situation is to first transform the momentum equations for the affected boundary nodes to a locally defined coordinate system oriented along the normal and tangent directions. An orthogonal transformation matrix can be constructed to operate on the appropriate element matrix equations. After transformation, the normal and tangential momentum equations can be constrained by the known values of the normal and/or tangential velocity components. After the finite element solution is obtained, the velocity components in the global coordinate frame can be obtained from the normal and tangential components computed on the boundary through a simple definition of the form (for two dimensions)

$$v_x = v_n n_x - v_t n_y, \quad v_y = v_n n_y + v_t n_x \quad (4.5.26)$$

where (v_n, v_t) denote the normal and tangential components of velocity and (n_x, n_y) are the components of the unit normal vector to the boundary. For incompressible flows, an essential part of this implementation is the definition of the local normal and tangential directions such that a divergence-free velocity field is maintained. A complete discussion of this topic can be found in the paper by Engelman, et al. [51].

4.6 Solution of Nonlinear Equations

4.6.1 General Discussion

The assembled finite element equations associated with Eqs. (4.2.15a), (4.3.19), (4.3.25), and (4.4.8) can be used to represent a very large group of fluid mechanics problems. Unfortunately, the solution of the discrete equation set is notoriously difficult for all but the simplest flows. These numerical difficulties all have their origins in the physical description of the viscous, incompressible flow problem, i.e., the Navier–Stokes equations. The initial, boundary value problems of fluid dynamics are some of the most mathematically challenging in the physical sciences and translating them into a computational form does not simplify the problem. Viscous flows typically have a large spectrum of length and time scales, which in the computational setting implies high resolution requirements, large meshes and very large equation sets (matrices). The description of advective transport in an Eulerian reference frame produces highly nonlinear and an unsymmetric equation set. Small fluid viscosities and even modest velocities combine to form large Reynolds numbers with the physical result being thin boundary layers. Computationally, the high Reynolds number means that the discrete equations behave very differently in different regions of the domain. Though the equations are formally elliptic over the entire domain, flow regions away from boundaries take on a very hyperbolic character. Finally, the assumption of incompressibility influences not only the formulation of the problem (mixed/penalty method, LBB stable elements) but has a strong influence on the permissible solution methods for the discrete equations. The ubiquitous nature of the incompressibility constraint is one of the reasons that makes it difficult to catalog finite element solution methods for this class of problems. The form of the solution algorithm is dependent on the finite element formulation and the element choice, both of which may be altered in an attempt to satisfy incompressibility in an accurate and efficient computational algorithm.

In the following sections, the focus will be on solution methods for the equations resulting from the weak-form Galerkin finite element model (GFEM) using a mixed formulation or a penalty formulation. The GFEM has proven to be very robust and well-suited to difficult flow simulations. Fixed point iteration methods, which will be described shortly, work well with this type of formulation and its associated finite elements. The fully-coupled equation methods associated with the GFEM are computationally difficult with the most troublesome feature being the form and size of the matrix equation. The matrix forms that arise from the mixed GFEM are unsymmetric, sparse and positive semi-definite [see Eq. (4.2.12)]; systems of this form are termed *saddle-point* problems. Saddle-point problems are notoriously difficult and have historically dictated the use of direct solution methods (e.g. Gauss elimination) with the complications of pivoting or equation re-ordering strategies to avoid problems with the zero diagonal entries [see Eqs. (4.2.12) or (4.2.14)]. Large,

three-dimensional GFEM simulations are impractical for direct matrix solution methods because of the large memory requirements and general lack of extension to parallel computing architectures, though some progress has been realized for parallel direct solvers [52]. The use of “black-box” iterative matrix solvers (e.g., conjugate gradient) have not proven generally effective for these types of equations because standard preconditioners are often inadequate, the presence of the incompressibility constraint again being the major impediment. However, substantial progress has been made on iterative methods and preconditioners that are specially tailored to saddle-point problems and GFEM formulations, e.g. [53,54]. These innovations have made iterative methods an increasingly useful alternative to the direct methods, allowed continued use of fully coupled formulations and opened the way to parallel computing. Another approach to solving the saddle-point problem is to alter the matrix problem that is being solved. A subsequent section describes one alternative that relies on decoupling or segregating the discrete equations prior to solution. The incompressibility constraint again appears prominently in this scheme as it forces a reformulation of the equations. However, a significant benefit of this reformulation is the ability to use either direct or iterative methods on the resulting linear equations. A final section in this catalog of solution methods outlines the “stabilized” formulations that alter the Galerkin weighted residual form to circumvent the LBB condition and ultimately ease some computational difficulties. Notice that throughout the discussion of solution methods, the characteristics of the solution method plays a large role in determining its utility. Direct and iterative methods used to solve linear systems are discussed briefly in Appendix B; this topic encompasses a large area in computational mathematics and is outside the scope of this book.

Before proceeding to the discussion of specific algorithms, it is appropriate to outline the common mathematical features of a solution technique that are basic to many solution methods. The present discussion will consider methods used for the linearization and iterative solution of the fully-coupled, nonlinear equations arising from the use of the mixed or penalty finite element model for incompressible flows.

Using the symbolic format of Eq. (4.2.15), both mixed and penalty finite element models of this chapter can be written as

$$\bar{\mathbf{M}}\dot{\mathbf{U}} + \bar{\mathbf{K}}(\mathbf{U})\mathbf{U} = \bar{\mathbf{F}} \quad (4.6.1)$$

which represents a set of coupled, nonlinear, differential equations in time. For the time-dependent case, these equations are reduced to nonlinear algebraic equations by means of a time-approximation scheme, as described in Chapters 2 and 3. The resulting equations are of the form

$$\tilde{\mathbf{K}}(\mathbf{U}^*)\mathbf{U}^* = \tilde{\mathbf{F}} \quad (4.6.2)$$

which must be solved at each time step; the vector \mathbf{U}^* represents the solution variables at some discrete time in the integration interval. For time-independent problems, Eq. (4.6.1) is simplified to a form such as (4.6.2) by omitting the $\dot{\mathbf{U}}$ term; the \mathbf{U}^* in this case would represent the solution to the steady-state problem. Therefore, regardless of the type of problem under consideration, the solution process

for the discrete system must ultimately be concerned with a set of nonlinear algebraic equations.

For convenience, Eq. (4.6.2) can be expressed simply as

$$\mathcal{R}(\mathbf{U}) = \mathbf{0} \quad (4.6.3)$$

where it is understood that \mathcal{R} is in general a nonlinear operator and \mathbf{U} is the vector of unknowns. Many of the solution methods of practical interest for Eq. (4.6.3) can be regarded as variants of the method of successive approximation or functional iteration. Rewrite Eq. (4.6.3) as

$$\mathbf{U} = \mathcal{G}(\mathbf{U}) = \mathbf{U} - \mathcal{R}(\mathbf{U}) \quad (4.6.4)$$

which is usually termed a *fixed-point problem* with \mathbf{U} being the *fixed point* of the operator \mathcal{G} (see Reddy [2]). The form of (4.6.4) suggests the following iteration scheme, where the superscript indicates the iteration number,

$$\mathbf{U}^{n+1} = \mathcal{G}(\mathbf{U}^n) = \mathbf{U}^n - \mathcal{R}(\mathbf{U}^n) \quad (4.6.5)$$

which is often termed the Picard method (or the direct iteration method). The convergence of the scheme in Eq. (4.6.5) is intimately connected to the contraction mapping theorem (see Reddy [2] and Appendix C). In essence, successive substitution can be proven to converge if the operator \mathcal{G} has certain properties (i.e., \mathcal{G} is a *contraction mapping*) and the initial guess for \mathbf{U} is sufficiently “close” to the solution. These properties are relatively difficult to prove for the discrete operators associated with the finite element equations described in this chapter. The true utility of Eq. (4.6.5) is in its suggestion of an iterative solution procedure which can be generalized to the following form

$$\mathbf{U}^{n+1} = \mathcal{G}(\mathbf{U}^n) = \mathbf{U}^n - \alpha^n \mathbf{A}(\mathbf{U}^n) \mathcal{R}(\mathbf{U}^n) \quad (4.6.6a)$$

where α^n is a constant (that may vary with the iteration number) and $\mathbf{A}(\mathbf{U}^n)$ is a nonsingular matrix which is a function of \mathbf{U} . The method specified by Eq. (4.6.6a) provides a recursive method for finding the fixed point of \mathcal{G} , i.e., the solution of $\mathcal{R}(\mathbf{U}) = \mathbf{0}$. The success of the algorithm will depend critically on the choices for the parameter α and the matrix \mathbf{A} . Note that the product $\mathbf{A}(\mathbf{U}^n) \mathcal{R}(\mathbf{U}^n)$ can be interpreted as a correction vector, $\mathbf{D}(\mathbf{U}^n)$, such that

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \alpha^n \mathbf{A}(\mathbf{U}^n) \mathcal{R}(\mathbf{U}^n) = \mathbf{U}^n - \alpha^n \mathbf{D}(\mathbf{U}^n) \quad (4.6.6b)$$

where α is now the scalar magnitude of the correction.

The correction vector $\mathbf{D}(\mathbf{x}^n)$ is often related to a gradient of \mathcal{R} (or more often a gradient of $\mathcal{R}^T \mathcal{R}$; see Appendix C), and the resulting methods are termed *gradient methods*. If \mathbf{D} can be constructed recursively, then (4.6.6a) leads to a number of iterative methods, such as the *method of steepest descent* and the *conjugate gradient methods*. These methods are particularly attractive for finite element analysis because the solution of a large matrix system is avoided; in some sense these are “explicit” methods since a global matrix need not be constructed. The major

difficulty with these iterative schemes is the task of rationally selecting effective correction vectors. The correction vector \mathbf{D} may also be constructed by defining the matrix \mathbf{A} ; in this case the correction vector is generated from the solution of a matrix problem and the method is somewhat more “implicit” in nature. Methods in this family of algorithms include the *chord method* and *Newton’s method*.

4.6.2 Fully Coupled Solution Methods

The general discussion of iterative methods for the steady-state problem will now be focused on the finite element equations for incompressible flows as generated by the mixed or penalty forms of the method of weighted residuals. These Galerkin finite element models (GFEM) are particularly important since many finite element models for flow problems are variations of this basic formulation.

For general flow problems that are independent of time, the equation set of interest is derived from (4.6.1) by setting $\dot{\mathbf{U}} = 0$. That is,

$$\bar{\mathbf{K}}(\mathbf{U})\mathbf{U} = \bar{\mathbf{F}} \quad (4.6.7)$$

where \mathbf{U} is defined as a vector containing all of the unknowns of the problem, $\bar{\mathbf{K}}$ contains the advection and diffusion terms as well as the incompressibility constraint, and $\bar{\mathbf{F}}$ contains all of the applied boundary and volumetric forcing functions. The choice of an iterative method for the nonlinear, algebraic system in Eq. (4.6.7) is governed by several considerations. The selected method should be capable of providing converged solutions for a wide range of problems with minimal sensitivity to variations in boundary conditions, material properties, and initial data, i.e., the method should have a large radius of convergence (see Appendix C). The rate of convergence should also be reasonably high for reasons of computational economy; the computational work per iterative cycle should also be “small” to increase computational efficiency. Though it is reasonable to believe that there is a “best” algorithm for each combination of characteristics in the equation set, it is impractical to have all of these solution methods available in a single computer code. Rather, a few of the most generally applicable schemes are usually made available with the hope that the characteristics of any particular problem will fall within the bounds of these few techniques. In this section the main focus is placed on methods that have proven to be of value for the most general types of flow problems; no attention is given to methods that only perform well for special types of flows. Some of these methods will be familiar from the previous chapter as they are all generally applicable to systems of nonlinear equations. Also, it must be recognized that even though the algorithms described here are used routinely in both private and commercial software, the development of more effective methods is still an active area of research.

4.6.2.1 Picard method

A particularly simple method derived directly from the fixed point problem is the successive substitution, Picard, or functional iteration method. For the system in (4.6.7) this algorithm is given by

$$\bar{\mathbf{K}}(\mathbf{U}^n)\mathbf{U}^{n+1} = \bar{\mathbf{F}}(\mathbf{U}^n) \quad (4.6.8)$$

where the superscript indicates the iteration number. This algorithm has a reasonably large radius of convergence; the rate of convergence is generally fairly low since it is only a first-order method. An improvement in convergence rate can sometimes be obtained by use of a relaxation formula,

$$\bar{\mathbf{K}}(\mathbf{U}^n)\mathbf{U}^* = \bar{\mathbf{F}}(\mathbf{U}^n) \quad (4.6.9a)$$

where

$$\mathbf{U}^{n+1} = \alpha\mathbf{U}^n + (1 - \alpha)\mathbf{U}^* \quad 0 \leq \alpha \leq 1 \quad (4.6.9b)$$

Strongly nonlinear problems often exhibit an oscillatory convergence behavior for successive substitution, in which case the use of (4.6.9) would be of significant benefit.

Although the Picard schemes are generally slow to converge, they are often a good choice for the first several iterations in a general solution strategy. The wide radius of convergence allows the Picard method to start with a relatively poor initial solution guess, \mathbf{U}^0 , and move the solution closer to the true solution and within the radius of convergence of higher-order, more rapidly convergent methods. Note that the starting vector \mathbf{U}^0 for the Picard scheme, as well as most other iterative algorithms will, by necessity, be selected as a constant or zero vector; the first pass through the Picard method will produce a solution to the linear diffusion (Stokes flow) problem associated with Eq. (4.6.7). Neither of these solution vectors is particularly “close” to the solution of the nonlinear problem. The availability of a better initial estimate for \mathbf{U}^0 can dramatically improve the convergence behavior of the Picard and other solution methods.

4.6.2.2 Newton's method

In order to enhance the rate of convergence, a second-order method, such as Newton's method, must be considered. First rewrite Eq. (4.6.7) as

$$\mathbf{R}(\mathbf{U}) = \bar{\mathbf{K}}(\mathbf{U})\mathbf{U} - \bar{\mathbf{F}}(\mathbf{U}) = \mathbf{0} \quad (4.6.10)$$

Newton's method is based on a truncated Taylor's series expansion of $\mathbf{R}(\mathbf{U})$ about the known solution \mathbf{U}^n :

$$\mathbf{0} = \mathbf{R}(\mathbf{U}^n) + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}^n} \Delta \mathbf{U} + O(\Delta \mathbf{U})^2 \quad (4.6.11)$$

where $\Delta \mathbf{U} = (\mathbf{U}^{n+1} - \mathbf{U}^n)$. Omitting the terms of order two and higher, we obtain

$$\mathbf{R}(\mathbf{U}^n) = -\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}^n} (\mathbf{U}^{n+1} - \mathbf{U}^n) \equiv -\mathcal{J}(\mathbf{U}^n)(\mathbf{U}^{n+1} - \mathbf{U}^n) \quad (4.6.12)$$

where \mathcal{J} is the Jacobian matrix (also known as the *tangent matrix*)

$$\mathcal{J} = \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}^n} \quad (4.6.13)$$

Solving Eq. (4.6.12) for \mathbf{U}^{n+1} , we obtain

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \mathcal{J}^{-1}(\mathbf{U}^n)\mathbf{R}(\mathbf{U}^n) \quad (4.6.14)$$

Note the correspondence between Eq. (4.6.14) and the general procedure given in Eq. (4.6.6); the use of an α scale factor in Eq. (4.6.14) is also possible.

To illustrate some of the detail associated with Newton's method, consider the case of a two-dimensional mixed finite element model; the extension to three dimensions is immediate. In this example, the components of the solution vector $\mathbf{U}^T = [\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{P}^T]$ and the components of the vector \mathbf{R} are

$$\begin{aligned}\mathbf{R}_1 &= \mathbf{C}_1(\mathbf{v}_1)\mathbf{v}_1 + \mathbf{C}_2(\mathbf{v}_2)\mathbf{v}_1 + (2\mathbf{K}_{11} + \mathbf{K}_{22})\mathbf{v}_1 + \mathbf{K}_{12}\mathbf{v}_2 - \mathbf{Q}_1\mathbf{P} - \mathbf{F}_1 \\ \mathbf{R}_2 &= \mathbf{C}_1(\mathbf{v}_1)\mathbf{v}_2 + \mathbf{C}_2(\mathbf{v}_2)\mathbf{v}_2 + \mathbf{K}_{21}\mathbf{v}_1 + (\mathbf{K}_{11} + 2\mathbf{K}_{22})\mathbf{v}_2 - \mathbf{Q}_2\mathbf{P} - \mathbf{F}_2 \\ \mathbf{R}_3 &= -\mathbf{Q}_1^T\mathbf{v}_1 - \mathbf{Q}_2^T\mathbf{v}_2\end{aligned}\quad (4.6.15)$$

where (no sum on j)

$$\mathbf{C}_j(\mathbf{v}_j) = \int_{\Omega^e} \rho_0 \Psi(\Psi^T \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \quad (4.6.16)$$

and \mathbf{K}_{ij} are defined in Eq. (4.2.13). Equations in (4.6.15) correspond to the two momentum equations and the incompressibility constraint.

The key to Newton's method is the formation of the Jacobian matrix, which in this case is

$$\mathcal{J} = \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \right] = \begin{bmatrix} \frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_1}{\partial \mathbf{P}} \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{P}} \\ \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{P}} \end{bmatrix} \quad (4.6.17)$$

The components of \mathcal{J} can be computed directly from the definition in Eq. (4.6.13). We obtain the following nine Jacobian entries:

$$\begin{aligned}\frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_1} &= \mathbf{C}_1(\mathbf{v}_1) + \mathbf{C}_1(1)\mathbf{v}_1 + \mathbf{C}_2(\mathbf{v}_2) + 2\mathbf{K}_{11} + \mathbf{K}_{22} \\ \frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_2} &= \mathbf{C}_2(1)\mathbf{v}_1 + \mathbf{K}_{12}, \quad \frac{\partial \mathbf{R}_1}{\partial \mathbf{P}} = -\mathbf{Q}_1, \quad \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_1} = \mathbf{C}_1(1)\mathbf{v}_2 + \mathbf{K}_{21} \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_2} &= \mathbf{C}_1(\mathbf{v}_1) + \mathbf{C}_2(\mathbf{v}_2) + \mathbf{C}_2(1)\mathbf{v}_2 + \mathbf{K}_{11} + 2\mathbf{K}_{22} \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{P}} &= -\mathbf{Q}_2, \quad \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_1} = -\mathbf{Q}_1^T, \quad \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_2} = -\mathbf{Q}_2^T, \quad \frac{\partial \mathbf{R}_3}{\partial \mathbf{P}} = \mathbf{0}\end{aligned}\quad (4.6.18)$$

The components of \mathcal{J} are combinations of the basic element matrices found in the original equations plus a few new terms that arise due to differentiation of terms that are nonlinear in the dependent variables, e.g., the advection matrices. By studying the form of the matrices \mathbf{C}_j given in Eq. (4.6.18), it can be shown that the needed derivatives of these terms can be computed analytically.

Not all of the nonlinearities found in the components of Eq. (4.6.15) were explicitly accounted for in the derivatives defining the Jacobian matrix. For example, when the viscosity varies with velocity gradients, the \mathbf{K} terms are functions of \mathbf{v} and should appropriately be differentiated with respect to the velocity. The complexity of treating all possible material property and boundary condition variations in

this rigorous manner is usually prohibitive and generally not warranted. These nonlinearities are usually mild enough that they do not significantly affect the convergence of the Newton algorithm when treated in a “first-order” manner using a Picard or successive substitution procedure. Therefore, a strict Jacobian formulation is normally used only for the highly nonlinear advection terms.

The Newton scheme has a superior (quadratic) rate of convergence compared to the Picard scheme in Eq. (4.6.8). However, Newton’s method also has a somewhat smaller radius of convergence (i.e., is more sensitive to the initial solution vector \mathbf{U}^0), and is often used in conjunction with a Picard scheme.

4.6.2.3 Modified and quasi-Newton methods

The major drawback to both the successive substitution and Newton algorithms is the computational expense involved in the solution of a large matrix problem at each iteration. One method for avoiding this expense in the Newton scheme is to not compute a new Jacobian matrix at each iteration but instead work with a fixed iteration operator. This procedure, termed a *modified Newton method*, can be expressed as a variant of Eq. (4.6.14),

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \mathcal{J}^{-1}(\mathbf{U}^0)\mathbf{R}(\mathbf{U}^n) \quad (4.6.19)$$

After the first iteration cycle the method is very inexpensive since it only requires the reduction of the force vector and a back-substitution for each cycle. Unfortunately, the method has a poor rate of convergence and is therefore not recommended.

A method that combines the general efficiency of a modified Newton method and the approximate convergence rate of the full Newton method is the quasi-Newton procedure. Rather than working with a constant or fully updated Jacobian matrix, the quasi-Newton methods seek to approximately update the Jacobian matrix using a simple recursive procedure. The efficiency of the method comes from the ability to update the inverse of the Jacobian matrix, thus saving both the assembly and factorization time of the full method; some additional storage is required to save updates to the Jacobian matrix but this is not a serious disadvantage. A number of variants of the quasi-Newton procedure exist. The most successful scheme is based on the Broyden update [55] as implemented by Engelman, et al. [56]. This scheme may also be related to the general method in Eq. (4.6.6).

4.6.2.4 Continuation methods

A common failure of all iterative methods described previously is the lack of a sufficiently large radius of convergence. Given an initial estimate of the solution vector, the iterative method may not be able to reach a converged solution, since the initial guess was in some sense “too far away” from the required result. There are two general approaches to the problem of obtaining good initial estimates for a solution vector, and both of them involve some type of “tracking” procedure for the solution. The first procedure is simply the method of false transients in which the solution is followed through the use of a time parameter. The ideas associated with dynamic relaxation, which have found use in solid mechanics [57], could also be included with these techniques. Transient algorithms are covered in a following section (see also Section 3.7.3). A second method for circumventing the problems with initial solution vectors consists of incrementally approaching the final solution

through a series of intermediate solutions. These intermediate results may be of physical interest or may simply be a means to obtain the required solution. The formal algorithms used to implement such a procedure are termed *continuation (imbedding, incremental loading) methods*, and they can be used in conjunction with any of the previously described iterative procedures.

Continuation methods are similar in some respects to Newton's method and can be derived using a similar approach. Assume that the solution to Eq. (4.6.7) depends on a real parameter, λ (e.g., the Reynolds number or magnitude of a boundary condition). Then Eq. (4.6.10) can be written as

$$\mathbf{R}(\mathbf{U}, \lambda) = \bar{\mathbf{K}}(\mathbf{U}, \lambda)\mathbf{U} - \bar{\mathbf{F}}(\mathbf{U}, \lambda) = \mathbf{0} \quad (4.6.20)$$

As with Newton's method, if \mathbf{R} has continuous derivatives with respect to \mathbf{U} and is differentiable with respect to λ , then a truncated, two parameter, Taylor's series of $\mathbf{R}(\mathbf{U}, \lambda)$ can be used to produce the following algorithm:

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}, \lambda} \Delta \mathbf{U} = -\frac{\partial \mathbf{R}}{\partial \lambda} \Big|_{\mathbf{U}, \lambda} \Delta \lambda$$

or

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \mathcal{J}^{-1}(\mathbf{U}^n, \lambda^n) \frac{\partial \mathbf{R}}{\partial \lambda} \Big|_{\mathbf{U}^n, \lambda^n} \Delta \lambda \quad (4.6.21)$$

where $\Delta \lambda = \lambda^{n+1} - \lambda^n$ and the Jacobian matrix is the same as that defined for Newton's method.

The solution scheme in Eq. (4.6.21) amounts to a piecewise linearization of the solution curve with respect to λ ; higher order schemes could be generated. When $\Delta \lambda$ is small, it is probable that the new solution \mathbf{U}^{n+1} is very close to the true solution for λ^{n+1} . However, as $\Delta \lambda$ is made larger, the predicted solution \mathbf{U}^{n+1} will generally depart from the true solution. In such cases, the predicted value may be corrected via any applicable iterative method, e.g., Newton's method. Convergence for this inner iteration should be rapid since the predicted value is relatively close to the true solution.

The continuation procedure outlined here is a very powerful method and may be used to investigate bifurcation and stability characteristics of various flow problems. Extensive developments in this area have been reported by Winters and co-workers [58–61]. For problems at large Reynolds numbers, the method is a good choice. The biggest disadvantage of the method is that there is little guidance for the selection of $\Delta \lambda$ during the solution sequence. Unless specific intermediate solutions are of interest, the sequence of $\Delta \lambda$ should be made as large as possible to ensure minimum computational effort. However, the size of any given $\Delta \lambda$ is constrained by the radius of convergence requirements of the iterative method (e.g., Newton's method) used to provide the corrected solution at λ^{n+1} . In practice, adaptive continuation has been accomplished mainly through the use of heuristically based $\Delta \lambda$ selection procedures [62]. The theoretical basis for adaptive continuation methods remains an area of research [63,64], especially for applications in solid mechanics.

As a final comment on continuation methods, we note that the algorithm in (4.6.21) is a first-order continuation method. A simple and effective “zeroth-order”

continuation method has also found fairly widespread use in fluid mechanics and heat transfer problems. In this case, a series of problems are solved at increasing values of the continuation parameter λ . The converged solution at one value of λ^n is used as the starting solution vector at the next higher value of $\lambda^{n+1} = \lambda^n + \Delta\lambda$ in whatever iterative scheme has been selected. The formalism in Eq. (4.6.21) is not used to track the solution curve; some of the bifurcation tracking capabilities are lost in this simplified procedure.

4.6.3 Pressure Correction/Projection Methods

All of the iteration methods discussed earlier were based on the idea of a fixed point method and were designed to work with fully-coupled sets of equations, i.e., the velocity and pressure equations were all solved simultaneously. For each iteration method, the resulting linearized problem was structured such that a direct matrix solution procedure (e.g., Gauss elimination) could be effectively employed. Direct solution methods, however, are decreasingly effective as problem size increases due to excessive demands on computer resources. Most three-dimensional simulations are beyond the capabilities of a direct solution method. As noted previously, iterative matrix solutions have not proved viable with fixed point methods due to inadequate preconditioners for the positive, semi-definite equations. The requirement to consider larger and more complex flow simulations has forced reconsideration of the fully-coupled equation methods and opened the search for solution algorithms that are more efficient for large-scale problems.

Following ideas developed in the finite difference community, the troublesome incompressibility constraint is removed from the equation set and replaced by a decoupled Poisson equation for the pressure. These types of methods are generally termed pressure correction or pressure projection methods; finite difference methods of this type would be labeled as SIMPLE or SIMPLER methods [65]. Note that this technique is only possible with a mixed finite element formulation; the penalty method does not allow an explicit replacement of the continuity equation.

Consider a simplified component form of the steady, two-dimensional, finite element equations given in (Eq. 4.2.12)

$$K_{xx}v_x + K_{xy}v_y - Q_xP = F_x \quad (4.6.22a)$$

$$K_{yx}v_x + K_{yy}v_y - Q_yP = F_y \quad (4.6.22b)$$

$$-Q_x^T v_x - Q_y^T v_y = 0 \quad (4.6.22c)$$

In writing (4.6.22) it is assumed that the nonlinear advection terms have been absorbed in the K matrices. It is now possible to manipulate the equations in (4.6.22) to form a consistent, elliptic equation for the pressure. Symbolically, Eqs. (4.6.22a,b) are first solved for the velocity components v_x and v_y by inverting the appropriate K matrices. The resulting expressions for v_x and v_y are then substituted into Eq. (4.6.22c) which leads to

$$[Q_x^T K_{xx}^{-1} Q_x + Q_y^T K_{yy}^{-1} Q_y]P = -Q_x^T K_{xx}^{-1} f_x - Q_y^T K_{yy}^{-1} f_y \quad (4.6.23)$$

where $f_x = F_x - K_{xy}v_y$ and $f_y = F_y - K_{yx}v_x$. Equation (4.6.23) is a discretized form of the pressure Poisson equation that can be used in place of the incompressibility

constraint. The system given by Eqs. (4.6.22a,b) and (4.6.23) forms a decoupled system that could (in theory) be solved in an iterative cycle for the unknowns (v_x, v_y) and P . Note that (4.6.22a) and (4.6.22b) can be solved together as a single equation set or individually by component. The practical drawback to the use of (4.6.23) is the fact that the inverse K matrices make the pressure matrix impractical to construct; the inverse of a banded matrix is a dense matrix. Fortunately, it has been found by Haroutunian, et al. [66] and others that a reasonable approximation to the inverse K matrices are sufficient to produce a usable algorithm. In particular, if the diagonal of K is used to approximate K , then the inverse is trivial and the pressure matrix can be readily constructed.

There are a number of methods by which Eqs. (4.6.22a,b) and (4.6.23) can be employed to form a time-independent solution technique. We will only review one possible algorithm here and refer the interested reader to [38] and [66–68]. The pressure projection algorithm is defined by the following basic steps that are performed at each iteration until convergence is obtained. An approximation for the pressure is obtained from the pressure Poisson equation

$$[Q_x^T \tilde{K}_{xx}^{-1} Q_x + Q_y^T \tilde{K}_{yy}^{-1} Q_y] P^* = -Q_x^T \tilde{K}_{xx}^{-1} f_x(v_x^n) - Q_y^T \tilde{K}_{yy}^{-1} f_y(v_y^n) \quad (4.6.24)$$

where the f vectors and approximate \tilde{K} matrices are evaluated with velocities from the previous iteration n . Since P^* will generally be a poor approximation to P^{n+1} , a relaxation scheme is often applied to obtain a better estimate for pressure P

$$P^{n+1} = \alpha P^n + (1 - \alpha) P^* \quad 0 \leq \alpha \leq 1 \quad (4.6.25)$$

With a new value of the pressure (gradient) the momentum equations in (4.6.22a,b) can be solved for a velocity field. In component form then

$$K_{xx} v_x^* = F_x + Q_x P^{n+1} - K_{xy} v_y^* \quad (4.6.26)$$

$$K_{yy} v_y^* = F_y + Q_y P^{n+1} - K_{yx} v_x^* \quad (4.6.27)$$

or as a coupled system

$$\begin{bmatrix} K_{xx} & K_{xy} \\ K_{yx} & K_{yy} \end{bmatrix} \begin{Bmatrix} v_x^* \\ v_y^* \end{Bmatrix} = \begin{Bmatrix} F_x + Q_x P^{n+1} \\ F_y + Q_y P^{n+1} \end{Bmatrix} \quad (4.6.28)$$

Note that Eqs. (4.6.26)–(4.6.28) represent generally unsymmetric systems, and the left-hand side would be evaluated using v_x^n and v_y^n where appropriate. The velocities computed in Eqs. (4.6.26)–(4.6.28) are not divergence free. The last step of the algorithm is the projection step where the velocities are forced to satisfy the discretized continuity equation. By rederiving Eq. (4.6.23) via a slightly different approach, a second pressure Poisson equation provides

$$[Q_x^T \tilde{K}_{xx}^{-1} Q_x + Q_y^T \tilde{K}_{yy}^{-1} Q_y] P^\lambda = -Q_x^T v_x^* - Q_y^T v_y^* \quad (4.6.29)$$

The quantity P^λ is actually a Lagrange multiplier that is used to adjust the velocities with the following relations:

$$v_x^{n+1} = v_x^* + \tilde{K}_{xx}^{-1} Q_x P^\lambda \quad (4.6.30a)$$

$$v_y^{n+1} = v_y^* + \tilde{K}_{yy}^{-1} Q_y P^\lambda \quad (4.6.30b)$$

Equations (4.6.24)–(4.6.30) are used repetitively until convergence is attained. Other methods that use a pressure Poisson equation differ mainly in the sequence of steps and how the pressure solution is utilized. The pressure projection method outlined here generally has the best rate of convergence.

The above scheme can be implemented with a direct solution method (e.g., Gauss elimination) used at each step that requires such a solution. By segregating the equations, the matrix problem for each equation is significantly made smaller than the fully-coupled system and, therefore, much less of a computational burden. Three-dimensional FEM problems can be processed with this algorithm. However, the most important feature of the solution algorithm is that each matrix that requires solution is of a form that is readily amenable to solution via an iterative solver, such as the conjugate gradient method. Through the application of iterative methods, the computational problem is further reduced in size as well as computational effort and hence larger problems can be analyzed. Also, note that the segregated iterative algorithm is also applicable to fully discretized time-dependent problems.

4.7 Time-Approximation Schemes

4.7.1 Preliminary Comments

Here we consider time approximation of the matrix differential equations of the type in (4.6.1):

$$\bar{\mathbf{M}}\dot{\mathbf{U}} + \bar{\mathbf{K}}(\mathbf{U})\mathbf{U} = \bar{\mathbf{F}} \quad (4.7.1)$$

Equation (4.7.1) represents an approximation to the original system of partial differential equations, which is discrete in space and continuous in time. A direct time integration procedure replaces the continuous time derivative with an approximation for the history of the dependent variables over a small portion of the problem time scale. The result is an incremental procedure that advances the solution by discrete steps in time. In constructing such a procedure, questions of numerical stability and accuracy must be considered.

Though explicit integration methods have been used (see [69]) in the solution of (4.7.1), preference is usually given to implicit procedures for this class of problems. The explicit methods are plagued by several difficulties, including: (1) the natural implicitness of the pressure in an incompressible flow, (2) severe time step restrictions needed to maintain stability of the integration process, (3) the problems of diagonalizing and inverting $\bar{\mathbf{M}}$ in a cost-effective manner for a variety of element types, and (4) reductions in accuracy due to the diagonalization of $\bar{\mathbf{M}}$. Implicit integration methods, though computationally expensive, are desirable due to their increased stability and the consistent treatment of the pressure.

In the following sections two implicit procedures and one explicit method will be described (see [69–71]). Both of the implicit algorithms make use of a predictor/corrector cycle to improve efficiency and accuracy; both may be used with either a fixed time step or a dynamic time step selection algorithm. The solution of the resulting nonlinear, algebraic system for each time plane is normally obtained by Newton's method, though most of the iterative methods of the previous section would also be applicable. All of these integration methods are direct extensions of the algorithms (e.g., the Crank–Nicolson method) discussed for the heat conduction problem in Chapters 2 and 3.

4.7.2 Forward/Backward Euler Schemes

A first-order integration method that is useful for the equations in (4.6.1) employs a forward Euler scheme as a predictor with the backward Euler method functioning as the corrector step. Omitting the details of the derivation, the application of the explicit, forward Euler formula to Eq. (4.6.1) yields

$$\bar{\mathbf{M}}\mathbf{U}_p^{n+1} = \bar{\mathbf{M}}\mathbf{U}^n + \Delta t_n [\bar{\mathbf{F}}(\mathbf{U}^n) - \bar{\mathbf{K}}(\mathbf{U}^n)\mathbf{U}^n] \quad (4.7.2)$$

This can be written in a form that is more suitable for computation by replacing the bracketed term with a rearranged form of Eq. (4.7.1)

$$\mathbf{U}_p^{n+1} = \mathbf{U}^n + \Delta t_n \dot{\mathbf{U}}^n \quad (4.7.3)$$

In Eqs. (4.7.2) and (4.7.3) the superscript on a vector indicates the time plane, the subscript p denotes a predicted value, and $\Delta t_n = t_{n+1} - t_n$ is the time step. By using the form shown in (4.7.3) a matrix inversion of $\bar{\mathbf{M}}$ is avoided; the “acceleration” vector $\dot{\mathbf{U}}^n$ is computed from a form of the corrector formula as shown below. Note that since the forward Euler scheme is explicit, it is applied only to the velocity and any auxiliary variables contained in the \mathbf{U} vector; the pressure, being implicit, is not predicted with this formula.

The corrector step of the first-order scheme is provided by the backward Euler (or fully implicit) method. When applied to Eq. (4.7.1) this implicit method yields

$$\bar{\mathbf{M}}\mathbf{U}^{n+1} = \bar{\mathbf{M}}\mathbf{U}^n + \Delta t_n [\bar{\mathbf{F}}(\mathbf{U}^{n+1}) - \bar{\mathbf{K}}(\mathbf{U}^{n+1})\mathbf{U}^{n+1}] \quad (4.7.4)$$

or in a form more suitable for computation

$$\left[\frac{1}{\Delta t_n} \bar{\mathbf{M}} + \bar{\mathbf{K}}(\mathbf{U}^{n+1}) \right] \mathbf{U}^{n+1} = \frac{1}{\Delta t_n} \bar{\mathbf{M}}\mathbf{U}^n + \bar{\mathbf{F}}(\mathbf{U}^{n+1}) \quad (4.7.5)$$

The implicit nature of this method is evident from the form of (4.7.5). Note that the pressure is included in the vector \mathbf{U} in Eq. (4.7.5).

With the discretization of the time derivative, the equation in (4.7.5) now represents a set of nonlinear algebraic equations for the solution vector \mathbf{U} at time t_{n+1} . This is precisely the same type of problem that was encountered in the time-independent case (see Section 4.6) and for which a number of iterative algorithms are available. In practice, Newton’s method is generally used for Eq. (4.7.4) due to its rapid convergence and the obvious need to limit the number of computations that are repeated at each time plane. A segregated iteration method would be useful for large simulations though its convergence rate is very modest. As indicated previously, the rate of convergence of Newton’s method is greatly increased if the initial estimate of the vector \mathbf{U} is “close” to the true solution. The solution \mathbf{U}_p , predicted from the explicit formula (4.7.3) provides this initial guess for the iterative procedure in a cost-effective manner. It should be noted that the use of the predictor step is not mandatory and the backward Euler scheme could be used alone to integrate the equation set. In such a situation, the first estimate of the solution at $n + 1$ would be obtained from the solution at the previous time plane and a number of iterations with Newton’s method would be required to converge the solution.

4.7.3 Adams–Bashforth/Trapezoid Rule

An integration method that is second-order in time can be developed along the same lines as described above. A second-order equivalent to the forward Euler method is the variable step, Adams–Bashforth predictor given by

$$\mathbf{U}_p^{n+1} = \mathbf{U}^n + \frac{\Delta t_n}{2} \left[\left(2 + \frac{\Delta t_n}{\Delta t_{n-1}} \right) \dot{\mathbf{U}}^n - \left(\frac{\Delta t_n}{\Delta t_{n-1}} \right) \dot{\mathbf{U}}^{n-1} \right] \quad (4.7.6)$$

where $\Delta t_n = t_{n+1} - t_n$ and $\Delta t_{n-1} = t_n - t_{n-1}$. This formula can be used to predict the solution vector (excluding the pressure) given two “acceleration” vectors from previous time planes; no matrix solution is required.

A compatible corrector formula for use with Eq. (4.7.6) is available in the form of the trapezoid rule. When applied to Eq. (4.7.1) the trapezoid rule yields the scheme,

$$\left[\frac{2}{\Delta t_n} \bar{\mathbf{M}} + \bar{\mathbf{K}}(\mathbf{U}^{n+1}) \right] \mathbf{U}^{n+1} = \frac{2}{\Delta t_n} \bar{\mathbf{M}} \mathbf{U}^n + \bar{\mathbf{M}} \dot{\mathbf{U}}^n + \bar{\mathbf{F}}(\mathbf{U}^{n+1}) \quad (4.7.7)$$

Equation (4.7.7) is again observed to be a nonlinear, algebraic system for the vector \mathbf{U}^{n+1} and can be solved using an iterative procedure such as Newton’s method. All of the comments regarding the forward/backward Euler method are also applicable here.

4.7.4 Implicit Integration and Time Step Control

The actual integration process using either of the above two methods can be outlined in a few simple steps that follow closely the heat conduction procedure of Chapters 2 and 3. At the beginning of each time step it is assumed that all of the required solution and “acceleration” vectors are known and the time increment for the next step has been selected. To advance the solution from time t_n to time t_{n+1} then requires the following steps (see Section 3.7):

1. A tentative solution vector, \mathbf{U}_p^{n+1} , is computed using the predictor Eqs. (4.7.3) or (4.7.6). The pressure variables are not included in this prediction.
2. The corrector equations (4.7.5) or (4.7.7) are solved for the “true” solution, \mathbf{U}^{n+1} . This involves the iterative solution of (4.7.5) or (4.7.7) via Newton’s method. The predicted values \mathbf{U}_p^{n+1} are used to initialize the Jacobian matrix for the Newton iteration.
3. The “acceleration” vectors are updated using the new solution \mathbf{U}^{n+1} and the “inverted” forms of the corrector formulas. For the first-order method the acceleration is computed from the backward Euler definition

$$\dot{\mathbf{U}}^{n+1} = \frac{1}{\Delta t_n} (\mathbf{U}^{n+1} - \mathbf{U}^n) \quad (4.7.8)$$

while the second-order accelerations are derived from the trapezoid rule

$$\dot{\mathbf{U}}^{n+1} = \frac{2}{\Delta t_n} (\mathbf{U}^{n+1} - \mathbf{U}^n) - \dot{\mathbf{U}}^n \quad (4.7.9)$$

4. A new integration time step is computed. The time step selection process is based on an analysis of the time truncation errors in the predictor and corrector formulas as described below. If a constant time step is being used, this step is omitted.
5. Return to step 1 for next time increment.

In actual implementation the Newton iteration process in step 2 is not usually carried to absolute convergence. Rather, a one-step Newton correction is employed as advocated in [71]. This procedure is quite efficient and can be very accurate provided the time step is suitably controlled. Some implementation improvements in this algorithm have recently been documented in [72].

The adaptive control of the time step is an added benefit in the use of a predictor/corrector method. The time step estimation formula for the above methods is

$$\Delta t_{n+1} = \Delta t_n \left(b \cdot \frac{\epsilon^t}{d_{n+1}} \right)^m \quad (4.7.10)$$

where $m = 1/2$, $b = 2$ for the first-order method and $m = 1/3$, $b = 3(1 + \Delta t_{n-1}/\Delta t_n)$ for the second-order scheme. The user specified error tolerance for the integration process is ϵ^t , which typically has a value of 0.001. The quantity d_{n+1} is an appropriate norm on the integration error, which is defined as the difference between the predicted solution and the corrected value. Typically, the following norms are used for each velocity variable,

$$d_{n+1}^V = \frac{1}{\sqrt{N} V_{max}} \left[\sum_{i=1}^N \left(V_i^{(n+1)} - V_{ip}^{(n+1)} \right)^2 \right]^{\frac{1}{2}} \quad (4.7.11)$$

where N is the number of velocity nodes in the problem, V_{max} is a constant velocity scale and V_i and V_{ip} are the corrected and predicted velocities at the nodes. The norm defined in (4.7.11) is used separately for each velocity component in the problem. During the integration process, new time steps are computed using each of the velocity norms; the resulting time steps are compared and the smallest is used for the next integration step. This procedure allows the dominant momentum equation to control the integration algorithm.

4.7.5 Explicit Integration

The primary difficulty in constructing an explicit integration method was first mentioned in Section 4.5.1, and relates to the inherently implicit nature of the pressure. From Eq. (4.2.15), it is clear that the lack of a coefficient matrix for \hat{P} makes the explicit advancement of this variable impossible. As in the case of the pressure correction methods in Section 4.6.3, this difficulty can be partially circumvented by replacing the standard continuity equation with a pressure Poisson equation. With this substitution, the velocity components can be advanced through time via an explicit integrator; the pressure at each time step must be computed from the implicit Poisson equation.

The explicit procedure that is outlined here was proposed by Gresho, et al. [69] and follows the general philosophy of the previously described pressure projection

method. Consider the simplified, segregated form of the two-dimensional, time-dependent, finite element equations given by Eq. (4.2.11)

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{K}(\mathbf{v})\mathbf{v} - \mathbf{Q}\mathbf{P} = \mathbf{F} \quad (4.7.12)$$

$$-\mathbf{Q}^T\mathbf{v} = \mathbf{0} \quad (4.7.13)$$

where the \mathbf{K} Matrix contains both the diffusive and advective terms and \mathbf{v} contains both velocity components. To construct a Poisson equation for the pressure, operate on (4.7.12) first with \mathbf{M}^{-1} and then with \mathbf{Q}^T ; noting from (4.7.13) that $\mathbf{Q}^T\mathbf{v} = \mathbf{0}$ allows the modified equation (4.7.12) to be written as

$$(\mathbf{Q}^T\mathbf{M}^{-1}\mathbf{Q})\mathbf{P} = \mathbf{Q}^T\mathbf{M}^{-1}(\mathbf{F} - \mathbf{K}(\mathbf{v})\mathbf{v}) \quad (4.7.14)$$

which is an elliptic equation for the pressure P . As in Section 4.6 the key to obtaining a viable algorithm is the construction of \mathbf{M}^{-1} . If \mathbf{M} is diagonalized, then \mathbf{M}^{-1} is diagonal and the Poisson equation is readily available. Note also that the left-hand side of Eq. (4.7.14) is a constant matrix and needs only to be formed and triangularized once.

The general explicit scheme described by Gresho, et al. [69] utilized a forward Euler integration method for Eq. (4.7.12) with a pressure solution obtained from (4.7.14). Various numerical procedures were used in conjunction with the basic method to produce a reliable, cost-effective method. Low order (trilinear) velocity elements were used so that \mathbf{M} could be diagonalized; one-point Gauss quadrature was used to speed up construction of element matrices. So-called “balancing tensor diffusivity” was used to offset the time truncation error associated with the forward Euler scheme. Also, subcycling was introduced for Eq. (4.7.14) in which a new pressure was computed only after several explicit velocity time steps. Though an explicit time integration method can be derived as shown here, its practical implementation is quite complex. The method is only conditionally stable and is still burdened with the solution of at least one matrix problem. More recent work on explicit methods with pressure projection can be found in [70,73].

4.8 Stabilized Methods

4.8.1 Preliminary Comments

Computational difficulties and perceived shortcomings of the Galerkin finite element (GFEM) model have lead to investigation and development of alternate finite element models. The objective in each of these alternate models has been to avoid either or both of the two main “problems” encountered with mixed and penalty finite element models. Whether or not real problems exist with the GFEM depends on analysis objectives and what can reasonably be expected from a numerical simulation of a complex physical phenomena.

The first problem area for the standard GFEM is the occurrence of “wiggles” in the velocity field. Typically, this is a node-to-node oscillation of the velocity components emanating from a boundary or some other flow feature that produces a large velocity gradient. The wiggle problem is most pronounced at high Reynolds numbers and/or on coarse computational meshes. The inability of the finite element

mesh to resolve the steep gradient results in an imbalance between the advective and diffusive terms in the equation. The weighted residual formulation, in trying to do a best fit to the solution, produces a field that oscillates about the true solution.

Historically, the wiggle problem was blamed on the centered difference approximation generated by the Galerkin method and its inability to account for an “upwind” (one-sided) representation of the advective term. A long history of research and development has lead to formulations that avoid wiggle problems and produce smooth solutions on virtually any mesh. Some of the early developers of this approach include Heinrich, et al. [74,75] and Christie, et al. [76]. The latter methods are attributed to Hughes and coworkers [77-79] and are usually termed streamline-upwind/Petrov–Galerkin (SUPG) methods. These were the first stabilized methods though the stabilization terminology is more recent than the method. The arguments for wiggle stabilization are that (a) upwinding mimics the physical reality of advection, (b) smooth solutions can be obtained inexpensively on coarse meshes, (c) difficult high Reynolds number problems can be stabilized and are therefore computationally tractable, and (d) stabilization is a consistent formulation (i.e., exact solutions of the Navier–Stokes equations satisfy the stabilized equations). Opponents of wiggle suppression schemes, such as SUPG, argue that (a) it is not needed because wiggles are just a mesh problem that can be alleviated by proper mesh design or mesh adaptivity, (b) nonphysical damping is always added to the equations, lowering the physical Reynolds number, and (c) it is not certain what boundary value problem is being solved when the added stabilization terms are present. A more extensive discussion of the wiggle problem can be found in Gresho and Sani [39].

The second area of concern with the GFEM has a number of facets and revolves around the ubiquitous incompressibility constraint. The saddle-point problem generated by the mixed GFEM formulation makes the problem difficult to solve, especially in three dimensions. As noted previously, direct solvers with pivoting are well suited to the mixed formulation but too costly for large-scale applications. Iterative solvers are becoming more useful and robust but must still be specially designed for the problem area. The LBB stability condition associated with the GFEM limits element choice with respect to computationally convenient choices for the velocity and pressure interpolation. A number of attempts have been made to circumvent the LBB condition and produce a more efficient method. The pressure projection algorithm of a previous section is an illustration of one such method. The pressure-stabilizing/Petrov–Galerkin (PSPG) method [80] is another approach to avoiding the LBB requirement and is in the same spirit as the SUPG. A third alternative is the polynomial pressure projection method (PPPS) of Dohrmann and Bochev [46] that does not rely on an equation residual.

Some of the stabilized methods noted above were somewhat ad hoc in nature, incremental in their development and refinement, and lacked mathematical rigor and physical meaning. The introduction and development of the Variational Multiscale method (VMS) by Hughes and colleagues [81,82] has remedied much of the confusion regarding stabilized procedures and provided much needed explanation and consistency of implementation. This general approach has the additional benefit that it leads to a natural and promising implementation of subgrid turbulence models, see e.g., [83,84].

In the following section, a general stabilization method, namely, the Galerkin/Least-squares (GLS) method, is outlined. This formulation contains forms of both the SUPG and the PSPG. The PPS method for dealing with equal-order velocity and pressure elements is also outlined and in a subsequent section the VMS method, as developed for stabilization applications, is described and related to the SUPG and PSPG methods.

4.8.2 Galerkin/Least-Squares Formulation

One method for enhancing the stability of the mixed or GFEM for incompressible flows involves the addition of various least-squares terms to the original Galerkin statement. The Galerkin/Least-squares approach is sometimes termed a residual method because the added least-squares terms are weighted residuals of the momentum equation; this form of the least-squares term implies the consistency of the method since the momentum residual is employed. The GLS is also known as a perturbation method since the added terms can be viewed as perturbations to the weighting functions. The fact that the weight and test functions are different also makes GLS a Petrov-Galerkin method. Development and popularization of the GLS methods for flow problems is primarily due to Hughes, Tezduyar and coworkers [41–43,80,85,86] and follows as a generalization of their work on SUPG and PSPG methods.

The usual approach to the GLS begins with the discontinuous in time Galerkin method and considers a finite element approximation for a space-time slab. Continuous polynomial interpolation is used for the spatial variation while a discontinuous time function is used within the space-time slab. The assumed time representation obviates the need to independently consider time integration methods. To simplify the present discussion of the GLS, the time independent form of the incompressible flow problem is considered which avoids the complexity of the space-time finite element formulation [42]. Using vector notation and following the weak form development of Section 4.2, the GLS variational form for the momentum and continuity equations can be written as

$$\begin{aligned} & \int_{\Omega} \mathbf{w} \cdot (\rho_0 \mathbf{v} \cdot \nabla \mathbf{v}) d\mathbf{x} + \int_{\Omega} \nabla \mathbf{w} \cdot (-\mathbf{P} + 2\mu \mathbf{D}(\mathbf{u})) d\mathbf{x} - \int_{\Omega} \rho_0 \mathbf{w} \mathbf{f} d\mathbf{x} \\ & + \int_{\Omega} Q \nabla \cdot \mathbf{v} d\mathbf{x} + \sum_{n=1}^{nel} \int_{\Omega_n} R_{GLS} d\mathbf{x} = - \int_{\Gamma} \mathbf{w} T_i ds \end{aligned} \quad (4.8.1)$$

where \mathbf{w} and Q are the weight functions for the momentum and continuity equations and the element residual is defined by

$$R_{GLS} = (\delta + \epsilon + \beta) \cdot (\rho_0 \mathbf{v} \cdot \nabla \mathbf{v} + \nabla P - 2\mu \mathbf{D}(\mathbf{u})) \quad (4.8.2)$$

with

$$\delta = \tau_{supg} \mathbf{v} \cdot \nabla \mathbf{w} \quad (4.8.3)$$

$$\epsilon = \tau_{pspg} \frac{1}{\rho_0} \nabla Q \quad (4.8.4)$$

$$\beta = -\tau_{gls} 2\mu \mathbf{D}(\mathbf{w}) \quad (4.8.5)$$

The τ coefficients are weighting parameters that will be discussed shortly. If the definitions for δ, ϵ and β are substituted into (4.8.2) and the various weighting parameters are equivalent to a single parameter τ , the result is

$$R_{GLS} = \tau [\rho_0 \mathbf{v} \cdot \nabla \mathbf{w} + \nabla Q - 2\mu \mathbf{D}(\mathbf{w})] \cdot [\rho_0 \mathbf{v} \cdot \nabla \mathbf{v} + \nabla P - 2\mu \mathbf{D}(\mathbf{u})] \quad (4.8.6)$$

$$R_{GLS} = \mathcal{L}(\mathbf{w}, Q) \tau \mathcal{L}(\mathbf{v}, P) \quad (4.8.7)$$

which are the standard definitions for the residual contribution to the GLS formulation. In Eq. (4.8.7) \mathcal{L} is the incompressible differential operator. The splitting of the first residual into three separate contributions in (4.8.6) was done to allow the original SUPG and PSPG formulations to be easily recovered. If β and ϵ are set to zero, the stabilized SUPG method is recovered; if β and δ are set to zero, the PSPG method is recovered. Setting only β to zero achieves both SUPG and PSPG stabilization. It is apparent from the form in (4.8.6) that the GLS is a more general formulation than the original wiggle and pressure stabilization methods.

Note that in Eq. (4.8.1) the first four integrals and the right-hand side define a standard Galerkin weighted residual method that is written in terms of global shape functions; the integrals are over the problem domain, Ω . The added residual term in (4.8.6) is defined over the interior of each element and basically contains the square of all or parts of the momentum residual. The various τ parameters are positive coefficients that have the dimension of time. The forms of these parameters are usually developed from error estimates, convergence proofs and/or dimensional analysis. Particular constants within each parameter are selected by optimizing the method on simple problems and generalizing to multidimensions. It should be noted that the development of the τ parameters is not a unique process. For the steady problems considered here a typical τ for the GLS method is

$$\tau = \left[\left(\frac{2\|\mathbf{v}\|}{h} \right)^2 + \left(\frac{4\nu}{h^2} \right)^2 \right]^{-1/2} \quad (4.8.8)$$

where h is an appropriate element length. For the SUPG and PSPG formulations, the τ values are a function of an element Reynolds number and the ratio of an element length to a velocity scale. Details on these parameters may be found in [42] and [87].

The error and convergence analysis for the GLS and PSPG have demonstrated that equal-order velocity and pressure interpolation for these methods is viable and the LBB condition is not a requirement [41,85]. In the finite element implementation, convenient elements, such as the bilinear Q_1/Q_1 , linear P_1/P_1 , biquadratic Q_2/Q_2 and quadratic P_2/P_2 , are now useful approximations while being unstable in the GFEM context. The low-order elements in this group have an advantage over their higher-order counterparts because some of the stabilization terms associated with viscous diffusion are identically zero. This considerably simplifies the element equation building process.

To complete this brief discussion of stabilized methods based on residuals, the finite element equations arising from the GLS formulation will be shown in matrix form. Using the notation from Section 4.2 the momentum equation is

$$[\mathbf{C}(\mathbf{v}) + \mathbf{C}_\delta(\mathbf{v}) + \mathbf{C}_\beta(\mathbf{v})] \mathbf{v} + [\mathbf{K} + \mathbf{K}_\delta + \mathbf{K}_\beta] \mathbf{v} - [\mathbf{Q} + \mathbf{Q}_\delta + \mathbf{Q}_\beta] \mathbf{P} = \mathbf{F} \quad (4.8.9)$$

and the continuity equation is

$$-\mathbf{Q}^T \mathbf{v} - \mathbf{Q}_\epsilon \mathbf{P} + \mathbf{C}_\epsilon(\mathbf{v})\mathbf{v} + \mathbf{K}_\epsilon \mathbf{v} = \mathbf{0} \quad (4.8.10)$$

The subscripted terms all belong to the various types of stabilization. If any τ parameter is set to zero the corresponding term is omitted and the Galerkin form can be recovered by setting all τ parameters to zero. Finally, note that since all of the τ parameters depend on the element size, in the limit as the element size gets small, the stabilized method reduces to the Galerkin form. It is only on coarse meshes, relative to the problem of interest, that stabilization plays a significant role in the method.

4.8.3 Polynomial Pressure Projection

The pressure stabilization portion of the GLS method alters the variational form of the continuity equation to

$$\int_{\Omega} Q \nabla \cdot \mathbf{v} d\mathbf{x} + \sum_{n=1}^{nel} \int_{\Omega_n} \tau_{pspg} \frac{1}{\rho_0} \nabla Q \mathcal{L}(\mathbf{v}, P) d\mathbf{x} = 0 \quad (4.8.11)$$

where τ_{pspg} is often defined by

$$\tau_{pspg} = \frac{h}{2\|\mathbf{v}\|} z(Re) \quad (4.8.12)$$

where h is the element length and $z(Re)$ is a function of element type and Reynolds number [42]. As before $\mathcal{L}(\mathbf{v}, P)$ is the incompressible differential operator and the residual of the momentum equation. The addition of the element level term in (4.8.11) allows equal-order pressure and velocity approximations but at the cost of computing residuals on each element and the construction of several addition matrices. The global matrix form corresponding to continuity can then be written as shown in (4.8.10), where the addition stabilization terms are readily identified and recognized as assemblies of the element level contributions from (4.8.11).

A less expensive and non-residual based pressure stabilization is available in the polynomial pressure projection stabilization (PPPS) method due to Dohrmann and Bochev [46]. This method is especially noteworthy in that it contains no adjustable parameters that must be developed for various element types. The variational form of the continuity equation for the PPPS method is

$$\int_{\Omega} Q \nabla \cdot \mathbf{v} d\mathbf{x} + \int_{\Omega} \frac{1}{\mu} (P - \Pi P)(Q - \Pi Q) d\mathbf{x} = 0 \quad (4.8.14)$$

where Π is an element level projection operator and μ is an average element viscosity. When finite element approximations are employed with (4.8.14) the resulting matrix form for continuity is

$$-\mathbf{Q}^T \mathbf{v} - \mathbf{Q}_\Pi \mathbf{P} = 0 \quad (4.8.15)$$

The stabilization matrix \mathbf{Q}_Π is obtained by assembly of element level matrices given by

$$\mathbf{Q}_{\Pi_e} = (\mathbf{M}_e - \mathbf{E}_e^T \mathbf{D}_e^{-1} \mathbf{E}_e) / \mu_e \quad (4.8.16)$$

and

$$\mathbf{M}_e = \int_{\Omega_e} \Phi \Phi^T d\mathbf{x}; \quad \mathbf{E}_e = \int_{\Omega_e} \Upsilon \Phi^T d\mathbf{x}; \quad \mathbf{D}_e = \int_{\Omega_e} \Upsilon \Upsilon^T d\mathbf{x} \quad (4.8.17)$$

The vector Φ is the standard pressure basis function for the element and Υ is a vector of low-order polynomials such as $\Upsilon^T = \{1, x, y\}$ for two-dimensional elements and $\Upsilon^T = \{1, x, y, z\}$ for three dimensions. Note the lack of adjustable parameters. Details of the derivation of (4.8.16) can be found in [46]. Like the PSPG method, the PPPS avoids the LBB condition and allows equal-order elements to be used with very good results [46,88] and reduced computational effort. The GFEM momentum equation is unchanged with either the PSPG or PPPS approach, as the stabilization focus is on the pressure equation. Both methods provide a type of pressure mass matrix to the system, thus eliminating the zero (diagonal) matrix in the assembled system. In addition to (equal-order) stabilization, the saddle-point problem is reduced in severity and matrix solver performance is enhanced.

4.8.4 Variational Multiscale Methods

The various stabilization methods embedded in the GLS formulation require the definition of various τ parameters for which little theory previously existed to guide development. The continual evolution and refinement of the stabilization parameters was a source of some criticism and uncertainty in the method. The Variational Multiscale method (VMS), as introduced by Hughes, et al. [81,82], has provided a clear explanation of the origins of the stabilization parameters in residual based methods. Here we will outline the VMS methodology and show its relation to stabilization; a later section will outline the use of VMS for turbulent flow formulations.

Mechanics problems in general and fluid mechanics in particular contain a wide spectrum of length and time scales. The basic idea of the VMS method is to additively decompose the macroscopic flow problem into a series of problems, each of which represents an appropriate length and time scale for the problem of interest. When cast in a computational (finite element) framework, some of the problem scales will be resolvable on the computational mesh and some will need to be solved and/or modeled on a smaller or subgrid scale. It is the use and specification of the subgrid quantities that determine how the VMS is categorized and why there is more than one VMS method. For some applications the subgrid quantities provide a method for stabilizing the basic computational procedure. The implementation of physical models, such as turbulence, is also conveniently incorporated through a VMS approach. Finally, the scale separation paradigm can be used for coupling various types of physical processes such as continuum and non-continuum behavior [89,90]. In the following we will be concerned with the first of these options.

To simplify the initial sketch of the VMS method, let \mathcal{L} be a differential operator on the region Ω (with boundary Γ) with the associated boundary value problem

$$\mathcal{L}u = f \quad \text{on } \Omega \quad (4.8.18a)$$

$$u = g \quad \text{on } \Gamma \quad (4.8.18b)$$

The variational form of (4.8.18) is

$$a(w, u) = (w, \mathcal{L}u) = (w, f) \quad (4.8.19)$$

where $(*, *)$ is the usual inner product for the relevant function spaces, w is the weighting function and $a(w, u)$ is a bilinear form. For the current application \mathcal{L} will be the linearized Navier–Stokes equations.

To solve (4.8.18) using the VMS method, assume that $u = \bar{u} + u'$ describes the additive decomposition of the dependent variable and likewise $w = \bar{w} + w'$ describes the weighting function. The bar quantities are associated with the larger or coarse scales of the problem and the prime quantities are the smaller or fine scales. The bar and prime quantities must be linearly independent and have the following properties

$$\bar{u} = g \quad \text{on } \Gamma \quad (4.8.20a)$$

$$u' = 0 \quad \text{on } \Gamma \quad (4.8.20b)$$

Substituting the u and w definitions into the variational statement produces

$$a(\bar{w} + w', \bar{u} + u') = (\bar{w} + w', \mathcal{L}(\bar{u} + u')) = (\bar{w} + w', f) \quad (4.8.21)$$

By using the linear independence of w , the following two problems are produced for the coarse and fine scales.

Coarse Scale

$$(\bar{w}, \mathcal{L}\bar{u}) + (\bar{w}, \mathcal{L}u') = (\bar{w}, f) \quad (4.8.22a)$$

or

$$(\bar{w}, \mathcal{L}\bar{u}) + (\mathcal{L}^*\bar{w}, u') = (\bar{w}, f) \quad (4.8.22b)$$

Fine Scale

$$(w', \mathcal{L}u') + (w', \mathcal{L}\bar{u}) = (w', f) \quad (4.8.23a)$$

or

$$(w', \mathcal{L}u') = (w', f) - (w', \mathcal{L}\bar{u}) = (w', f - \mathcal{L}\bar{u}) = (w', R_{\bar{u}}) \quad (4.8.23b)$$

where \mathcal{L}^* is the adjoint operator and $R_{\bar{u}}$ is the residual of the coarse scale equation.

In the coarse scale equation, the use of the adjoint allows the influence of the fine scales to be explicitly incorporated into the coarse scale equation. Similarly, in the fine scale equation (4.8.23b), it can be seen that the coarse scale residual (or error) drives the fine scale equation. It is the basic objective of the VMS method to derive an analytic solution for the fine scale in terms of the coarse scale, i.e., solve (4.8.23b) to produce $u' = \mathcal{F}(\bar{u})$. This fine scale solution, when substituted into the coarse scale equation (4.8.22b) provides a single equation for the coarse scales but with a proper accounting of the fine scale phenomena. In a computational procedure, the coarse scales are identified with the mesh resolvable solution and the fine scales with the subgrid quantities. Note that it is of course possible to consider more than two scales in the decomposition.

Use of a Green's function allows the fine scale solution to be written formally as

$$u' = - \int g'(\mathcal{L}\bar{u} - f) d\Omega = -M' R_{\bar{u}} = -M'(\mathcal{L}\bar{u} - f) \quad (4.8.24)$$

where g' is the Green's function associated with the fine scale adjoint. This fine scale solution could be substituted into the coarse scale equation,

$$(\bar{w}, \mathcal{L}\bar{u}) - (\mathcal{L}^*\bar{w}, M'(\mathcal{L}\bar{u} - f)) = (\bar{w}, f) \quad (4.8.25)$$

which must then be solved for \bar{u} .

This outline has considered only smooth, continuous functions for both scales. When used in a finite element framework, details of the method change, but the overall procedure and objectives remain the same. Generally, the fine scale solution will only be obtained approximately via a numerical method. The overall algorithm may have a single coarse scale equation as described here or be a two step (equation) procedure. For some VMS applications the fine scale yields to a simple approximation and its addition to the coarse scale provides a stabilizing influence and/or a redistribution of the residual. This is the essence of the GLS, SUPG and PSPG stabilization methods. Note the similarity of the fine scale term in Eq. (4.8.25) and the stabilizing residual term in the GLS formulation (4.8.7). The VMS approach provides a clear method for deriving and describing the τ parameters essential to previous stabilization methods.

To complete this discussion, the incompressible, viscous flow problem will be developed using a VMS approach. The starting point is the variational form of the equations which were given previously as

$$\begin{aligned} \int_{\Omega} \left[\rho_0 w_i \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) + \frac{\partial w_i}{\partial x_j} (-P \delta_{ij}) + \frac{\partial w_i}{\partial x_j} \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] d\mathbf{x} \\ = \int_{\Omega} \rho_0 w_i f_i d\mathbf{x} + \oint_{\Gamma} w_i T_i ds \end{aligned} \quad (4.8.26)$$

$$- \int_{\Omega^e} Q \frac{\partial v_i}{\partial x_i} d\mathbf{x} = 0 \quad (4.8.27)$$

where the usual integration-by-parts has been used to transform the stress divergence term in the momentum equation. The weighting functions for momentum and continuity are w_i and Q , respectively. To proceed with the VMS approach, assume an additive decomposition of the solution fields and the weighting functions into coarse (large) and fine (small) scales as

$$\begin{aligned} v_i(x_i, t) &= \bar{v}_i(x_i, t) + v'_i(x_i, t) \\ P(x_i, t) &= \bar{P}(x_i, t) + P'(x_i, t) \end{aligned} \quad (4.8.28)$$

$$\begin{aligned} w_i(x_i) &= \bar{w}_i(x_i) + w'_i(x_i) \\ Q(x_i) &= \bar{Q}(x_i) + Q'(x_i) \end{aligned} \quad (4.8.29)$$

where the bar denotes the coarse scale and the prime refers to the fine scale. The weight functions are not functions of time which is a standard assumption. This decomposition is usually introduced through a projector [91] from the infinite dimensional space of the continuous problem to the finite dimensional space of the approximate problem. Note that an important characteristic of the scale separation is the linear independence of the coarse and fine scale functions.

Substituting (4.8.28) and (4.8.29) into the variational form in (4.8.26) and (4.8.27) yields the following coupled equations.

Coarse Scale

$$\begin{aligned} & \int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial \bar{v}_i}{\partial t} + v_j^* \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & + \int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial v'_i}{\partial t} + v_j^* \frac{\partial v'_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} (-P' \delta_{ij}) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & = \int_{\Omega} \rho_0 \bar{w}_i \bar{f}_i d\mathbf{x} + \int_{\Omega} \rho_0 \bar{w}_i f'_i d\mathbf{x} + \oint_{\Gamma} \bar{w}_i \bar{T}_i ds + \oint_{\Gamma} \bar{w}_i T'_i ds \end{aligned} \quad (4.8.30)$$

$$- \int_{\Omega} \bar{Q} \frac{\partial \bar{v}_i}{\partial x_i} d\mathbf{x} - \int_{\Omega} \bar{Q} \frac{\partial v'_i}{\partial x_i} d\mathbf{x} = 0 \quad (4.8.31)$$

Fine Scale

$$\begin{aligned} & \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial v'_i}{\partial t} + v_j^* \frac{\partial v'_i}{\partial x_j} \right) + \frac{\partial w'_i}{\partial x_j} (-P' \delta_{ij}) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & + \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial \bar{v}_i}{\partial t} + v_j^* \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial w'_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & = \int_{\Omega} \rho_0 w'_i \bar{f}_i d\mathbf{x} + \int_{\Omega} \rho_0 w'_i f'_i d\mathbf{x} + \oint_{\Gamma} w'_i T'_i ds + \oint_{\Gamma} w'_i \bar{T}_i ds \end{aligned} \quad (4.8.32)$$

$$- \int_{\Omega} Q' \frac{\partial v'_i}{\partial x_i} d\mathbf{x} - \int_{\Omega} Q' \frac{\partial \bar{v}_i}{\partial x_i} d\mathbf{x} = 0 \quad (4.8.33)$$

At this point, the only assumption that has been made in writing Eqs. (4.8.30)–(4.8.33) is that the nonlinear advection term in the momentum equation can be linearized with a velocity v_j^* .

To proceed with the stabilizing computational form of the VMS method, a number of assumptions are made. As the finite element method is the computational method of interest, the assumptions will be stated in terms of this approach. First, the fine scale is localized to the element with $v'_i = 0$ on the boundary of each element; likewise the fine scale weighting is $w'_i = 0$ on the element boundary. This is the localization of the appropriate Green's function for the fine scale. The fine scale pressure P' , and weighting Q' are normally neglected; this is not mandatory and has been explored by Codina [92]. This assumption is used here to simplify the fine scale equations. The fine scale localization allows the boundary terms for the fine scale to be neglected. The fine scale body force f'_i is neglected for this isothermal case. Finally, the time dependence of the fine scale is generally neglected for stabilization applications and this simplification is also followed here. Using these assumptions the coarse and fine scale equations may be rewritten as follows.

Coarse Scale

$$\begin{aligned} & \int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial \bar{v}_i}{\partial t} + v_j^* \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & + \int_{\Omega} \left[\rho_0 \bar{w}_i \left(v_j^* \frac{\partial v'_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & = \int_{\Omega} \rho_0 \bar{w}_i \bar{f}_i d\mathbf{x} + \oint_{\Gamma} \bar{w}_i \bar{T}_i ds \end{aligned} \quad (4.8.34)$$

$$-\int_{\Omega} \bar{Q} \frac{\partial \bar{v}_i}{\partial x_i} d\mathbf{x} - \int_{\Omega} \bar{Q} \frac{\partial v'_i}{\partial x_i} d\mathbf{x} = 0 \quad (4.8.35)$$

Fine Scale

$$\begin{aligned} & \int_{\Omega} \left[\rho_0 w'_i \left(v_j^* \frac{\partial v'_i}{\partial x_j} \right) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & + \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial \bar{v}_i}{\partial t} + v_j^* \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial w'_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & = \int_{\Omega} \rho_0 w'_i \bar{f}_i d\mathbf{x} + \oint_{\Gamma} w'_i \bar{T}_i ds \end{aligned} \quad (4.8.36)$$

The first four terms and the right-hand side terms of the coarse scale Eq. (4.8.34) are the standard Galerkin terms for the linearized Navier–Stokes equations; the first term of (4.8.35) is also a FEM term. The fifth and sixth terms in the second integral of Eq. (4.8.34) and the second term in the continuity equation (4.8.35) provide the influence of the fine scales on the coarse scale solution. The first four terms in the fine scale Eq. (4.8.36) are a weighted residual statement for the fine scale variables and the remaining terms in the second integral represent the projection of the coarse scale residual. A fine scale continuity equation is not required due to the assumption that $Q' = 0$.

The next step is to solve the fine scale Eq. (4.8.36) for the v'_i variables. A variety of methods have been proposed and demonstrated for the solution of the fine scale and associated multiscale problems. Early work employed polynomial bubble functions [93] to represent the fine scale variables on each element. Later, other methods, such as residual free bubbles [94], subgrid scale methods [95], hierarchical basis function methods [96] and a two level method [97] were studied. The references cited here are a very limited sample of the large literature on VMS methodologies and the interested reader should consult [91] for additional references. Here we will first outline the bubble function method as it shows the close tie between the VMS and the previous residual-based stabilization methods.

Let the fine scale velocity and weighting functions on each element be represented by polynomial bubble functions where

$$v'_i = \mathbf{\Gamma}^T \mathbf{v}_i^b ; \quad w'_i = \mathbf{\Lambda}^T \mathbf{b}_i \quad (4.8.37)$$

and \mathbf{v}_i^b and \mathbf{b}_i are the bubble function coefficients. Assume that the coarse scale residual is constant on each element. Using (4.8.37) in the fine scale equation and recognizing that the \mathbf{b}_i are arbitrary, produces

$$\begin{aligned} & \int_{\Omega^e} \rho_0 \mathbf{\Lambda} v_j^* \frac{\partial \mathbf{\Gamma}^T}{\partial x_j} d\mathbf{x} \mathbf{v}_i^b + \int_{\Omega^e} \mu \frac{\partial \mathbf{\Lambda}}{\partial x_j} \frac{\partial \mathbf{\Gamma}^T}{\partial x_j} d\mathbf{x} \mathbf{v}_i^b + \int_{\Omega^e} \mu \frac{\partial \mathbf{\Lambda}}{\partial x_j} \frac{\partial \mathbf{\Gamma}^T}{\partial x_i} d\mathbf{x} \mathbf{v}_j^b \\ & = - \int_{\Omega^e} \mathbf{\Lambda} \left[\rho_0 \frac{\partial \bar{v}_i}{\partial t} + \rho_0 v_j^* \frac{\partial \bar{v}_i}{\partial x_j} - \frac{\partial \bar{\sigma}_{ij}}{\partial x_j} - \rho \bar{f}_i \right] d\mathbf{x} \end{aligned} \quad (4.8.38a)$$

for each of the fine scale bubble components. Note again that the integration is only over a single element. In writing Eq. (4.8.38a), the second-order terms for the

fine scale have not been simplified by use of the continuity equation. This leads to a coupled system for the bubble coefficients which is inconvenient for the analytic solution. Therefore, using the continuity equation allows (4.8.38a) to be rewritten as

$$\begin{aligned} \int_{\Omega^e} \rho_0 \boldsymbol{\Lambda} v_j^* \frac{\partial \boldsymbol{\Gamma}^T}{\partial x_j} d\mathbf{x} \mathbf{v}_i^b + \int_{\Omega^e} \mu \frac{\partial \boldsymbol{\Lambda}}{\partial x_j} \frac{\partial \boldsymbol{\Gamma}^T}{\partial x_j} d\mathbf{x} \mathbf{v}_i^b &= - \int_{\Omega^e} \boldsymbol{\Lambda} d\mathbf{x} \mathcal{R}_i(\bar{u}) \\ &= - \int_{\Omega^e} \boldsymbol{\Lambda} \left[\rho_0 \frac{\partial \bar{v}_i}{\partial t} + \rho_0 v_j^* \frac{\partial \bar{v}_i}{\partial x_j} - \frac{\partial \bar{\sigma}_{ij}}{\partial x_j} - \rho \bar{f}_i \right] d\mathbf{x} \end{aligned} \quad (4.8.38b)$$

The pressure and viscous stress terms for the coarse scale have been integrated-by-parts to produce a strong form of the residual, which being assumed constant, is brought outside the integral. The stress tensor is used in the coarse scale terms to simplify the expression.

With the given assumptions, equation (4.8.38b) may be solved for the bubble coefficients \mathbf{v}_i^b which then provides the solution for the fine scale variables from (4.8.37). Therefore

$$v'_i = \boldsymbol{\Gamma}^T \mathbf{v}_i^b = -\tau \left[\rho_0 \frac{\partial \bar{v}_i}{\partial t} + \rho_0 v_j^* \frac{\partial \bar{v}_i}{\partial x_j} - \frac{\partial \bar{\sigma}_{ij}}{\partial x_j} - \rho f_i \right] = -\tau \mathcal{R}_i(\bar{v}) \quad (4.8.39)$$

where the stabilization parameter is

$$\tau = \boldsymbol{\Gamma}^T \int_{\Omega^e} \boldsymbol{\Lambda} d\mathbf{x} \left[\int_{\Omega^e} \boldsymbol{\Lambda} \rho_0 v_j^* \frac{\partial \boldsymbol{\Gamma}^T}{\partial x_j} d\mathbf{x} + \int_{\Omega^e} \mu \frac{\partial \boldsymbol{\Lambda}}{\partial x_j} \frac{\partial \boldsymbol{\Gamma}^T}{\partial x_j} d\mathbf{x} \right]^{-1} \quad (4.8.40)$$

The bubble functions in this application are usually polynomial functions and could be linear, quadratic or cubic. The $\boldsymbol{\Gamma}$ and $\boldsymbol{\Lambda}$ functions could be the same if a Galerkin method is used for the fine scale or the weighting function $\boldsymbol{\Lambda}$ could be an upstream weighted function if a Petrov-Galerkin method is considered [83]. Notice that a τ parameter is defined for each velocity component, i.e., τ is a matrix. Also, the stabilization parameters are easily defined and computed, element-by-element, from the definition in (4.8.40).

Returning to the coarse scale equations (4.8.34)–(4.8.35), the fine scale terms are integrated-by-parts a second time to produce the adjoint and leave the fine scale undifferentiated. That is

$$\begin{aligned} &\int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial \bar{v}_i}{\partial t} + v_j^* \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ &+ \int_{\Omega^e} \left[-\rho_0 \frac{\partial \bar{w}_i}{\partial x_j} v_j^* - \frac{\partial^2 \bar{w}_i}{\partial x_i^2} \mu \right] v'_i d\mathbf{x} \\ &= \int_{\Omega} \rho_0 \bar{w}_i f_i d\mathbf{x} + \oint_{\Gamma} \bar{w}_i \bar{T}_i ds \end{aligned} \quad (4.8.41)$$

$$-\int_{\Omega} \bar{Q} \frac{\partial \bar{v}_i}{\partial x_i} d\mathbf{x} + \int_{\Omega^e} \frac{\partial \bar{Q}}{\partial x_i} v'_i d\mathbf{x} = 0 \quad (4.8.42)$$

The second-order terms for the fine scale were simplified using continuity and the second term in the coarse scale continuity equation was also integrated. The fine

scale solution from (4.8.39) and (4.8.40) may be substituted into this form to obtain a single equation system for the coarse scales or

$$\int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial \bar{v}_i}{\partial t} + v_j^* \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} + \sum_{n=1}^{nel} \int_{\Omega^e} \left[\rho_0 \frac{\partial \bar{w}_i}{\partial x_j} v_j^* + \mu \frac{\partial^2 \bar{w}_i}{\partial x_i^2} \right] \tau \mathcal{R}_i(\bar{u}) d\mathbf{x} = \int_{\Omega} \rho_0 \bar{w}_i f_i d\mathbf{x} + \oint_{\Gamma} \bar{w}_i \bar{T}_i ds \quad (4.8.43)$$

$$- \int_{\Omega} \bar{Q} \frac{\partial \bar{v}_i}{\partial x_i} d\mathbf{x} - \sum_{n=1}^{nel} \int_{\Omega^e} \frac{\partial \bar{Q}}{\partial x_i} \tau \mathcal{R}_i(\bar{u}) d\mathbf{x} = 0 \quad (4.8.44)$$

Aside from a sign difference, this is the same form as the GLS stabilized system given in (4.8.1)–(4.8.7). This demonstrates the close relation between the GLS approach to stabilization and the VMS methodology. The VMS procedure provides a clear definition of the stabilizing parameter and methods for altering assumptions and defining other stabilizations.

Writing the steady form of (4.8.43) and (4.8.44) as a matrix system with the residual definition produces

$$[\mathbf{C}(\mathbf{v}^*)\mathbf{v} - \mathbf{Q}\mathbf{P} + \mathbf{K}\mathbf{v}] + [\mathbf{C}_{\delta}\mathbf{v} + \mathbf{Q}_{\delta}\mathbf{P} - \mathbf{K}_{\delta}\mathbf{v}] + [\mathbf{C}_{\beta}\mathbf{v} + \mathbf{Q}_{\beta}\mathbf{P} - \mathbf{K}_{\beta}\mathbf{v}] = \mathbf{F} \quad (4.8.45)$$

and

$$-\mathbf{Q}^T \mathbf{v} - \mathbf{C}_{\epsilon}(\mathbf{v}^*)\mathbf{v} - \mathbf{Q}_{\epsilon}\mathbf{P} - \mathbf{C}_{\epsilon}(\mathbf{v})\mathbf{v} + \mathbf{K}_{\epsilon}\mathbf{v} = \mathbf{0} \quad (4.8.46)$$

which can be rearranged to show the correspondence with the GLS in (4.8.9)–(4.8.10). Note that when low-order finite elements are used in either the GLS or VMS formulations, the subscript β matrices are zero due to the strong form of the diffusion operator.

Another approach to the solution of the fine scale equation that is less explicit than the polynomial bubble function method proceeds with some type of subdivision of each coarse element. Shown in Figure 4.8.1 are examples of some two-dimensional fine scale meshes used to represent the “bubble” part of the solution; other refinements on other types of elements are possible. The regular refinement was used by Franca and Nesliturk [97] in a two-level method and by Gravemeier, et al. [98], in a three-level method. The irregular refinement was suggested and tested by Dohrmann [99] where the interior mesh spacing was dictated by the local Reynolds number. On each coarse element, the fine scale variables are represented by the submesh assemblage of elements using standard basis functions; the fine scale variables are still constrained to be zero on the element boundary. A fine scale matrix problem, developed from Eq. (4.8.36) can be constructed based on a current estimate of the coarse scale residual. Static condensation of the fine scale degrees of freedom allows the fine scale to be expressed in terms of the coarse scale unknowns in the element. Symbolically,

$$\mathbf{K}_{cc}\mathbf{v}_c - [\mathbf{K}_{cf} \cdot \mathbf{K}_{ff}^{-1} \cdot \mathbf{K}_{fc}] \mathbf{v}_c = \mathbf{F}_c - \mathbf{K}_{cf} \cdot \mathbf{K}_{ff}^{-1} \mathbf{F}_f \quad (4.8.47)$$

where the subscript refers to the fine and coarse scales and the \mathbf{K} matrices represent the assembled terms of the equation. The term in the bracket, written at the element

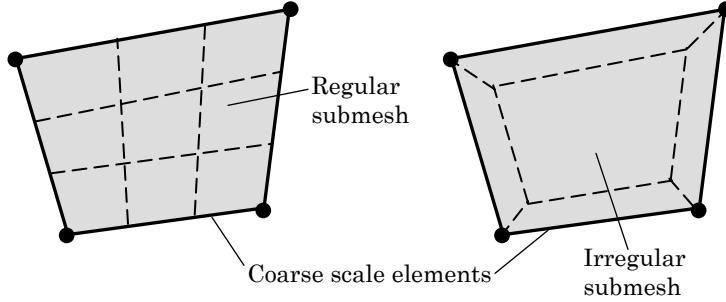


Figure 4.8.1: Two possible submesh grid refinements for the fine scale.

level, corresponds to the stabilization term with an embedded τ parameter. The inverse of the \mathbf{K}_{ff} matrix would be relatively inexpensive to compute due to its small size. As with other stabilization schemes, this process is easily accomplished in parallel implementations as everything is done element-by-element.

A final note on stabilization methods is worth reiterating. These modern stabilization approaches are mathematically consistent in the sense that solutions to the weak form are also solutions to the stabilized form of the variational statement.

4.9 Least-Squares Finite Element Models

4.9.1 Introduction

The application of the weak form Galerkin formulation to problems whose solutions can be characterized as global minimizers results in global minimization of quadratic functionals, such is the case for linear elasticity problems; see Reddy [2,100]. In this case, the finite element solution is an orthogonal projection of the exact solution onto the trial space, i.e., the finite element solution is a minimizer of an energy functional on the trial space so that it represents the best possible approximation in the energy norm. Such a setting, called a *variational setting*, is the most desirable for finite element model development.

Problems of the saddle-point type, whose solution can be interpreted as a constrained minimization of a convex functional by requiring adherence of the discrete spaces to restrictive compatibility conditions, lack many of the attractive features of the variational setting. Implementation of weak form Galerkin finite element models without accounting for the restrictive conditions leads to the occurrence of spurious oscillations in the numerical solution. A typical example is the primitive variable (mixed) formulation of the Stokes problem for which the velocity and pressure approximation spaces cannot be chosen independently and must satisfy an inf-sup condition, as already pointed out.

In the context of the Stokes and/or the Navier-Stokes equations, various finite element models attempting to fully or partially recover some of the properties of the variational setting have been proposed, and among them the Galerkin-least-squares and stabilized Galerkin methods have been extensively researched, and they were discussed in Section 4.8.

In the last two decades finite element models based on least-squares variational principles have drawn considerable attention (see, for example, [6-15, 101–116]). Given a partial differential equation or a set of partial differential equations, the least-squares method allows us to define a convex, unconstrained minimization principle so that a finite element model can be developed in a variational setting. Application of least-squares principles to the Navier–Stokes equations expressed in terms of the primitive variables (\mathbf{v}, P) requires that the approximation functions be *at least* once continuously differentiable across element boundaries (C^1 continuity)¹. Early implementations failed to gain popularity partly because of this requirement and partly because researchers did not realize the importance of the mesh parameter h , polynomial degree p , and degree of inter-element continuity k on the convergence and accuracy of least-squares finite element models.

Least-squares finite element models offer several theoretical and computational advantages over the weak-form Galerkin finite element models for viscous incompressible flows (see, e.g., Bochev and Gunzburger [15]); e.g., circumventing the inf-sup condition, thus allowing approximation spaces for velocities and pressure to be chosen independently, including the choice of equal-order interpolation. Furthermore, the resulting algebraic system is symmetric and positive-definite (SPD), which can be solved by robust and fast iterative methods, such as preconditioned conjugate gradient methods.

Unlike weak-form Galerkin formulations where regularity requirements of the finite element spaces are weakened by the integration-by-parts step, least-squares formulations require higher regularity of the finite element spaces, dictated by the order of the governing equation(s) being solved. To reduce the higher regularity requirements, the governing equation(s) are first recast into an equivalent lower-order system by introducing additional independent variables and then constructing the least-squares model of the system. Thus, to allow the use of practical C^0 nodal/modal expansions in the resulting finite element model, the Navier–Stokes equations are first recast as an equivalent first-order system and the least-squares functional defined in terms of L_2 norms only (see Jiang [6], Pontaza and Reddy [8,10,11], Prabhakar and Reddy [12,13], Bochev and Gunzburger [15,101,102], Cai, et al. [103], and Proot and Gerritsma [104], among others).

First-order systems that allow the construction of a L_2 least-squares functional that is H^1 -norm equivalent are commonly referred to as *H^1 -coercive formulations*. Such systems yield optimal error estimates with respect to the H^1 -norm for all variables [15] and ensure the optimality of multiplicative and additive multigrid methods [103,104], which could be used either as a solver or a preconditioner for the conjugate gradient method. However, not all L_2 least-squares formulations are H^1 -coercive. Such formulations are termed *non-equivalent* formulations, because the L_2 least-squares functional does not define an equivalent norm in H^1 . Nevertheless, a *non-equivalent* formulation does not imply that the method is not optimal. It simply means that the optimality of the resulting method cannot be established a priori using standard elliptic theory.

¹ A more strict requirement for variationally consistent formulations is that the finite element approximation functions belong to Hilbert space $\hat{H}^k(\Omega)$ (see [15, 100] for a discussion of function spaces), where $k \geq m$, m being the order of the differential equation; see Surana, et al. [7,9,14].

In the context of least-squares finite element formulations for the Navier–Stokes equations, predominantly low order nodal expansions have been used to develop the discrete finite element model. When the L_2 least-squares functional is not H^1 -norm equivalent (defining a non-equivalent formulation), low order nodal expansions tend to lock, and non-standard least-squares procedures such as collocation must be used to obtain acceptable numerical results. This is the preferred procedure in the work presented by Jiang [15], although he refers to the collocation solution as a reduced integration solution. It is important to note that reduced integration techniques will only result in a collocation solution if a strict balance between the number of collocation points and total number of degrees of freedom is satisfied. Thus, in general, blind application of reduced integration techniques will not result in a collocation solution. In addition, the least-squares functional cannot be used to measure the quality of the solution as it identically vanishes at the collocation points.

Even though a non-equivalent formulation departs from the ideal mathematical setting, it does not lead to disastrous results, as a violation of the inf-sup condition would in a mixed weak form Galerkin formulation. As shown by Pontaza and Reddy [8,10] and Prabhakar and Reddy [12,13], when high-order element nodal/modal expansions are used to construct the discrete finite element model, non-equivalent least-squares formulations are able to recover optimal properties. For smooth solutions, the L_2 least-squares functional decays exponentially fast as the element expansion order (p -level) is increased.

The most popular transformation to an equivalent first-order system for the incompressible Navier–Stokes equations is the vorticity based first-order system. In two-dimensions, the total number of variables is only increased by one, and this formulation has the benefit of directly solving for a quantity of physical relevance, the vorticity. However, the velocity-pressure-vorticity first-order system is not H^1 -coercive for particular sets of boundary conditions (details are given in [101,102]) and therefore fails to completely emulate a variational setting. Nevertheless, the formulation has had widespread acceptance, and in actual implementations performs exceptionally well (see Jiang [105] and Tang and Tsang [106]). Yet another approach is to introduce the stresses as independent variables, which leads to a stress based first-order system (see Prabhakar and Reddy [13]). In two dimensions the total number of variables is increased by three; however, the velocity-pressure-stress first-order system is not H^1 -coercive, regardless of the choice of boundary conditions [102]. A third option is to introduce all components of the gradient of the velocity vector field as independent variables [8,12]. By adding additional constraints to weaken the dependencies between variables this equivalent first-order system, hereafter referred to as the velocity gradient based first-order system, is always H^1 -coercive [15,103,107]. In two dimensions the total number of variables is increased by four and this formulation has the added benefit to easily compute (in the post-processing stage) physical quantities of interest that are linear combinations of the partial derivatives of the velocity vector field, e.g., vorticity and stresses.

Jiang and Sonnad [108] implemented a p -version least-squares formulation for the numerical solution of the stationary incompressible Navier–Stokes equations based on the velocity-pressure-vorticity first-order system. However, no detailed numerical results were reported. Winterscheidt and Surana [109] and Bagheri and

Surana [110] presented numerical results for the stationary incompressible Navier–Stokes stress based first-order system using a p -version least-squares formulation based on a modal basis derived from equi-spaced Lagrange polynomials. Reddy and his coworkers [12,111,112] formulated penalty least-squares finite element models of the Navier–Stokes equations for viscous incompressible flows.

Bell and Surana [113,114] presented numerical results for the non-stationary incompressible Navier–Stokes equations. They used the stress based first-order system and a space-time coupled least-squares formulation with a modal basis derived from equi-spaced Lagrange polynomials. Pontaza and Reddy [10] and Prabhakar and Reddy [112] presented space-time finite element models using mixed least-squares and penalty least-squares formulations. The works of Reddy and his coworkers [8,10-13,111,112] made use of the nodal and modal expansions associated with quadrilateral elements (see Warburton et al. [115,116]).

In this section we present the least-squares formulation for the Navier–Stokes equations governing flows of viscous incompressible fluids. We focus attention on first-order systems based on the velocity-vorticity formulation associated with the mixed formulation. We present numerical results for some benchmark problems, including flow around a cylinder [117], in Section 4.13.

4.9.2 Governing Equations

We consider the solution of the stationary Navier–Stokes equations governing incompressible flow, which in dimensionless form can be stated as follows:

Find the velocity $\mathbf{v}(\mathbf{x})$ and pressure $P(\mathbf{x})$ such that

$$(\mathbf{v} \cdot \nabla) \mathbf{v} + \nabla P - \frac{1}{\text{Re}} \nabla \cdot [(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T] = \mathbf{f} \quad \text{in } \Omega \quad (4.9.1)$$

$$\nabla \cdot \mathbf{v} = 0 \quad \text{in } \Omega \quad (4.9.2)$$

$$\mathbf{v} = \mathbf{v}^s \quad \text{on } \Gamma_v \quad (4.9.3)$$

$$\hat{\mathbf{n}} \cdot \sigma = \mathbf{f}^s \quad \text{on } \Gamma_\sigma \quad (4.9.4)$$

where $\Gamma = \Gamma_v \cup \Gamma_f$ and $\Gamma_v \cap \Gamma_f = \emptyset$, $\text{Re} = \rho U L / \mu$ is the Reynolds number, $\sigma = -P \mathbf{I} + (1/\text{Re}) [(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T]$, \mathbf{f} is a dimensionless force, $\hat{\mathbf{n}}$ is the outward unit normal on the boundary of Ω , \mathbf{v}^s is the prescribed velocity on the boundary Γ_v , and \mathbf{f}^s are the prescribed tractions on the boundary Γ_f . We assume that the problem is well posed and that a unique solution exists.

In situations where outflow boundary conditions need to be modeled, the Navier–Stokes equations in the $\nabla^2 \mathbf{v}$ form are preferred [118,119]. In such cases, using the incompressibility constraint given in Eq. (4.9.2), we would drop the $(\nabla \mathbf{v})^T$ term in Eq.(4.9.1), and the boundary conditions in Eq. (4.9.4) would then become

$$\hat{\mathbf{n}} \cdot \tilde{\sigma} = \tilde{\mathbf{f}}^s \quad \text{on } \Gamma_f \quad (4.9.5)$$

where $\tilde{\sigma}$ is a pseudo-stress, $\tilde{\sigma} = -P \mathbf{I} + (1/\text{Re}) \nabla \mathbf{v}$, and $\tilde{\mathbf{f}}^s$ are the prescribed pseudo-tractions on the boundary Γ_f .

Introducing the vorticity vector, $\omega = \nabla \times \mathbf{v}$, and making use of the vector identities

$$\nabla \times \nabla \times \mathbf{v} = -\nabla^2 \mathbf{v} + \nabla (\nabla \cdot \mathbf{v}), \quad \nabla \cdot [(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T] = \nabla^2 \mathbf{v} + \nabla (\nabla \cdot \mathbf{v}) \quad (4.9.6)$$

and the incompressibility constraint given in Eq. (4.9.2), Eqs. (4.9.1)–(4.9.4) can be replaced by the following equivalent first-order system:

$$(\mathbf{v} \cdot \nabla) \mathbf{v} + \nabla P + \frac{1}{\text{Re}} \nabla \times \omega = \mathbf{f} \quad \text{in } \Omega \quad (4.9.7)$$

$$\omega - \nabla \times \mathbf{v} = \mathbf{0} \quad \text{in } \Omega \quad (4.9.8)$$

$$\nabla \cdot \mathbf{v} = 0 \quad \text{in } \Omega \quad (4.9.9)$$

$$\nabla \cdot \omega = 0 \quad \text{in } \Omega \quad (4.9.10)$$

$$\mathbf{v} = \mathbf{v}^s \quad \text{on } \Gamma_v \quad (4.9.11)$$

$$\omega = \omega^s \quad \text{on } \Gamma_\omega \quad (4.9.12)$$

The seemingly redundant equation (4.9.10) is needed in the three-dimensional case to make the system of equations uniformly elliptic [15]. Typically $\Gamma_v \cap \Gamma_\omega = \emptyset$, i.e., if velocity is specified at a boundary, vorticity need not be specified there. Outflow boundary conditions can be treated through the least-squares functional and will be discussed in Section 4.13.

For the two-dimensional flows of viscous incompressible fluids, the vorticity based equivalent first-order system is given by

$$v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{\partial P}{\partial x} + \frac{1}{\text{Re}} \frac{\partial \omega_z}{\partial y} = f_x \quad (4.9.13)$$

$$v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} + \frac{\partial P}{\partial y} - \frac{1}{\text{Re}} \frac{\partial \omega_z}{\partial x} = f_y \quad (4.9.14)$$

$$\omega_z + \frac{\partial v_x}{\partial y} - \frac{\partial v_y}{\partial x} = 0 \quad (4.9.15)$$

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \quad (4.9.16)$$

Note that for the two-dimensional case the other two component of the vorticity vector are identically zero, $\omega = (0, 0, \omega_z)$.

4.9.3 Least-Squares Formulation

The L_2 least-squares functional associated with the velocity-pressure-vorticity formulation is given by

$$J(\mathbf{v}, P, \omega; \mathbf{f}) = \frac{1}{2} \left(\| (\mathbf{v} \cdot \nabla) \mathbf{v} + \nabla P + \frac{1}{\text{Re}} \nabla \times \omega - \mathbf{f} \|_0^2 + \| \omega - \nabla \times \mathbf{v} \|_0^2 \right. \\ \left. + \| \nabla \cdot \mathbf{v} \|_0^2 + \| \nabla \cdot \omega \|_0^2 \right) \quad (4.9.17)$$

Considering the homogeneous pure velocity boundary condition case, the least-squares principle for functional (4.9.17) can be stated as: find (\mathbf{v}, P, ω) such that for all (\mathbf{v}, Q, ψ)

$$J(\mathbf{v}, P, \omega; \mathbf{f}) \leq J(\mathbf{v}, Q, \psi; \mathbf{f}) \quad (4.9.18)$$

The variational statement resulting from this minimization problem forms the basis of the associated finite element model.

For the two-dimensional case represented by the system (4.9.13)–(4.9.16), the least-squares functional J in (4.9.17) over a typical element Ω^e takes the form

$$J^e = \frac{1}{2} \int_{\Omega^e} (\mathcal{R}_1^2 + \mathcal{R}_2^2 + \mathcal{R}_3^2 + \mathcal{R}_4^2) dx dy \quad (4.9.19)$$

where

$$\begin{aligned} \mathcal{R}_1 &= v_x \frac{\partial v_x}{\partial x} + v_y \frac{\partial v_x}{\partial y} + \frac{\partial P}{\partial x} + \frac{1}{Re} \frac{\partial \omega_z}{\partial y} - f_x \\ \mathcal{R}_2 &= v_x \frac{\partial v_y}{\partial x} + v_y \frac{\partial v_y}{\partial y} + \frac{\partial P}{\partial y} - \frac{1}{Re} \frac{\partial \omega_z}{\partial x} - f_y \\ \mathcal{R}_3 &= \omega_z + \frac{\partial v_x}{\partial y} - \frac{\partial v_y}{\partial x} \\ \mathcal{R}_4 &= \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} \end{aligned} \quad (4.9.20)$$

4.9.4 Finite Element Model

The variables of the problem (v_x, v_y, P, ω_z) are approximated by expansions of the form

$$\begin{aligned} v_x(x, y) &= \sum_{i=1}^N \psi_i(x, y) v_x^i, & v_y(x, y) &= \sum_{i=1}^N \psi_i(x, y) v_y^i \\ P(x, y) &= \sum_{i=1}^N \psi_i(x, y) P_i, & \omega_z(x, y) &= \sum_{i=1}^N \psi_i(x, y) \omega_z^i \end{aligned} \quad (4.9.21)$$

where ψ_i can be the usual or spectral family of Lagrange interpolation functions and $(v_x^i, v_y^i, P_i, \omega_z^i)$ are nodal values of (v_x, v_y, P, ω_z) .

In developing the finite element model, we assume that the convective term in the residuals associated with the two momentum equations has been linearized, i.e., v_x and v_y in the convective terms are evaluated from known values of v_x^i and v_y^i (from the preceding iteration). In essence this amounts to linearizing the least-squares functional prior to minimization. In principle, linearization after minimization is the correct approach, but in practice, it is found that both approaches yield the same results but the latter is more robust.

Minimizing the least-squares functional in Eq. (4.9.19) with respect to the nodal values of velocities, pressure, and vorticity, we obtain

$$\delta I^e = \frac{\partial I^e}{\partial v_x} \delta v_x + \frac{\partial I^e}{\partial v_y} \delta v_y + \frac{\partial I^e}{\partial P} \delta P + \frac{\partial I^e}{\partial \omega_z} \delta \omega_z = 0$$

which yields four sets of N equations each over a typical element:

$$\frac{\partial I^e}{\partial v_x^i} = 0, \quad \frac{\partial I^e}{\partial v_y^i} = 0, \quad \frac{\partial I^e}{\partial P_i} = 0, \quad \frac{\partial I^e}{\partial \omega_z^i} = 0 \quad (4.9.22)$$

for $i = 1, 2, \dots, N$. The resulting finite element equations are given by

$$\left[\begin{array}{cccc} [C^{00}(\mathbf{v})] & [0] & [C^{01}(\mathbf{v})] & \frac{1}{Re} [C^{02}(\mathbf{v})] \\ [0] & [C^{00}(\mathbf{v})] & [C^{02}(\mathbf{v})] & -\frac{1}{Re} [C^{01}(\mathbf{v})] \\ [C^{10}(\mathbf{v})] & [C^{20}(\mathbf{v})] & [0] & [0] \\ \frac{1}{Re} [C^{20}(\mathbf{v})] & -\frac{1}{Re} [C^{10}(\mathbf{v})] & [0] & [0] \end{array} \right] \left\{ \begin{array}{c} \{v_x\} \\ \{v_y\} \\ \{P\} \\ \{\omega_z\} \end{array} \right\} +$$

$$\begin{aligned}
& \begin{bmatrix} [S^{11} + S^{22}] & [S^{12} - S^{21}] & [0] & [S^{20}] \\ [S^{21} - S^{12}] & [S^{11} + S^{22}] & [0] & -[S^{10}] \\ [0] & [0] & [S^{11} + S^{22}] & \frac{1}{Re} [S^{12} - S^{21}] \\ [S^{02}] & -[S^{01}] & \frac{1}{Re} [S^{21} - S^{12}] & \frac{1}{Re^2} [S^{11} + S^{22}] + [S^{00}] \end{bmatrix} \begin{Bmatrix} \{v_x\} \\ \{v_y\} \\ \{P\} \\ \{\omega_z\} \end{Bmatrix} \\
&= \begin{Bmatrix} \{F^1\} \\ \{F^2\} \\ \{F^3\} \\ \{F^4\} \end{Bmatrix}
\end{aligned} \tag{4.9.23}$$

where the coefficient matrices are defined by

$$\begin{aligned}
C_{ij}^{00}(\mathbf{v}) &= \int_{\Omega^e} \mathcal{C}_i \mathcal{C}_j \, dx dy, & \mathcal{C}_i &= v_x \frac{\partial \psi_i}{\partial x} + v_y \frac{\partial \psi_i}{\partial y} \\
C_{ij}^{01}(\mathbf{v}) &= \int_{\Omega^e} \mathcal{C}_i \frac{\partial \psi_j}{\partial x} \, dx dy, & C_{ij}^{02}(\mathbf{v}) &= \int_{\Omega^e} \mathcal{C}_i \frac{\partial \psi_j}{\partial y} \, dx dy \\
C_{ij}^{10}(\mathbf{v}) &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial x} \mathcal{C}_j \, dx dy, & C_{ij}^{20}(\mathbf{v}) &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial y} \mathcal{C}_j \, dx dy
\end{aligned} \tag{4.9.24a}$$

$$\begin{aligned}
S_{ij}^{00} &= \int_{\Omega^e} \psi_i \psi_j \, d\Omega \\
S_{ij}^{01} &= \int_{\Omega^e} \psi_i \frac{\partial \psi_j}{\partial x} \, dx dy, & S_{ij}^{02} &= \int_{\Omega^e} \psi_i \frac{\partial \psi_j}{\partial y} \, dx dy \\
S_{ij}^{10} &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial x} \psi_j \, dx dy, & S_{ij}^{20} &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial y} \psi_j \, dx dy \\
S_{ij}^{11} &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial x} \frac{\partial \psi_j}{\partial x} \, dx dy, & S_{ij}^{22} &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial y} \frac{\partial \psi_j}{\partial y} \, dx dy \\
S_{ij}^{12} &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial x} \frac{\partial \psi_j}{\partial y} \, dx dy, & S_{ij}^{21} &= \int_{\Omega^e} \frac{\partial \psi_i}{\partial y} \frac{\partial \psi_j}{\partial x} \, dx dy
\end{aligned} \tag{4.9.24b}$$

$$\begin{aligned}
F_i^1 &= \int_{\Omega^e} \mathcal{C}_i f_x \, dx dy, & F_i^2 &= \int_{\Omega^e} \mathcal{C}_i f_y \, dx dy \\
F_i^3 &= \int_{\Omega^e} \left(\frac{\partial \psi_i}{\partial x} f_x + \frac{\partial \psi_i}{\partial y} f_y \right) \, dx dy \\
F_i^4 &= \int_{\Omega^e} \frac{1}{Re} \left(\frac{\partial \psi_i}{\partial y} f_x - \frac{\partial \psi_i}{\partial x} f_y \right) \, dx dy
\end{aligned} \tag{4.9.24c}$$

4.9.5 Computational Aspects

Inspection of the structure of the finite element equations in (4.9.23) reveals that the system is symmetric and positive-definite. In contrast with the mixed finite element model, where the resulting system of discrete equations is unsymmetric and indefinite (zeroes along the diagonal), the least-squares finite element model offers great advantages from a computational point of view. The symmetric positive-definiteness property allows the use of robust iterative methods for the solution of the discrete system of equations. Iterative solution techniques such as

preconditioned conjugate gradient methods can be implemented without the need of global assembly. Large scale problems can be solved using element-by-element solution procedures in a fully parallel environment.

In the context of least-squares finite element models for the incompressible Navier–Stokes equations, predominantly low order expansions have been used in the past. Although not commonly emphasized, low order approximations tend to lock, and reduced integration techniques must be used to obtain acceptable numerical results. When enough redundant degrees of freedom are constrained, the least-squares finite element model using reduced integration yields a collocation finite element model. However, the collocation finite element model may not always be reliable and the least-squares functional cannot be used to measure the quality of the solution. Moreover, the collocation solution may not be smooth at the nodes and post-processing is needed to recover nodal values from the reduced integration points.

Appropriate minimization of the least-squares functional is done using full integration and p -refinement (see Surana and coworkers [109,110] and Reddy and coworkers [8,10-13,111,112]). The quality of the numerical solution may be judged by the value of the least-squares functional, which decays exponentially as the expansion order of the basis is increased. Commonly used elements suited for p -refinement are either of the nodal or modal type. A nodal expansion is of the Lagrange type; when the node spacing is chosen such that the nodes coincide with the location of the roots of a Jacobi polynomial, the basis is known as a spectral basis. Modal bases are nodeless expansions whose coefficients are associated with modes of a hierarchical basis. Multidimensional nodal and modal bases can be easily constructed by taking tensor product of the one-dimensional basis. Details on the construction of both the nodal and modal expansions can be found in the work of Warburton et al. [115].

4.10 Post-Processing

4.10.1 Stress Computation

The results of interest from a flow analysis generally include fluid velocities, forces, and flow patterns (e.g., plots of velocities and pressure). Many of these items are directly available from the finite element results in terms of nodal point quantities; other quantities of interest may be derived from these primary variables. The computation of the viscous stress fields for the fluid flow follow directly from the finite element approximations of these variables, i.e., using kinematic and constitutive relations. A brief discussion of the stress calculation is presented next.

For a planar two-dimensional geometry, the components of the stress tensor ($\sigma_{xx}, \sigma_{yy}, \sigma_{xy}$) are known in terms of the pressure P and velocity components $v_1 = v_x$ and $v_2 = v_y$:

$$\sigma_{xx} = -P + 2\mu \frac{\partial v_x}{\partial x} \quad (4.10.1a)$$

$$\sigma_{yy} = -P + 2\mu \frac{\partial v_y}{\partial y} \quad (4.10.1b)$$

$$\sigma_{xy} = \mu \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right) \quad (4.10.1c)$$

where μ is the viscosity of the fluid. Substitution of the finite element approximations (4.2.6a,b) for v_i and P into Eqs. (4.10.1a-c) yields

$$\sigma_{xx} = -\Phi^T \mathbf{P} + 2\mu \frac{\partial \Psi^T}{\partial x} \mathbf{v}_x \quad (4.10.2a)$$

$$\sigma_{yy} = -\Phi^T \mathbf{P} + 2\mu \frac{\partial \Psi^T}{\partial y} \mathbf{v}_y \quad (4.10.2b)$$

$$\sigma_{xy} = \mu \left(\frac{\partial \Psi^T}{\partial y} \mathbf{v}_x + \frac{\partial \Psi^T}{\partial x} \mathbf{v}_y \right) \quad (4.10.2c)$$

where $(\mathbf{P}, \mathbf{v}_x, \mathbf{v}_y)$ denote the vectors of nodal values of (P, v_x, v_y) . The spatial derivatives of the interpolation functions in Eqs. (4.10.2a-c) can be converted to derivatives in terms of the local (element) coordinates (ξ, η) through the use of coordinate transformations defined in Eqs. (3.3.3) or (4.5.4). When this isoparametric transformation is invoked, the expressions for the stress components become

$$\begin{aligned} \sigma_{xx} &= -\Phi^T \mathbf{P} + \frac{\mu}{|\mathbf{J}|} \left[J^{*11} \frac{\partial \Psi^T}{\partial \xi} \mathbf{v}_x + J^{*12} \frac{\partial \Psi^T}{\partial \eta} \mathbf{v}_x \right] \\ \sigma_{yy} &= -\Phi^T \mathbf{P} + \frac{\mu}{|\mathbf{J}|} \left[J^{*21} \frac{\partial \Psi^T}{\partial \xi} \mathbf{v}_y + J^{*22} \frac{\partial \Psi^T}{\partial \eta} \mathbf{v}_y \right] \\ \sigma_{xy} &= \frac{\mu}{|\mathbf{J}|} \left[J^{*21} \frac{\partial \Psi^T}{\partial \xi} \mathbf{v}_x + J^{*22} \frac{\partial \Psi^T}{\partial \eta} \mathbf{v}_x + J^{*11} \frac{\partial \Psi^T}{\partial \xi} \mathbf{v}_y + J^{*12} \frac{\partial \Psi^T}{\partial \eta} \mathbf{v}_y \right] \end{aligned} \quad (4.10.3)$$

The function \mathbf{Y} [see Eqs. (4.5.4) and (4.5.7)] that occurs in the Jacobian matrix and its inverse (see Section 2.11), J^* , is defined as either a quadratic or linear interpolation function, depending on whether the particular element is isoparametric or subparametric. For axisymmetric flows, the stress field is computed in the same manner as discussed above but with the following definitions for the stress components:

$$\sigma_{rr} = -P + 2\mu \frac{\partial v_r}{\partial r}; \quad \sigma_{zz} = -P + 2\mu \frac{\partial v_z}{\partial z} \quad (4.10.4a)$$

$$\sigma_{\theta\theta} = -P + 2\mu \frac{v_r}{r}; \quad \sigma_{rz} = \mu \left(\frac{\partial v_z}{\partial r} + \frac{\partial v_r}{\partial z} \right) \quad (4.10.4b)$$

Computation of stresses in three-dimensional problems follows directly from the procedure presented above.

The expressions in Eq. (4.10.3) allow the stress components to be evaluated at any point (ξ_0, η_0) within an element for a known element geometry and solution field. Note that since the stresses depend on velocity gradients they are discontinuous between elements. As was the case for the heat flux evaluation, the derivative quantities are normally evaluated at the $2 \times 2 \times 2$ Gauss points for the interior of a hexahedral element, the 2×2 Gauss points for a quadrilateral element, or the 2 Gauss points on the edge of an element. These points have optimal accuracy for the shape function derivatives and correspond to a least-squares approximation for

the derivative [120,121]. The stresses computed at interior integration points can be extrapolated to the nodes by a simple linear extrapolation procedure, and they may be appropriately averaged between adjacent elements to produce a continuous stress field.

For some applications the vorticity field is also of interest and this quantity is computed in the same manner as the fluid stresses. By definition, the vorticity for a two-dimensional flow is given by [see Eq. (4.1.9)]

$$\omega = \frac{\partial v_y}{\partial x} - \frac{\partial v_x}{\partial y} \quad (4.10.5)$$

By using the same methods as outlined for the stress components, the vorticity can be evaluated at any point within an element. For three-dimensional flows, the additional components of the vorticity vector may also be computed. However, these quantities are seldom used in practical applications.

4.10.2 Stream Function Computation

A quantity that is often useful in the graphic display of computed flow fields is the stream function. For two-dimensional incompressible flows, the stream function is the remaining nonzero component of a vector potential which satisfies the conservation of mass equation identically. By definition,

$$v_1 = v_x = \frac{\partial \psi}{\partial y}; \quad v_2 = v_y = -\frac{\partial \psi}{\partial x} \quad (4.10.6)$$

and since the change in the stream function, $\delta\psi$, is an exact differential, then

$$\delta\psi = \int_A^B \mathbf{v} \cdot \hat{\mathbf{n}} ds = \int_A^B (v_x n_x + v_y n_y) ds \quad (4.10.7)$$

with

$$\mathbf{v} = v_x \hat{\mathbf{e}}_x + v_y \hat{\mathbf{e}}_y; \quad \hat{\mathbf{n}} = n_x \hat{\mathbf{e}}_x + n_y \hat{\mathbf{e}}_y \quad (4.10.8)$$

where $\hat{\mathbf{n}}$ is the unit normal to the integration path ds , \mathbf{v} is the velocity vector along the path, and $\hat{\mathbf{e}}_i$ are unit vectors in the coordinate directions.

The calculation of the change in the stream function within a finite element can be carried out using (4.10.7) once a suitable integration path AB is identified. In most applications, the integration path is taken along the element boundaries. Consider the typical element boundary shown in Figure 4.10.1 with the following definitions:

$$v_x = \hat{\Psi}^T \mathbf{v}_x; \quad v_y = \hat{\Psi}^T \mathbf{v}_y \quad (4.10.9a)$$

$$x = \hat{\Upsilon}^T \mathbf{x}; \quad y = \hat{\Upsilon}^T \mathbf{y} \quad (4.10.9b)$$

where $\hat{\Psi}$ and $\hat{\Upsilon}$ are interpolation (edge) functions and \mathbf{v}_x , \mathbf{v}_y , \mathbf{x} , and \mathbf{y} are vectors of nodal point velocities and coordinates. The unit normal vector is given by

$$\hat{\mathbf{n}} = \frac{1}{\Delta} \frac{\partial y}{\partial s} \hat{\mathbf{e}}_x - \frac{1}{\Delta} \frac{\partial x}{\partial s} \hat{\mathbf{e}}_y \quad (4.10.10)$$

with ds defined by

$$ds = \left[\left(\frac{\partial x}{\partial \hat{s}} \right)^2 + \left(\frac{\partial y}{\partial \hat{s}} \right)^2 \right]^{\frac{1}{2}} d\hat{s} = \Delta d\hat{s} \quad (4.10.11a)$$

Using the definitions of Eq. (4.10.9), we obtain

$$ds = \left[\left(\frac{\partial \hat{\mathbf{Y}}^T \mathbf{x}}{\partial \hat{s}} \right)^2 + \left(\frac{\partial \hat{\mathbf{Y}}^T \mathbf{y}}{\partial \hat{s}} \right)^2 \right]^{\frac{1}{2}} d\hat{s} = \Delta d\hat{s} \quad (4.10.11b)$$

Combining these relations with the definition for $\delta\psi$ produces

$$\delta\psi = \int_{-1}^{+1} \left(\frac{\partial \hat{\mathbf{Y}}^T \mathbf{y}}{\partial \hat{s}} \hat{\mathbf{Y}}^T \mathbf{v}_x - \frac{\partial \hat{\mathbf{Y}}^T \mathbf{x}}{\partial \hat{s}} \hat{\mathbf{Y}}^T \mathbf{v}_y \right) d\hat{s} \quad (4.10.12)$$

The interpolation function definitions were given in Section 4.5; the vector $\hat{\mathbf{Y}}$ can be either linear or quadratic, depending on the shape of the element boundary. The change in the stream function along any element boundary can be computed from Eq. (4.10.12) once the element geometry and velocity fields are specified. Computation of the stream function field for an entire finite element mesh is generated by applying Eq. (4.10.12) along successive element boundaries, starting at a node for which a reference value of ψ has been specified.

The calculation of the stream function for axisymmetric flows follows a similar procedure with the appropriate definition for ψ , \mathbf{v} , and $\hat{\mathbf{n}}$

$$v_1 = v_r = \frac{1}{r} \frac{\partial \psi}{\partial z}; \quad v_2 = v_z = -\frac{1}{r} \frac{\partial \psi}{\partial r} \quad (4.10.13)$$

$$\mathbf{v} = v_r \hat{\mathbf{e}}_r + v_z \hat{\mathbf{e}}_z; \quad \hat{\mathbf{n}} = n_r \hat{\mathbf{e}}_r + n_z \hat{\mathbf{e}}_z \quad (4.10.14)$$

A second approach to computing the stream function for a given velocity field relies on the definition of the vorticity. By substituting the stream function definition in Eq. (4.10.6) into the vorticity relation in (4.10.5), the following elliptic partial differential equation is obtained:

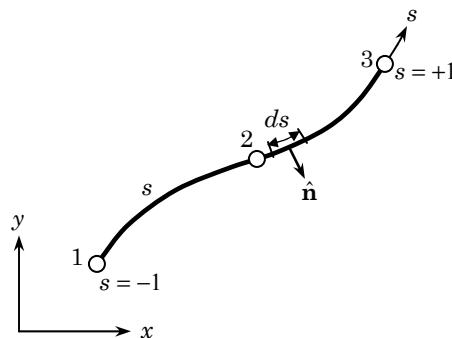


Figure 4.10.1: Element boundary for stream function computation.

$$-\nabla^2\psi = \omega \quad (4.10.15)$$

This, being a Poisson equation, can be easily cast into a finite element model and solved for ψ , since ω is known everywhere in the domain (see Problem 4.4). The main disadvantage to this approach is that boundary conditions for ψ must be supplied for all points of the boundary and, more importantly, a separate matrix problem must be solved for each velocity field. The element-by-element procedure is computationally more efficient and it is generally the method of choice.

4.10.3 Particle Tracking

Another useful method for visualizing fluid motion involves following the movement of a massless particle that is introduced at a convenient point in the flow. The tracking of particle paths is governed by the set of ordinary differential equations

$$\frac{dx_i}{dt} = \dot{x}_i = v_i \quad (4.10.16)$$

Given the initial location of a point, $P(x_i)$, the initial value problem in (4.10.16) must be solved to describe the particle trajectory for the known flow field v_i . The integration of (4.10.16) is most easily accomplished on an element-by-element basis. Using the standard definitions for an isoparametric mapping of the element geometry and the element approximation for velocity produces

$$\mathbf{J} \begin{Bmatrix} \dot{\eta} \\ \dot{\xi} \\ \dot{\zeta} \end{Bmatrix} = \begin{Bmatrix} \Psi^T(\xi, \eta, \zeta)\mathbf{v}_1 \\ \Psi^T(\xi, \eta, \zeta)\mathbf{v}_2 \\ \Psi^T(\xi, \eta, \zeta)\mathbf{v}_3 \end{Bmatrix} \quad (4.10.17)$$

where the Jacobian matrix \mathbf{J} was defined in Section 3.4 and (ξ, η, ζ) are the local element coordinates. For a known starting location (ξ_0, η_0, ζ_0) and velocity field, Eq. (4.10.17) can be integrated by standard methods to yield the particle trajectory across the element. When the particle crosses an element boundary, the procedure is restarted with a new element Jacobian matrix and a new element velocity field.

As shown in (4.10.17), the velocity field is independent of time and the integration can be performed with the fixed values of \mathbf{v} . Time-dependent flows may also be computed [in which case $\mathbf{v} = \mathbf{v}(t)$]. Conceptually, the method is the same but the numerical implementation now requires interpolation to find the needed particle tracking velocities from the finite element time planes.

It is also possible to generalize this method to consider the motion of real particles in the flow. In this case, a force balance on the particle is required that accounts for drag, gravity effects, or other forces, and predicts the particle velocity. The particle velocity is then used in Eq. (4.10.16) in place of the fluid velocity; the integration procedure is similar to the one shown in (4.10.17).

4.11 Free Surface Flows

4.11.1 Preliminary Comments

Many Newtonian and non-Newtonian flow problems of practical interest contain a complicating feature in the form of a free surface boundary. Boundary conditions for the free surface problem were given in Section 1.10.1 and were seen to be somewhat different for the steady-state flow problem and the case of a dynamic free surface. Likewise, the solution algorithms for steady and time-dependent free surface flows tend to be quite different since the boundary interface conditions dictate the numerical procedure. In this section we provide an overview of some types of procedures used for both static (quasi-static) free surfaces and advancing front type problems. A complete discourse on free surface methods is available in the text by Hervouet [122].

4.11.2 Time-Independent Free Surfaces

Steady flow problems with a free surface boundary are somewhat easier than the dynamic case since only a single location and shape of the free surface needs to be computed. For some geometries, such as extrusions and jets with a simple cross section, the free surface shapes are relatively simple and the required algorithms are straightforward and easily implemented. When the free surface shape is very complex, such as found in coating flows or fluid drop problems, the numerical algorithms are necessarily more sophisticated. Steady free surface methods universally employ some type of mesh updating or mesh movement algorithm; these are surface capturing methods in which the free surface always coincides with edges (surfaces) of the finite element discretization. Such procedures allow for the accurate representation of surface phenomena, such as surface tension forces, but at the cost of mesh manipulation. The complexity of the following algorithms is directly proportional to the methods used for mesh adjustment. In the following, the derivations and discussion are limited to two-dimensional problems to simplify the notation and equations; the extension to three dimensions is conceptually the same, although the details are more complex.

The appropriate boundary conditions for a free surface involve the balance of forces in the directions normal and tangent to the surface. Neglecting drag effects from the surrounding medium, these conditions are given by Eq. (1.10.7) and are written for the two-dimensional case as

$$\sigma_{nn} = \gamma \left(\frac{1}{R} \right) - P_{amb} \quad (4.11.1)$$

$$\sigma_{ns} = 0 \quad (4.11.2)$$

where γ is the surface tension, R is the principal radius of curvature of the surface, and P_{amb} is the ambient pressure. Note that Eq. (4.11.1) is written for a planar problem; an axisymmetric problem would require consideration of a second radius of curvature. In addition to the boundary conditions (4.11.1) and (4.11.2), a kinematic condition on the surface is required; the free surface is a streamline and thus

$$v_i n_i = 0 \quad (4.11.3)$$

where n_i ($i = 1, 2, 3$) are the components of the outward normal, and v_i are the components of the velocity vector. For many applications, surface tension effects are negligible. Also, without loss of generality the ambient pressure may be set to zero. Thus, the appropriate boundary conditions in many cases correspond to vanishing of the normal and tangential components of the stress vector on the free surface.

The zero stress conditions encountered for the zero surface tension case are precisely the conditions that are enforced by the “natural” boundary conditions arising in the Galerkin finite element form of the momentum equations. With these conditions automatically enforced, it remains to find a method for locating the position of the free surface. For simple free surface shapes a procedure proposed by Tanner, et al. [123] and used by others, (see [124,125]) can be employed.

The computation starts with an assumed free surface shape and a given finite element mesh. The appropriate boundary conditions are applied and the problem solved for a velocity and pressure field. In general, the computed velocity field will not satisfy the kinematic constraint in Eq. (4.11.3) along the assumed free surface boundary. Referring to Figure 4.11.1, the location of the free surface is updated using the following relation

$$\frac{d}{dx}f(x) = \frac{v_y}{v_x} \quad (4.11.4)$$

where (v_x, v_y) are the velocity components on the surface. Equation (4.11.4) can be integrated numerically (e.g., Simpson’s rule) to produce a new surface shape $f(x)$. The success of the integration is dependent on knowing at least one (fixed) point on the free surface. With a new free surface location a new finite element mesh can be constructed and the above procedure is repeated until convergence is obtained. Mesh reconstruction in this type of algorithm usually amounts to a one-dimensional rescaling of the element locations in a particular coordinate direction. More sophisticated remeshing methods could be used but are generally not cost effective since the mesh movement is relatively small. For problems with material and/or convective nonlinearities, the free surface iteration is normally intertwined with the other iterative procedures. Mapping of the current solution field to the new mesh is not normally required due to the size of the mesh movement.

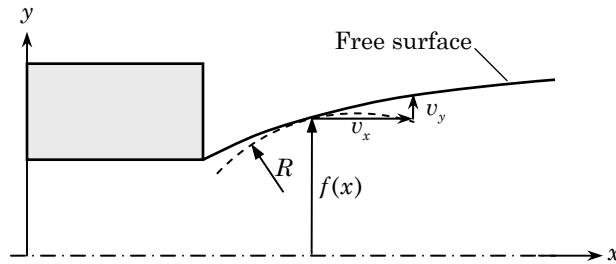


Figure 4.11.1: Nomenclature for free surface boundary computations.

When surface tension is not negligible, the procedure outlined above may still be used but with minor modifications. Instead of applying the zero normal stress boundary condition, a normal force due to the surface tension must be applied at each iteration. Again referring to Figure 4.11.1, the radius of curvature is given by

$$R = \frac{\left[1 + \left(\frac{df}{dx}\right)^2\right]^{3/2}}{\left(\frac{d^2f}{dx^2}\right)} \quad (4.11.5)$$

which may be computed at any iteration since $f(x)$ is then known. Equations (4.11.1) and (4.11.5) allow the normal stress at the free surface to be computed and applied at each cycle of the free surface iteration. The procedure outlined above has been used to study a number of extrusion and jet flow problems with good success. However, for applications where the shape of the free surface is rather complex and the indicated procedure is not adequate, more sophisticated algorithms must be employed.

An alternative approach, developed by Saito and Scriven [126] and Scriven and Kistler [127], employs a finite element parameterization in which location parameters (degrees of freedom) for the free surface become unknowns in the general problem formulation. Though the details of implementation are fairly involved, the general idea can be obtained from the sketch in Figure 4.11.2. As indicated by the figure, nodes on the free surface are free to move along fixed lines, called spines, such that their location relative to a fixed base point is determined by a single nodal parameter, h_i . The coordinates for the free surface nodes are given by

$$x_i^{fs} = \alpha_x^i h_i + x_i^b, \quad y_i^{fs} = \alpha_y^i h_i + y_i^b \quad (4.11.6)$$

where α^i are components of the direction vector along the spine, and x_i^b and y_i^b are coordinates of the base point. Note that in an actual implementation, nodes for all the elements between the free surface and the base line would also be parameterized such that node spacing ratios would be maintained with movement of the free surface. To complete the formulation for the free surface, it is observed that the surface is composed of a series of element edges with each edge containing two or three nodes, depending on the type of finite element being employed. The isoparametric description of the element edge then has the form

$$\mathbf{x} = \boldsymbol{\Upsilon}^T \mathbf{x}^{fs}, \quad y = \boldsymbol{\Upsilon}^T \mathbf{y}^{fs} \quad (4.11.7)$$

where \mathbf{x}^{fs} and \mathbf{y}^{fs} are computed using Eq. (4.11.6), and $\boldsymbol{\Upsilon}$ is an appropriate linear or quadratic edge interpolation function.

With the above relations, the standard finite element discretization of the momentum and continuity equations can be implemented. The proper accounting must be made in the boundary integrals for the free surface stresses acting on the parameterized element edges. A final equation is also needed since the h parameters are now unknowns in the global equation set. The last equation needed is the kinematic condition (4.11.3) which can be written in a weighted residual form as

$$\int_{\Gamma_{fs}} \boldsymbol{\Upsilon} v_n d\Gamma = \int_{\Gamma_{fs}} \boldsymbol{\Upsilon} (v_x n_x + v_y n_y) d\Gamma = 0 \quad (4.11.8)$$

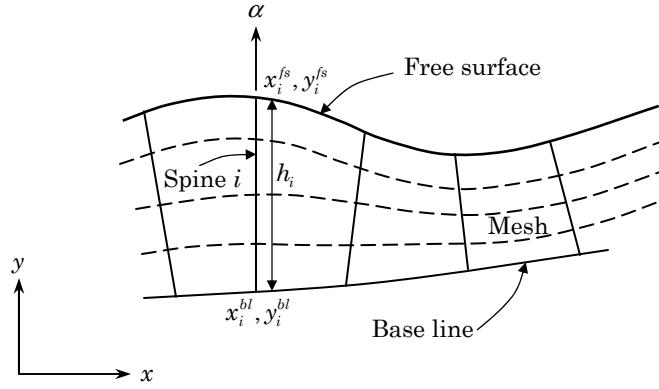


Figure 4.11.2: Nomenclature for spines used in free surface description.

where the weighting function is the interpolation function defining the free surface [see Eq. (4.11.7)]. With the proper definition for the velocities in the element and the mass consistent normals at the surface [128], the surface integrals in Eq. (4.11.8) can be evaluated and assembled to provide an equation for each boundary node. Analogous with the matrix equations (4.11.5) and (4.11.6), the finite element model for a free surface flow is then given by

$$\mathbf{C}(\mathbf{u})\mathbf{u} + \mathbf{K}(\mathbf{v})\mathbf{v} - \mathbf{Q}\mathbf{P} + \mathbf{B}\mathbf{h} = \mathbf{F} \quad (4.11.9a)$$

$$-\mathbf{Q}^T\mathbf{v} = \mathbf{0} \quad (4.11.9b)$$

$$\mathbf{K}_{fs}\mathbf{v} = \mathbf{0} \quad (4.11.9c)$$

where \mathbf{h} is the global vector of free surface parameters, \mathbf{B} is the matrix corresponding to the surface stress conditions, and \mathbf{K}_{fs} is the matrix resulting from the kinematic constraint in Eq. (4.11.8).

The solution to Eq. (4.11.9) proceeds in the usual way, with the use of Newton's method being the preferred algorithm. Other fixed point schemes such as the Picard method are not employed here since at each iteration the velocities would have to be interpolated onto the new nodal point locations defined by new \mathbf{h} values. However, the Newton procedure avoids this difficulty if the full Jacobian matrix is used, i.e., the variations of all equations with respect to the free surface parameters \mathbf{h} are constructed. The construction of these new entries of the Jacobian matrix is not trivial since both the integrands and the limits of integration depend on \mathbf{h} . The usual methods for isoparametric mapping of the elements can be employed to simplify this task and make the Newton algorithm usable.

The method of spines is a formal method of embedding the kinematic free surface condition directly into the equations for the system. The method given in Eq. (4.11.4) uses the kinematic condition outside the basic equations as a control on the (free surface) iteration procedure. Though the spine algorithm is more general and can handle some problems of great complexity, it is not without its drawbacks. The fact that nodes are constrained to a “one-dimensional” motion along the spine limits severely the ability of the mesh to conform to a complex surface shape. Re-entrant corners are particularly difficult to describe with spines. In essence, the initial mesh design and spine placement must anticipate the final free surface shape.

The method is practical in some two-dimensional applications and a limited number of three-dimensional flows.

To avoid some of these difficulties, work has progressed on changing the way the mesh is parameterized but retaining most of the other solution features of the spine approach. Christodoulou and Scriven [129] have proposed using body-fitted coordinates and elliptic mesh generation to parametrize the entire mesh. This appears to be more flexible and useful than spines and is more readily extendable to three dimensions. Elliptic mesh generation does, however, have its limitations with regard to mesh control and mesh quality. Other mesh generation methods could be used, though high quality automatic meshing of complex geometries remains an area of substantial research and development. A review of some of these free surface algorithms and mesh update methods can be found in [130].

A more recent development with regard to mesh adjustment is the use of a pseudo-structural method to define the movement of the elements [131,132]. If the meshed region is viewed as a solid body then the element edges can be treated as springs with a given distribution of spring (stiffness) constants. When the free surface is adjusted, i.e., given a displacement, the interior nodes of the elements are adjusted to a new equilibrium position and a mesh update is completed. A more formal approach treats the meshed region as an elastic body with a distribution of elastic material parameters. When the free surface displacements are specified, an elastic boundary value problem is defined that can be solved via a finite element method. The new displacement field defines a new mesh for the subsequent resolution of the flow problem. The elasticity problem and mesh movement may be fully coupled to the flow problem as described above for the spine parameterization [133]. The elasticity problem can also be decoupled from the flow problem and solved separately in a staggered type of algorithm [134]. In this latter case, a mapping or projection of the velocity field from the old mesh onto the new mesh will be required for problems with a non-zero Reynolds number.

The usual pseudo-structural approach employs the equilibrium equations for an elastic body which can be written as

$$\nabla \cdot \sigma^s + \mathbf{f}^s = \mathbf{0} \quad (4.11.10)$$

where \mathbf{f}^s is the body force vector and σ is the stress tensor for the solid. By Hooke's law, we have

$$\sigma^s = \lambda (\text{tr } \varepsilon^s) \mathbf{I} + 2\mu \varepsilon^s \quad (4.11.11)$$

and the strain tensor ε^s is related to the displacement vector \mathbf{d} by

$$\varepsilon^s = \frac{1}{2} [\nabla \mathbf{d} + (\nabla \mathbf{d})^T] \quad (4.11.12)$$

In Eq. (4.11.11), λ and μ are the Lamé constants. The above equations describe a boundary value problem for the elastic response of a body and may be cast in computational form via a weak or variational formulation. A finite element approximation ultimately produces a matrix problem of the form

$$\mathbf{K}^s \mathbf{d} = \mathbf{F} \quad (4.11.13)$$

for the nodal point displacements \mathbf{d} . Some of the boundary conditions for the displacements in (4.11.13) will be generated from the movement (adjustment) of the free surface to satisfy the kinematic free surface condition. The remaining boundary conditions on displacement will need to be specified so as to allow the mesh to conform to boundaries of the flow domain while minimizing mesh distortion. Once the boundary displacements are specified and a distribution of material parameters λ and μ for the solid have been selected, the problem in (4.11.13) may be solved for a new set of interior displacements. The solution to Eq. (4.11.13) provides displacements that can be used to update the locations of the nodes in the mesh. A projection of the fluid velocity field from the old mesh to the new mesh completes the iterative cycle and permits the flow problem to be solved again on a new geometry.

A simple linear elastic constitutive relation was described in (4.11.11) and this may not be suitable for many mesh moving applications. Some nonlinear relations have been suggested [133] as well as methods for selecting the distribution of elastic constants over the meshed region [134,135]. Even though the mesh movement in this approach is based on a physical principle there is no guarantee that acceptable or accurate meshes will result. Severe mesh distortions remain a problem for complex flows and if distortion cannot be controlled, remeshing of the geometry may be required. In many applications free surface analysis is still something of an art form.

4.11.3 Time-Dependent Free Surfaces

Methods for time-dependent free surface problems fall naturally into two distinct types of algorithms. The moving mesh methods, as outlined in the previous section, can be extended to handle some types of dynamic free surfaces. These methods would typically be applicable to flows in which the free surface motion was limited and mesh distortion would not be a significant difficulty. When free surface motions are large, the quasi-Lagrangian methods based on mesh updating are no longer very convenient. The second type of method relies on a fixed (Eulerian) mesh and attempts to track the location of the free surface through a variable defined on the mesh. Whereas the mesh moving methods explicitly define the free surface, the fixed mesh schemes implicitly define the free surface location and motion. Physical phenomena that occur on the free surface (e.g., surface tension) are easily accommodated with the explicit moving mesh methods since the finite element variables coincide with the free surface. Surface physics are difficult to accurately include in the basic fixed mesh algorithms though complex bulk flows are well described with these methods. In the following, the mesh moving algorithms are briefly updated for time dependence and then fixed mesh algorithms are described.

The mesh moving methods outlined in the previous section can, in some cases, be extended to handle time-dependent free surface problems. The pseudo-structural method is probably the most robust and most generally useful of the various update methods. To incorporate this technique into a time dependent problem primarily requires the redefinition of the material or convected derivative in the fluid equation of motion. The convected derivative was defined in Eq. (4.1.4) and needs to be redefined as

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + (\mathbf{v} - \mathbf{v}^m) \cdot \nabla, \quad \frac{D}{Dt} \equiv \frac{\partial}{\partial t} + (v_j - v_j^m) \frac{\partial}{\partial x_j} \quad (4.11.14)$$

where the superscript m indicates the velocity of the mesh. Use of this derivative in the fluid momentum equation (4.1.2) or in the weak form (4.2.5) allows the fluid motion to be described in a time dependent mesh. This type of formulation is termed an Arbitrary Lagrangian Eulerian (ALE) method since the mesh motion is described in a Lagrangian frame while the fluid motion remains in an Eulerian description.

The remaining part of the ALE scheme is the algorithm for moving the mesh and generating the mesh velocity \mathbf{v}^m . To utilize the pseudo-structural method in the time dependent case, the equilibrium equation (4.11.10) could be differentiated with respect to time to directly produce a stress rate and subsequently a displacement rate or mesh velocity. The equilibrium equations could also be solved in their steady form for the displacement that occurs over the time step; the displacement increment divided by the time step produces the mesh velocity. The overall ALE method may be fully coupled, in which case the fluid and pseudo-solid equations are solved simultaneously at a time step, with Newton's method providing a typical iteration procedure. The fluid and mesh motion could also be solved in a decoupled manner. Iterative convergence within a time step would be slower than the fully coupled method, though the computational work per iteration would generally be reduced.

One of the major drawbacks to ALE type methods is the problem of mesh distortion. Except in special cases, such as almost one-dimensional free surface motions or some types of periodic motions, the movement of the elements will eventually reach the point where the mesh is badly deformed and the accuracy of the finite element approximation is severely degraded. To continue the simulation the flow domain must be remeshed on the current geometry and the solution fields transferred from the old mesh to the new mesh. Automatic remeshing of a general geometry is a nontrivial task and is still the subject of current research. The transfer of field data from one mesh to another is done most simply by interpolation. Specialized programs exist that perform this function by finding new node locations in the old mesh and using the old mesh shape functions to interpolate the dependent variables [136,137]. For an incompressible flow, the interpolated velocities will not generally satisfy the incompressibility constraint; restarting the simulation with an erroneous velocity field may produce nonphysical results, especially the pressure, depending on the time integration method that is used. A mass consistent velocity field can be produced on the new mesh but at the expense of solving another matrix problem [138]. The mesh distortion and remeshing problems are sufficiently difficult and computationally expensive that ALE methods are usually limited to transient free surface flows with small motions or special geometries that avoid mesh distortion.

When large scale free surface motions are to be simulated, as in many cavity filling flows, fixed mesh methods with some type of front tracking procedure are normally the most useful. Many of the techniques that are used in a finite element context fall in the category of methods commonly known as Volume of Fluid (VOF) methods, or more correctly, Volume Tracking (VT) methods. These methods originated in finite difference applications and have evolved to a variety of algorithms that differ mainly in the details of the implementation. Rider and Kothe [139] present a summary of many of the different approaches used in VT methods. Here the method is outlined in a generic form in a finite element context.

Before describing the volume tracking algorithm, consider a simpler method that has some of the basic features of many fixed mesh methods. A fluid concentration variable, usually denoted by F , is defined as a nodal point variable on the finite element mesh. The concentration variable has a value of unity at locations within the fluid and zero at locations outside the fluid; the free surface is implicitly located between the 0 to 1 extremes. As a passive scalar concentration variable, the F function is transported by the fluid motion, without diffusion, according to the hyperbolic equation

$$\frac{\partial F}{\partial t} + v_j \frac{\partial F}{\partial x_j} = 0 \quad (4.11.15)$$

where v_j is the fluid velocity. In a finite element framework, Eq. (4.11.15) could be discretized using a Galerkin weighted-residual formulation to produce an advection equation for the nodal values of F . Being hyperbolic, Eq. (4.11.15) requires only initial values of F . As the function F is computed at nodes throughout the mesh, the relevant fluid momentum and continuity equations are solved on an ever-changing set of grid points indicated by the condition $F = 1$. The overall method is a staggered, cyclic procedure in which the flow problem is solved on the domain defined by $F = 1$, followed by an updating of the flow domain by advancing (4.11.15) in time. Equation (4.11.15) could be solved using either an explicit or implicit time integration method, while the fluid mechanics part of the problem is solved implicitly. In either case, the concentration equation can be subcycled to improve the temporal accuracy of the free surface evolution. At the free surface (usually indicated by $F = 0.5$) the usual stress boundary conditions are applied; the standard choice is for a traction free surface as surface tension effects are somewhat difficult to incorporate in this formulation.

Unfortunately, free surface boundary conditions are not the only difficulty with the method outlined above. Theoretically, the free surface position in this algorithm is described by the propagation of a square wave profile in F on a fixed mesh; computationally, the steepest or sharpest variation in F that can be resolved is fixed by the smallest node spacing in the flow direction. In any case, it is very difficult, if not impossible, to transport a steep function across a variably spaced mesh without significant dispersion and spatial oscillation of the profile. This difficulty prohibits the accurate description of the free surface evolution and requires the modification of the above approach. The usual volume tracking algorithm eliminates the precise definition of the free surface in favor of a new variable that evaluates the volume of fluid in each element as a function of time. This new variable, the volume fraction, is defined for each element j by

$$f_j = \frac{1}{V_j} \int_{V_j} F dV_j \quad (4.11.16)$$

where F is the concentration variable in (4.11.15) and V_j is the element volume; j runs from 1 to the number of elements. The volume fraction also varies between 0 and 1, with $f_j = 1$ being a filled element, $f_j = 0$ being an empty element and $0 < f_j < 1$ being a partially filled element that contains the free surface.

The VT algorithm proceeds in two stages and is a cyclic, staggered algorithm as noted above. After initialization of f_j , the fluid volumes are first advected forward in time followed by an interface reconstruction step. The fluid mechanics problem is then solved on an updated domain to complete the cycle. Initialization consists of defining f_j for each element once the initial free surface location is specified. Given a fluid velocity field and the reconstructed location of the free surface, the volume fractions are evolved forward in time using some form of the advection equation in Eq. (4.11.15). Like the concentration variable F , the volume fraction variable must also satisfy a conservation equation

$$\frac{\partial f}{\partial t} + v_j \frac{\partial f}{\partial x_j} = 0 \quad (4.11.17)$$

if f is viewed as a continuous function. Since the volume fraction has been defined on each element it is discontinuous and the partial differential form of (4.11.17) is rarely used. Instead, a control volume form of (4.11.17) is defined that balances the net inflow to each element with the change in volume fraction within the element. A typical element balance is

$$\frac{\partial f_j}{\partial t} = \frac{f_j^{n+1} - f_j^n}{\Delta t} = \frac{1}{V_j} \sum_{k=1}^{nfaces} q_{jk}^n \quad (4.11.18)$$

where the flow rate for face k of the element is

$$q_{jk} = - \int_k v_i n_i dA \quad (4.11.19)$$

The control volume relation in (4.11.18) has been written as an explicit time integration and is therefore subject to a time step limit that respects the hyperbolic nature of the original equation. This time step limit or Courant–Friedrich–Levy (CFL) stability limit also prevents the over-filling or emptying of an element. An implicit integration method is not practical in this cyclic type of procedure since the velocity would be required at the new time level. Note that in (4.11.19) the integral is over the area of the element face. In a partially filled or filled element, where (4.11.18) and (4.11.19) would be used, the part of the element face over which a fluid velocity is defined depends on the location of the free surface within the element and within the adjoining elements. This dependence leads to the other step in the VT algorithm where the fluid interface is reconstructed from the volume fraction data. This step actually precedes the advection step but since the algorithm is cyclic, the order of explanation is not critical.

When the volume fraction variable is adopted, precise information on the location of the fluid interface within an element is lost. This information can be recovered through a reconstruction algorithm that varies depending on the particular VT method. It is important to realize that the reconstruction process is not unique and may be based on heuristic, geometric or algebraic methods [139]. Because so many reconstruction methods are available no attempt will be made to catalog the various approaches and implementations. Rather a simple geometric method in two dimensions will be used to indicate the basic ideas.

The interface reconstruction has a limited amount of information to work with. The volume fraction of the element in question and the volume fractions of the surrounding elements form the sole basis for many algorithms. Consider the patch of elements in Figure 4.11.3, where the center element has a volume fraction of $f_e = 0.25$. If element b is filled or partially filled ($f_b > 0$) and the remaining elements are empty ($f_d = f_f = f_h = 0$), then one reconstruction method would place the fluid volume in element e along the boundary with element b in a square shape (see Figure 4.11.3a). In this particular method the partial volume fraction within an element is always square or rectangular in shape and is described by constant coordinate lines in the master or parent element. Figure 4.11.3b shows the case where an additional element is completely filled ($f_f = 1$) and the reconstruction would move the partial fluid volume to the corner of element e while maintaining the square shape. If element f had been only partially filled, element e would have remained, as shown in Figure 4.11.3a, since location is most heavily weighted toward completely filled elements. For the case where both elements b and f are partially filled the pattern would be as shown in Figure 4.11.3b since the weighting is equal between the two elements. Figure 4.11.3c illustrates the case of three surrounding elements that are partially filled or completely filled and the change in the partial volume shape in element e to a rectangle. Finally, the case where opposing elements are partially or completely filled is shown in Figure 4.11.3d and the volume fraction in element e is again rectangular.

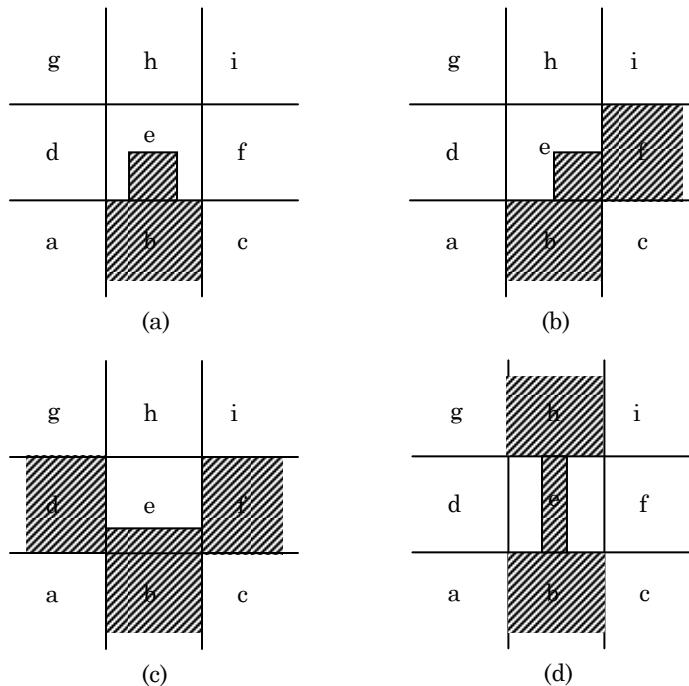


Figure 4.11.3: Examples of reconstructions for the fluid interface in a volume tracking algorithm.

Other combinations of partially filled and filled elements are possible but have not been shown since the general pattern of the algorithm is clear. The different patterns may be cataloged according to the number of filled or partially filled neighbors. The shape and location of the reconstructed surface is then defined from a table, while the values of the master coordinate lines are computed from the value of the volume fraction. This type of pattern definition would be required for each type of element used in a VT scheme and the number of possible patterns gets quite large for three-dimensional elements. Once the fluid interface has been constructed for each element, the surface areas needed for the flow rate in Eq. (4.11.19) and the advection step can be easily established. Reference [140] has a description of many of the details associated with the implementation of the VT method.

Though the various volume tracking methods have been employed with good success, the overall methodology is dominated by nonunique rules governing the interface reconstruction process. Another fixed mesh method that has recently become popular is the Level Set Method (LSM). This technique has a much stronger theoretical basis than the VT methods and shows great promise in resolving very complex free surface problems. The method is well described in the text by Sethian [141] and there are many applications in the literature.

The Level Set Method is related to the simple advection method outlined previously in conjunction with the concentration equation (4.11.15). Rather than describe a concentration function, the level set method usually begins with the kinematic condition that describes the location of the free surface. In any case, the equations are the same. From Eq. (1.10.4) the free surface is defined by the function $F(x_i, t)$ and the condition

$$\frac{\partial F}{\partial t} + v_j \frac{\partial F}{\partial x_j} = 0 \quad (4.11.20)$$

The difficulty with solving this equation as a concentration equation was the use of a function with a steep gradient to represent the free surface location. The level set method alters the F function to a smoother function that can be more easily represented on general meshes. The standard level set function is the signed distance function that is defined by

$$F(x_i, 0) = \begin{cases} +d, & x_i \in \Omega_f \\ 0, & x_i \in \Gamma \\ -d, & x_i \in \Omega_0 \end{cases} \quad (4.11.21)$$

where d is the distance from the interface Γ , Ω_f is the fluid region and Ω_0 is the unfilled region. A weighted residual, finite element method applied to Eq. (4.11.20) produces an equation set for the time advancement of the F function given the velocity field v_j . Using Eq. (4.11.21) as an initial condition for the function, the evolution of the free surface can be computed by tracking the zero level set of the function F . Since F is a relatively smooth function, dispersion and other computational errors should not degrade the accuracy of the front motion. Also, because F is a smooth function it is easy to take derivatives and therefore compute the unit normal vector to the free surface

$$\hat{\mathbf{n}} = \left. \frac{\nabla F}{|\nabla F|} \right|_{F=0} \quad (4.11.22)$$

and the curvature of the free surface

$$\kappa = \nabla \cdot \hat{\mathbf{n}} = \nabla \cdot \left(\frac{\nabla F}{|\nabla F|} \right) \Big|_{F=0} \quad (4.11.23)$$

Both of these quantities may be important in certain applications and they are not available from the volume tracking methods.

Two additional points that must be addressed when using a level set method are extensional velocities and re-initialization. In the advection equation for the level set it is implied that the velocity field is specified everywhere, i.e., for all the level sets, not just the zero level set. This is not usually the case in flow problems where the fluid is filling an otherwise empty cavity. Methods for defining a useful velocity away from the interface are described by Sethian [141] and others [142]. The problem of re-initialization comes from the use of nonuniform velocity fields. When F is evolved in time using Eq. (4.11.20) the initial distance function will not generally remain a distance function. This is particularly true when fluid velocity fields are used around the free surface (zero level set). The solution to this difficulty is to re-initialize the level set to a distance function based on the current location of the zero level set. A simple iterative procedure proposed by Sussman, et al. [143] produces a new signed distance function and allows the computation to proceed.

4.12 Turbulence

4.12.1 Preliminary Comments

In the previous sections, many of the essential aspects of finite element techniques for viscous incompressible flows were demonstrated. Even within this relatively narrow class of problems a number of important fluid mechanics topics were omitted. Here we will briefly consider an extension to the basic problem area and describe some aspects of turbulent flows. This type of problem is of major importance for industrial problems.

Turbulence is a highly complex physical phenomenon that is pervasive in flow problems of scientific and engineering concern. A simple, precise definition of turbulence is difficult, though the phenomenon is often associated with the ideas of randomness, disorder, and chaos. Following Hinze [144], we say that turbulence is defined as an “irregular flow condition showing random variations with respect to both time and space coordinates with discernible statistical properties.” Though a standard definition of turbulence may be elusive, a number of well-established characteristics of a turbulent flow can be listed. A turbulent flow is a

- highly nonlinear flow process,
- highly diffusive flow,
- three-dimensional flow,
- flow with multiple length and time scales, and
- time-dependent (stochastic) phenomenon with identifiable statistical properties.

Turbulence is one of the unsolved problems in physics, especially in the sense that universally applicable mathematical models of the phenomenon are not available. Despite these theoretical difficulties, the demand for numerical computation of

realistic (turbulent) flows remains high. Our interest here will, therefore, focus on the modeling and simulation of turbulence from an engineering point of view. This approach implies that the detailed resolution of a turbulent flow will be eliminated in favor of some type of averaged flow description. Turbulence effects will enter the flow description via a model that is typically based on a combination of theory and experiment.

In the following sections, the equations for the mean flow will be described followed by an outline of various types of turbulence models. Sections on finite element implementation and boundary conditions are also included.

4.12.2 Governing Equations

A majority of researchers accept the notion that, in principle, the Navier–Stokes equations are capable of fully describing a turbulent flow [145,146]. The natural question that follows this premise is if the Navier–Stokes equations are valid, why not solve them directly (via a numerical method) to obtain the needed turbulent solution? Recall from the brief introduction that turbulent flows are inherently unsteady and three-dimensional and contain a wide spectrum of length scales. To accurately simulate such a flow, the mesh resolution would have to be on the same order as the smallest length scale. Typically, the smallest eddies in a turbulent flow are $\approx O(0.001L)$ where L is the characteristic flow dimension. This translates into a three-dimensional grid with a minimum of $\approx 10^9$ mesh points. In addition, the flow must be resolved on the shortest relevant time scale, which is again associated with the smallest eddies. The computational demands of such a direct simulation of turbulence, even for simple geometries, is generally beyond the range of current computer resources. For industrially relevant problems and complex geometries, the direct simulation approach is clearly not possible with present technologies.

The standard alternative to the direct numerical simulation (DNS) approach involves the solution to some form of *averaged* Navier–Stokes equations. In most flow problems of interest it is the mean flow that is of most concern, with the turbulent fluctuations only being important in how they influence the mean flow evolution. By performing a suitable average on the instantaneous Navier–Stokes equations, a standard mean flow problem can be derived where the effects of the turbulence are relegated to a few terms that can be *modeled*. This approach forms the basis for most of the current computational work.

To outline this approach, let the instantaneous fluid velocity and pressure fields be expressed as the sum of a mean and fluctuating component. That is

$$v_i = V_i + v'_i \quad (4.12.1)$$

$$p = P + P' \quad (4.12.2)$$

where the upper case represents a mean quantity and the superscript prime denotes a fluctuation. Substituting these definitions into the equations of viscous incompressible flows of Section 4.1.2 and performing an ensemble average [144,146], one arrives at

$$\frac{\partial V_i}{\partial x_i} = 0 \quad (4.12.3)$$

$$\rho \left(\frac{\partial V_i}{\partial t} + V_j \frac{\partial V_i}{\partial x_j} \right) = - \frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} \left[\mu \left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i} \right) - \rho \overline{v'_i v'_j} \right] + \rho g_i \quad (4.12.4)$$

wherein the usual summation convention is implied. The horizontal overbar indicates an averaged quantity. Equations (4.12.3) and (4.12.4) describe the behavior of the mean fluid velocity and pressure fields and are very similar in form to the laminar equations discussed extensively in a previous section. In arriving at Eqs. (4.12.3) and (4.12.4), an ensemble average was employed, although space and time averaging could also be utilized. The various averages can be related through the ergodic hypothesis. The details and subtleties of averaging are beyond this text and can be found in many standard references [146–148].

The extra term that appears in Eq. (4.12.4) is often termed the Reynolds stress and represents the effects of the turbulent velocity fluctuations on the mean flow. Note that these second-order moments form the nine components of a second-order tensor, and the symmetry considerations reduce this to a total of six independent components.

Since no additional equations were added to those appearing in Eqs. (4.12.3) and (4.12.4), the appearance of the Reynolds stress unknowns leads to the well-known closure problem for turbulence. Through appropriate manipulation of the Navier–Stokes equations, six equations for the Reynolds stress variables may be derived. Unfortunately, these derived equations contain additional unknowns in the form of their higher-order moments. Similarly, equations derived for the third-order moments contain fourth-order moments. This problem of cascading unknowns is known as the closure problem. In order to achieve closure to the boundary value problem, the equation derivation must be halted at some level and a model introduced for the remaining unknowns. A number of turbulence models will be discussed in the next section.

4.12.3 General Turbulence Models

The completion of the mean flow boundary value problem described by Eqs. (4.12.3) and (4.12.4) requires that six additional equations for the Reynolds stresses be provided. This specification constitutes a turbulence model. There are an enormous variety of turbulence models, ranging in complexity from simple algebraic statements to descriptions involving multiple, nonlinear partial differential equations. Unfortunately, there is no universal method of classification for such models, which adds greatly to the confusion within the field. Here we follow the classification scheme by Ferziger [149,150], which groups turbulence models according to the following labels:

- Correlations
- Integral methods
- One point closure
- Two point closure
- Large eddy simulation
- Direct numerical simulation

In this section we will briefly outline some of the salient features of the models in each category. A subsequent section focuses on the one point closure methods which are the most prominent in current numerical work.

4.12.3.1 Correlations

The general usefulness of models based on correlations is very limited as they are available for only a few standard problem geometries that have been extensively studied by experimental methods. Though this is a relatively primitive approach, it is still used in many industrial problems. Correlations are typically developed to predict friction factors and heat transfer coefficients for various types of channel flows. Correlations generally play no role in field equation simulations.

4.12.3.2 Integral methods

Integral methods are useful when dealing with problems that have a parabolic nature, e.g., boundary layer flows. Generally, this approach utilizes a standard mean field equation that has been integrated over one coordinate to reduce the mathematical complexity of the problem. Empirical information is easily incorporated with this approach and leads to a fast and simple numerical procedure. However, these models lack generality and are limited again to those types of flows that have been extensively studied. This type of model still finds use in industry for repetitive design computations, but has largely been abandoned by the research community. Reviews of this type of turbulence modeling can be found in [146,151,152].

4.12.3.3 One-point closure

One point closure models are the most popular turbulence models, particularly for computational work. By one point closure it is meant that the correlations between the fluctuating velocity components, u'_j and u'_i , always employ values taken at the same physical location. Despite this restriction, the one point methods encompass a wide variety of successful models. Additional discussion of one point closure models is provided in Section 4.12.4.

4.12.3.4 Two-point closure

Two point closure models remove the restriction of the one point closure approach and explicitly introduce the effect of length scale. This approach is still in the development stage and has been limited in application to homogeneous and isotropic turbulence. An excellent introduction to this type of closure model can be found in [145] and [153].

4.12.3.5 Large eddy simulation

As the name implies, Large Eddy Simulation (LES) or Subgrid Scale Modeling (SGM) attempts to simulate the large scale motions of the flow and only model the very small scale eddies. From theory and experiment it is known that the smallest scale motions tend to be universal in their behavior and thus LES would seem to be an ideal approach for a broadly applicable turbulence model. The drawback to LES is that it is still relatively expensive because grid refinement remains a primary consideration. A good introduction to LES can be found in the early works of Yoshizawa [154], Rogallo and Moin [155] and the later texts by Sagaut [156] and John [157].

From this description of LES in terms of large and small scales, it should be recognized that the Variational Multiscale (VMS) method (see Section 4.8.4) is well suited to this application. Indeed, the early interest in VMS, besides its association

with stabilization, was directed toward improved LES models [83,84]. The VMS approach to turbulence is outlined in Section 4.12.6.

4.12.3.6 Direct numerical simulation (DNS)

This type of turbulence model was discussed previously in Section 4.12.2.

4.12.4 One-Point Closure Turbulence Models

As noted previously, the majority of computational work, especially for industrial applications, has relied on some form of the one point closure model. Even within this category there is a wide variety of possible models and various ways to catalog them. Shown in Figure 4.12.1 is a classification scheme that will be used in the following discussion. Of the two major branches shown in the figure, we will concentrate on the so-called eddy viscosity models. The Reynolds stress models (RSM), though generally more sophisticated than the eddy viscosity approach, lead to large systems of partial differential equations and a large number of empirical parameters. The computational burden of these models, coupled with a lack of certainty regarding the parameters, makes the RSM unattractive for present engineering applications.

Returning to the eddy viscosity models, it is important to note that this approach is based on one major assumption, the Boussinesq hypothesis. By analogy with the molecular diffusion of momentum, the Boussinesq hypothesis relates the turbulent momentum transport to the gradients of the mean velocity field. The Reynolds stresses in Eq. (4.12.4) are then expressed by

$$-\rho v_i^{\bar{v}} v_j' = \mu_T \left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i} \right) \quad (4.12.5)$$

where μ_T is the eddy viscosity. Unlike the molecular viscosity, μ , which is a fluid property, the eddy viscosity is a local property of the flow. When the definition in Eq. (4.12.5) is substituted into the momentum equation, then the equations for the mean flow become

$$\frac{\partial V_i}{\partial x_i} = 0 \quad (4.12.6)$$

$$\rho \left(\frac{\partial V_i}{\partial t} + V_j \frac{\partial V_i}{\partial x_j} \right) = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} \left[(\mu + \mu_T) \left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i} \right) \right] + \rho g_i \quad (4.12.7)$$

Once the form of the eddy viscosity is specified, then the mean flow can be solved in the same manner as a laminar flow since the equations are the same except for an augmented viscosity.

Though the turbulent flow problem has been reduced to a familiar set of partial differential equations, there remains the nontrivial task of specifying how the eddy viscosity varies with the flow field. Scaling arguments show that the eddy viscosity is proportional to a characteristic eddy velocity, v_e , and an eddy length, ℓ_e . That is

$$\mu_T \propto \rho v_e \ell_e \quad (4.12.8)$$

In general, it is easier to specify the variation of v_e and ℓ_e for a turbulent flow, and most turbulence models use these parameters or some closely related variable. From Figure 4.12.1 the type of turbulence model being used is determined by the number of equations used to specify the variation of the variables in Eq. (4.12.8).

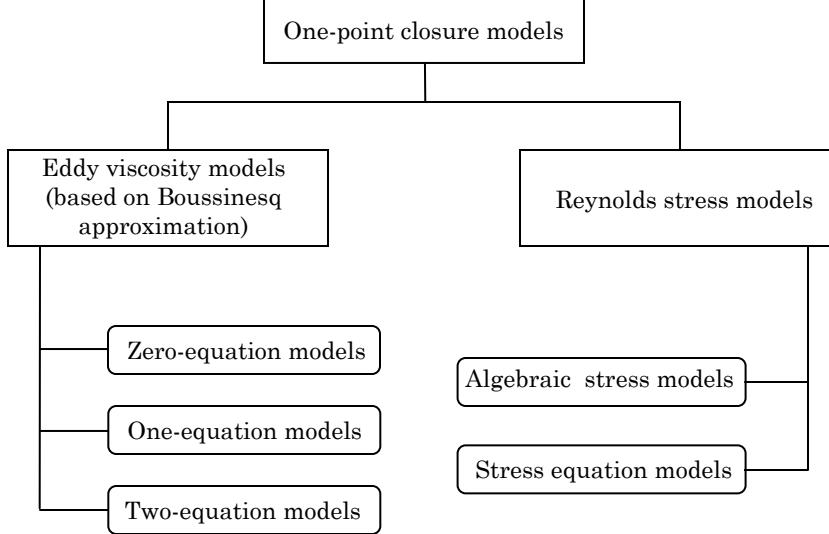


Figure 4.12.1: A classification scheme for one-point closure models.

4.12.4.1 Zero-equation models

These models calculate the eddy viscosity in Eq. (4.12.8) by an algebraic prescription of v_e and ℓ_e and therefore are the simplest of all the one point closure models. Most of these models are based on Prandtl's mixing length, which specifies ℓ_e to be the length scale across which turbulent mixing takes place. Prandtl gave the characteristic velocity, v_e , to be

$$v_e = \ell_e \left[\left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i} \right) \frac{\partial V_i}{\partial x_j} \right]^{1/2} \quad (4.12.9)$$

which when substituted into (4.12.8) yields

$$\mu_T = \rho \ell_e^2 \left[\left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i} \right) \frac{\partial V_i}{\partial x_j} \right]^{1/2} \quad (4.12.10)$$

For a number of geometrically simple flows, the variation of the mixing length, ℓ_e , is well known and can be evaluated via simple formulas. Equation (4.12.10) then allows μ_T to be derived and the turbulence model completed. Flows that are amenable to such treatment include pipe and channel flows, jets, wakes, and boundary layers. A compilation of useful mixing length formulas can be found in [158,159]. Though the above model is quite simple, good results can be obtained if the model is employed for the types of flows for which it is strictly intended.

4.12.4.2 One-equation model

The limitation of the zero-equation model to simple flows, and the need to simulate more complex geometries, has led naturally to the development of more sophisticated

turbulence models. As a first step in this direction the algebraic specification of v_e can be replaced with a more generally applicable transport equation. Since the characteristic velocity, v_e , is proportional to the square root of the turbulent kinetic energy, k , then

$$v_e \propto k^{1/2} \quad (4.12.11)$$

and from Eq. (4.12.8)

$$\mu_T \propto \rho k^{1/2} \ell_e \quad (4.12.12)$$

A partial differential equation for k can be derived from the Navier–Stokes equation and is given by

$$\rho \left(\frac{\partial k}{\partial t} + V_j \frac{\partial k}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(\frac{\mu_T}{\sigma_k} \frac{\partial k}{\partial x_j} \right) + \rho G - \rho \epsilon \quad (4.12.13)$$

where G is a generation term, ϵ is the turbulent dissipation, and σ_k is a constant.

Despite the apparent sophistication involving the evaluation of k , the model still relies on an algebraic specification of the mixing length. Like the zero-equation model, variations in ℓ_e are problem dependent and are well characterized for only a few types of flows. The model is marginally better than the zero-equation model but is not heavily used in numerical simulations.

4.12.4.3 Two-equation model

A natural evolution of the one-equation model involves the replacement of the algebraic relation for mixing length with a second transport equation. Dimensional arguments lead to the proportionality

$$\ell_e \propto \frac{k^{3/2}}{\epsilon} \quad (4.12.14)$$

where ϵ is the viscous dissipation. Substituting Eqs. (4.12.14) and (4.12.11) into Eq. (4.12.8) produces the proportionality

$$\mu_T \propto \rho \frac{k^2}{\epsilon}$$

or the Kolmogorov–Prandtl relation

$$\mu_T = C_\mu \rho \frac{k^2}{\epsilon} \quad (4.12.15)$$

which relates the eddy viscosity directly to the turbulence variables, k and ϵ . In the two-equation model, the turbulent kinetic energy is given by Eq. (4.12.13)

$$\rho \left(\frac{\partial k}{\partial t} + V_j \frac{\partial k}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(\frac{\mu_T}{\sigma_k} \frac{\partial k}{\partial x_j} \right) + \rho G - \rho \epsilon \quad (4.12.16)$$

and the turbulent dissipation is described by an equation of similar form

$$\rho \left(\frac{\partial \epsilon}{\partial t} + V_j \frac{\partial \epsilon}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(\frac{\mu_T}{\sigma_\epsilon} \frac{\partial \epsilon}{\partial x_j} \right) + \rho \frac{\epsilon}{k} (c_1 G - c_2 \epsilon) \quad (4.12.17)$$

where G is again a shear generation term and c_1 and c_2 are empirically derived constants as are σ_k and σ_g . Note that Eqs. (4.12.16) and (4.12.17) are obtained from manipulation of the instantaneous momentum equation. However, to provide closure, certain terms, such as G , are modeled by relating their exact form to some function of the mean flow.

The two-equation, $k-\epsilon$ model described by Eqs. (4.12.16) and (4.12.17) can be used in conjunction with the mean flow equations and the definition of μ_T given by Eq. (4.12.15), to arrive at a continuum description of turbulent flow. The equation set is highly nonlinear, with a strong coupling between the various transport equations. However, this additional mathematical and computational complexity is offset by a substantial increase in realism and reliability of the turbulence predictions. The $k-\epsilon$ model is far from universal and has a number of weaknesses, though it remains one of the most heavily used methods for flow simulation. A good discussion of this model can be found in [160] and [161].

4.12.5 Finite Element Modeling of Turbulence

Once a turbulence model has been selected for a particular application, it is important to consider how a numerical technique, such as the finite element method, would be used to produce useful flow simulations. In previous chapters, various solution methods for the mean flow equations such as Eqs. (4.12.6) and (4.12.7) were described in detail and need not be reconsidered here. However, it is appropriate to outline some of the added complications that must be addressed when a turbulence model is added to the mean flow system. To limit the discussion, only the zero-equation and two-equation, $k-\epsilon$, forms of the one-point closure models will be considered. These two approaches are the ones most commonly encountered in practical applications.

4.12.5.1 Zero-equation model

The finite element implementation of a zero-equation turbulence model is straightforward since the eddy viscosity is prescribed by a simple algebraic relation, such as Eq. (4.12.10). Since the eddy viscosity varies over the flow domain, the methods introduced earlier in this chapter for modeling variable coefficients may be utilized. From Eq. (4.12.10) it is apparent that the turbulent viscosity depends on velocity gradients, which is directly analogous to the behavior of inelastic, non-Newtonian fluids (see Chapter 6). For the best accuracy, the velocity gradients should be evaluated at element integration points. The dependence of μ_T on the mean field velocity produces another source of nonlinearity in the numerical problem, though this is not of major significance for most iterative solution algorithms.

The most difficult aspect of these models comes from the specification of the mixing length over the flow domain. These empirical formulas will vary from region to region as different types of flows (e.g., boundary layers, shear layers, jets) are encountered. Also, many of the ℓ_e functions depend on distance from a boundary, a dependence that may be awkward to specify easily and uniquely in a complex geometry. Despite these inconveniences the simplicity of the model makes it a popular and useful choice for modeling.

4.12.5.2 Two-equation model

The finite element implementation of a $k-\epsilon$ model is also conceptually straightforward since the turbulent transport equations (4.12.16) and (4.12.17) are of the familiar advection-diffusion type as will be discussed in Chapter 5. A weak form of the $k-\epsilon$ system can be readily constructed, and when coupled with finite element interpolations for the unknowns, a standard set of coupled ordinary differential equations can be defined. The discretized forms of Eqs. (4.12.16) and (4.12.17) will be highly nonlinear and strongly coupled to the mean flow equations. It is obvious that the size of the computational problem for a two equation turbulent simulation has increased substantially over its laminar counterpart. Not so obvious is the fact that the numerical difficulties have also increased significantly.

The major problems that occur with the $k-\epsilon$ model come from two sources: the nonlinear behavior of the partial differential equations and boundary conditions for the turbulent transport and mean field equations. It has been observed by a number of investigators that the dissipation equation may cause instabilities in the flow simulation that lead to a poor or nonconvergence of the numerical solution algorithm. Another manifestation of these stability problems is the prediction of negative values for both k and ϵ [160–165]. This nonphysical occurrence is sometimes attributed to the inaccurate modeling of the source terms for k and ϵ . The practical solution, in many cases, involves a clipping procedure in which negative values are replaced by small positive values [160].

The sensitive nature of the $k-\epsilon$ model leads to some practical recommendations for solution methods in turbulent flows. Generally, it has been found advantageous to employ Picard iteration methods as opposed to the higher-order Newton schemes. Though the convergence rate is reduced in this case, the overall stability of the algorithm is improved. Relaxation factors greater than 0.5 have also been observed to substantially enhance the solution process. Also related to the solution stability is the grading of the computational mesh. Computations that have abrupt changes in mesh density have been reported to show poor behavior in the k and ϵ variables and in some cases can lead to divergence of the solution process. In monitoring the convergence of a turbulent flow simulation it is noteworthy that the velocity and pressure fields converge first; convergence of the k and ϵ fields is very slow and may not be monotonic.

The topic of boundary conditions for the $k-\epsilon$ model and the modeling of turbulence close to solid boundaries is quite involved and will only be summarized here. The $k-\epsilon$ model described in Eqs. (4.12.16) and (4.12.17) is known as a high Reynolds number model, since its derivation was predicated on a single length scale associated with an unbounded, fully developed flow. Close to solid boundaries, the turbulent nature of the flow changes to a low Reynolds number model, where the small eddies associated with dissipation become important. Of course, very near the wall a thin viscous sublayer exists in which a turbulence model is no longer appropriate. From a computational viewpoint these near-wall regions are extremely challenging, since not only does the type of turbulence model vary with distance from the boundary, but all of the flow variables experience their greatest variation in this region. In most cases, attempts to solve the mean flow and $k-\epsilon$ equations all the way to the wall would be prohibitively expensive, especially for complex three-dimensional geometries.

An alternative to the above approach was developed at Imperial College [166,167] and relies on the use of universal models or wall laws to empirically estimate the flow behavior in the viscous sublayer and the buffer layer. The development of such a model allows the computational domain to be displaced from the wall and located in a fully turbulent region where the k - ϵ model is valid. Boundary conditions for the mean flow variables, k and ϵ , at the edge of the computational domain are provided by the wall laws. A wall law is typically a “universal” velocity profile that is derived from a combination of theory and experimental observation; standard profiles for k and ϵ near a boundary can also be derived. These parameterized profiles provide values of the mean field shear stress, k and ϵ that can be directly applied as boundary conditions to the computational domain. A good description of wall laws and modeling is available in [160,167]. The major disadvantage of this approach is that the computational domain must be offset from the wall by a distance that depends on the wall law and the flow field. The location of the domain may therefore have to be altered during or after the solution process. Despite this difficulty, the wall law approach is the only practical way to model turbulent flows in the vicinity of a boundary.

An improvement to the wall law formulation for finite element applications was developed in conjunction with the FIDAP code [168]. A standard representation of a universal velocity profile (wall law) consists of a linear profile for the viscous sublayer, a log-linear profile in the buffer region, and a logarithmic profile in the fully developed turbulent region. It is possible to develop an interpolation or shape function that represents this composite profile and therefore construct a special finite element that represents the wall law region of a turbulent flow. Though the problem with proper location of the edge of the special wall element remains, the development of such an element solves many other problems involving mesh generation and the specification of boundary conditions on complex surfaces. The precursor to this work, and some of the first work on specialized elements for fluid mechanics, can be found in [163] and [169].

Though the primary discussion has been related to boundary conditions at solid walls, it must be remembered that the k - ϵ equations require boundary values to be specified on all computational boundaries. Especially troublesome in this regard are k and ϵ values for inlet boundaries, since to a large extent these values will set the turbulence level for the entire flow. Haroutunian [160] provided a method for consistently generating these values.

4.12.6 Variational Multiscale (VMS) Turbulence Modeling

The previously discussed turbulence models focused on the averaged Navier–Stokes equations where the fluctuating component of the velocity v'_i required some form of model. The averaged velocity V_i was viewed as a large or computationally resolvable scale, and the small or unresolved scale was associated with v'_i . Despite the fact that turbulence is a multi-scale phenomena, no explicit separation of the large and small scales is accomplished with most of these models. Standard large eddy simulation (LES) methods filter the Navier–Stokes into large and small scales but then directly model the small scales in terms of the large scales [157,158]. This type of closure, with some scale separation, is often criticized for a variety of shortcomings [91] but primarily because of the incorrect energy transfer from the large scales.

The Variational Multiscale (VMS) approach does not depend on the a priori filtering of the Navier–Stokes equations and provides an explicit separation of the velocity scales. This was demonstrated for the linearized form of the Navier–Stokes equations in Section 4.8.4 where the method was associated with stabilization. Here we will consider the use of VMS for turbulent flows and its use with the fully nonlinear, incompressible flow equations. The VMS approach to turbulence modeling is not exclusive to finite element methods and has been used with higher-order methods (e.g., spectral), to study the basics of turbulence. However, VMS seems a very natural method to use with finite elements and will be outlined here with respect to a GFEM formulation.

As with other parts of this chapter, the discussion begins with the variational form of the incompressible Navier–Stokes equations as given, for example, by (4.8.26)–(4.8.27) or

$$\begin{aligned} \int_{\Omega} \left[\rho_0 w_i \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) + \frac{\partial w_i}{\partial x_j} \left(-P \delta_{ij} \right) + \frac{\partial w_i}{\partial x_j} \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] d\mathbf{x} \\ = \int_{\Omega} \rho_0 w_i f_i d\mathbf{x} + \oint_{\Gamma} w_i T_i ds \end{aligned} \quad (4.12.18)$$

$$- \int_{\Omega^e} Q \frac{\partial v_i}{\partial x_i} d\mathbf{x} = 0 \quad (4.12.19)$$

with standard boundary and initial conditions. The velocity and pressure weighting functions are w_i and Q , respectively, and the usual integration-by-parts has been executed on the stress divergence terms.

The VMS method for turbulence parallels the VMS approach for stabilization and permits the elimination of some details of the derivation, as these were covered in Section 4.8.4. Assume an additive decomposition of the solution fields and weight functions into coarse and fine scales. That is,

$$\begin{aligned} v_i(x_i, t) &= \bar{v}_i(x_i, t) + v'_i(x_i, t) \\ P(x_i, t) &= \bar{P}(x_i, t) + P'(x_i, t) \end{aligned} \quad (4.12.20)$$

$$\begin{aligned} w_i(x_i) &= \bar{w}_i(x_i) + w'_i(x_i) \\ Q(x_i) &= \bar{Q}(x_i) + Q'(x_i) \end{aligned} \quad (4.12.21)$$

where again the bar quantities denote the coarse (large) scales and the primed quantities are the fine (small) scales. Note that the bar and prime quantities are not the same as in the filtered (averaged) Navier–Stokes case [91]. As usual the weighting functions are not considered functions of time. The coarse and fine scale functions are linearly independent and when substituted into the variational statement produce the following coupled equations

Coarse Scale

$$\begin{aligned} \int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial \bar{v}_i}{\partial t} + \bar{v}_j \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} \left(-\bar{P} \delta_{ij} \right) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ + \int_{\Omega} \left[\rho_0 \bar{w}_i \left(\frac{\partial v'_i}{\partial t} + \bar{v}_j \frac{\partial v'_i}{\partial x_j} + v'_j \frac{\partial \bar{v}_i}{\partial x_j} + v'_j \frac{\partial v'_i}{\partial x_j} \right) + \frac{\partial \bar{w}_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ = \int_{\Omega} \rho_0 \bar{w}_i \bar{f}_i d\mathbf{x} + \oint_{\Gamma} \bar{w}_i \bar{T}_i ds \end{aligned} \quad (4.12.22)$$

$$-\int_{\Omega} \bar{Q} \frac{\partial \bar{v}_i}{\partial x_i} d\mathbf{x} - \int_{\Omega} \bar{Q} \frac{\partial v'_i}{\partial x_i} d\mathbf{x} = 0 \quad (4.12.23)$$

Fine Scale

$$\begin{aligned} & \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial v'_i}{\partial t} + v'_j \frac{\partial v'_i}{\partial x_j} + v'_j \frac{\partial \bar{v}_i}{\partial x_j} + \bar{v}_j \partial v'_i \partial x_j \right) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & + \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial \bar{v}_i}{\partial t} + \bar{v}_j \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial w'_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & = \int_{\Omega} \rho_0 w'_i \bar{f}_i d\mathbf{x} + \oint_{\Gamma} w'_i \bar{T}_i ds \end{aligned} \quad (4.12.24)$$

The fine scale pressure and pressure weighting are neglected; the fine scale is localized to the element such that $v'_i = 0$ on the element boundary. The first four terms of the coarse scale equation and the right-hand side are the standard Galerkin terms; the first term in (4.12.23) is also a Galerkin term. The fifth through seventh terms provide the fine scale influence on the coarse scale as in the stabilization methods. The sixth term is noteworthy for turbulence modeling as it contains the cross-stress and classic Reynolds' stress terms. In the fine scale equation, the first four terms are a weighted residual statement for the fine scale variables and the right-hand side of the equation is a weighting of the coarse scale residual. The second term in the residual is recognized as the Leonard stress. These equations were derived in the same manner as the stabilization VMS equations without a linearization of the advection term and by retaining the time dependence of the fine scale.

At this point a fine scale model is required. In the fine scale equation the term $w'_i \rho_0 v'_j \frac{\partial v'_i}{\partial x_j}$ is “responsible for the rapidly fluctuating behavior in the small scales” [83]. This term cannot be resolved on standard meshes and would require a DNS type of resolution. Here it is modeled by an eddy viscosity model of the Smagorinski form [157]. A term of the form

$$R' = 2\mu'_T D'_{ij} = \mu'_T \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \quad (4.12.25)$$

is added to the fine scale equation where the eddy viscosity, μ'_T , remains to be defined. With this addition to the fine scale, the fine scale equation becomes

Fine Scale

$$\begin{aligned} & \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial v'_i}{\partial t} + v'_j \frac{\partial v'_i}{\partial x_j} + v'_j \frac{\partial \bar{v}_i}{\partial x_j} + \bar{v}_j \partial v'_i \partial x_j \right) + \frac{\partial w'_i}{\partial x_j} \mu \left(\frac{\partial v'_i}{\partial x_j} + \frac{\partial v'_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & + \int_{\Omega} \left[\rho_0 w'_i \left(\frac{\partial \bar{v}_i}{\partial t} + \bar{v}_j \frac{\partial \bar{v}_i}{\partial x_j} \right) + \frac{\partial w'_i}{\partial x_j} (-\bar{P} \delta_{ij}) + \frac{\partial w'_i}{\partial x_j} (\mu + \mu'_T) \left(\frac{\partial \bar{v}_i}{\partial x_j} + \frac{\partial \bar{v}_j}{\partial x_i} \right) \right] d\mathbf{x} \\ & = \int_{\Omega} \rho_0 w'_i \bar{f}_i d\mathbf{x} + \oint_{\Gamma} w'_i \bar{T}_i ds \end{aligned} \quad (4.12.26)$$

The coarse scale equation remains the same as written in Eqs. (4.12.22) and (4.12.23), is exact and contains no models. The fine scale is the only equation with a model.

There are two standard choices for the Smagorinsky eddy viscosity that differ by the scales that are to be utilized. The choices are

$$\frac{1}{\rho_0} \mu'_T = (C'_S \Delta')^2 |\nabla^S u'| \quad (4.12.27)$$

$$\frac{1}{\rho_0} \mu'_T = (C'_S \Delta')^2 |\nabla^S \bar{u}| \quad (4.12.28)$$

where

$$\nabla^S u = \frac{1}{2} \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) ; |\nabla^S u| = (2 \nabla^S u \cdot 2 \nabla^S u)^{1/2} \quad (4.12.29)$$

Previous investigations have indicated the somewhat better performance of the eddy viscosity based on the small scale Eq. (4.12.27). A complex analysis by Hughes, et al. [83] has shown the relation between the parameters $C'_S \Delta'$ and the fine scale and the cut-off scale of the mesh. A simple method proposed by Koobus and Farhat [170] provides $C'_S = 0.1$ and sets Δ' to the local mesh size.

With the above equations and models the fine scale variables may be computed element-by-element using methods similar to those described in the stabilized VMS method. The coarse scale is then solved using the fine scale solution where needed in (4.8.22)–(4.8.23). The overall algorithm then provides the solution for a turbulent flow. The method has been used for an increasing number of studies of turbulent flows, see e.g. [171–174]. Note that there are still issues that need to be explored with respect to boundary conditions for turbulent flow using the VMS method [175].

4.13 Numerical Examples

4.13.1 Preliminary Comments

The true test of any numerical method is its performance in the solution of problems of engineering and scientific interest. In this section, a small sampling of flow problems solved using the finite element models discussed herein are presented. No attempt is made to examine the problems in depth but rather the intent is to illustrate the variety and, to some extent, the complexity of problems that can be solved using the finite element method.

The examples discussed here were solved using the computer codes FEM2DHT, NACHOS II [176], FIDAP [168], GOMA [133], and least-squares finite element code developed in [8,10]. Program FEM2DHT is a modified version of FEM2D from Reddy [1] and contains the reduced-integration penalty finite element model for the Stokes problem, NACHOS II contains the mixed finite element model for the Navier–Stokes equations, FIDAP contains both mixed and penalty finite element models of the Navier–Stokes equations, and GOMA contains both mixed and stabilized finite element models with an emphasis on moving mesh free surface problems. The objective of the first several examples, which deal with Stokes flows, is to evaluate the accuracy of the penalty and mixed finite element models in the light of available analytical or numerical results and to illustrate the effect of the penalty parameter on the accuracy of the solutions. Following the linear problems, several benchmark problems using the Navier–Stokes equations are presented. For additional examples, the reader may consult papers cited in the body of this chapter.

4.13.2 Fluid Squeezed between Parallel Plates

Consider the slow flow of a viscous incompressible material squeezed between two long parallel plates (see Figure 4.13.1a). When the length of the plates is very large compared to both the width and the distance between the plates, we have a case of plane flow. Although this is a moving boundary problem, we wish to determine the velocity and pressure fields for a fixed distance between the plates, assuming that a state of plane flow exists. An approximate (analytical) solution to this two-dimensional problem is provided by Nadai [177](also see [1,2]).

Let V_0 be the velocity with which the two plates move toward each other (i.e., squeezing out the fluid), and let $2b$ and $2a$ denote, respectively, the distance between and the width of the plates (see Figure 4.13.1a). Due to the biaxial symmetry present in the problem, it suffices to model only a quadrant of the domain. A 5×3 nonuniform mesh of nine-node quadratic elements is used in the mixed model, and a 10×6 mesh of the four-node linear elements and 5×3 mesh of nine-node quadratic elements are used in the penalty model (see Figure 4.13.1b). Nonuniform meshes, with smaller elements near the free surface (i.e., at $x = a$), are used to approximate accurately the singularity in the shear stress at the point $(a, b) = (6, 2)$. The mesh used for the penalty model has exactly the same number of nodes as the mesh used for the mixed model. There are no specified nonzero secondary variables in the problem. The computer program FEM2D (penalty model) from Reddy [1] is used to analyze the problem. The reduced integration rule suggested earlier for the evaluation of penalty terms is used.

The velocities $v_x(x, 0)$ obtained with the two models compare well with the analytical solution, as shown in Table 4.13.1. The nine-node element gives very good results for both the penalty and mixed models. The influence of the penalty parameter on the accuracy of the solution is clear from the results. Whether the element is linear or quadratic, it is necessary to use a large value of the penalty parameter, γ . For every computer there is an upper limit on the value of γ beyond which the contribution from the viscous terms will not be recognized in comparison to the penalty terms (and the coefficient matrix, being singular, is not invertible).

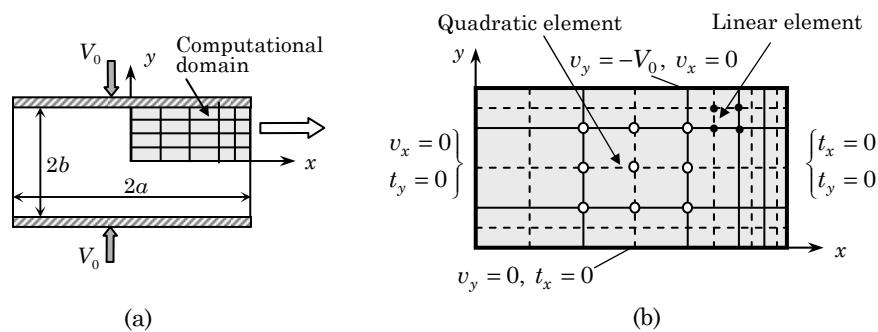


Figure 4.13.1: Domain and mesh for fluid squeezed between parallel plates.

Table 4.13.1: Comparison of finite element solutions $v_x(x, 0)$ with the analytical solution for fluid squeezed between plates (for $y = 0$).

x	$\gamma = 1.0$		$\gamma = 100$		$\gamma = 10^8$		Mixed model	Series solution
	4-node	9-node [†]	4-node	9-node	4-node	9-node		
1.00	0.0303	0.0310	0.6563	0.6513	0.7576	0.7605	0.7497	0.7500
2.00	0.0677	0.0691	1.3165	1.3062	1.5135	1.4992	1.5031	1.5000
3.00	0.1213	0.1233	1.9911	1.9769	2.2756	2.2557	2.2561	2.2500
4.00	0.2040	0.2061	2.6960	2.6730	3.0541	3.0238	3.0203	3.0000
4.50	0.2611	0.2631	3.0718	3.0463	3.4648	3.4307	3.4292	3.3750
5.00	0.3297	0.3310	3.4347	3.3956	3.8517	3.8029	3.8165	3.7500
5.25	0.3674	0.3684	3.6120	3.5732	4.0441	3.9944	3.9893	3.9375
5.50	0.4060	0.4064	3.7388	3.6874	4.1712	4.1085	4.1204	4.1250
5.75	0.4438	0.4443	3.8316	3.7924	4.2654	4.2160	4.2058	4.3125
6.00	0.4793	0.4797	3.8362	3.7862	4.2549	4.1937	4.2364	4.5000

[†] The 3×3 Gauss rule for non-penalty terms and the 2×2 Gauss rule for penalty terms are used for quadratic elements.

Next, a 12×8 mesh of linear elements and a 6×4 mesh of quadratic elements were used to evaluate the relative accuracies of the rectangular and triangular elements for the penalty model (meshes are not shown). The 12×8 mesh of linear triangular elements with full integration of \mathbf{K}^1 and \mathbf{K}^2 ($\mathbf{K} = \mu\mathbf{K}^1 + \gamma\mathbf{K}^2$) and selective integration (i.e., full integration of \mathbf{K}^1 and reduced integration of \mathbf{K}^2) both give the same results for the velocity field. However, in both cases, erroneous results for pressure and stresses are obtained. The 6×4 mesh of quadratic triangular elements with full and selective integrations give the same velocity fields, while the stresses and pressure are predicted to be the same at the same quadrature points. Both the 12×8 mesh of linear rectangular elements and the 6×4 mesh of nine-node rectangular elements give good results for velocities, pressure, and stresses. The solution accuracy can be increased by using refined meshes.

Figure 4.13.2 contains plots of the velocity $v_x(x, y)$ for $x = 4$ and 6 , and Figure 4.13.3 contains plots of pressure $P(x, y)$, for $y = \text{constant}$, computed using the six-node quadratic triangular (T6) and nine-node rectangular (R9) elements. The finite element solutions are compared with the analytical solutions of Nadai [177]. The pressure in the penalty model was computed using Eq. (4.3.15) with the 2×2 Gauss rule for the quadratic rectangular element and the one-point formula for the quadratic triangular element. If the pressure (see Figure 4.13.3) in the penalty model were computed using the full quadrature rule for rectangular elements, we would obtain erroneous values. The linear triangular element with full as well as reduced integrations gives unstable pressures, while the quadratic triangular element with one- or two-point rules yields good results. In general, the same quadrature rule as that used for the evaluation of the penalty terms in the coefficient matrix must be used to evaluate the pressure, and one should avoid using the linear triangular element in the penalty finite element model.

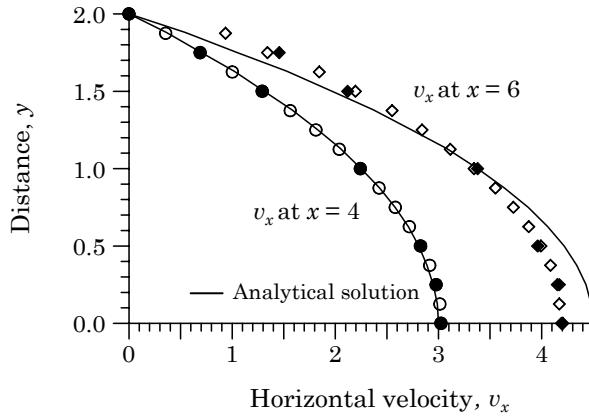


Figure 4.13.2: Velocity fields for fluid squeezed between parallel plates.

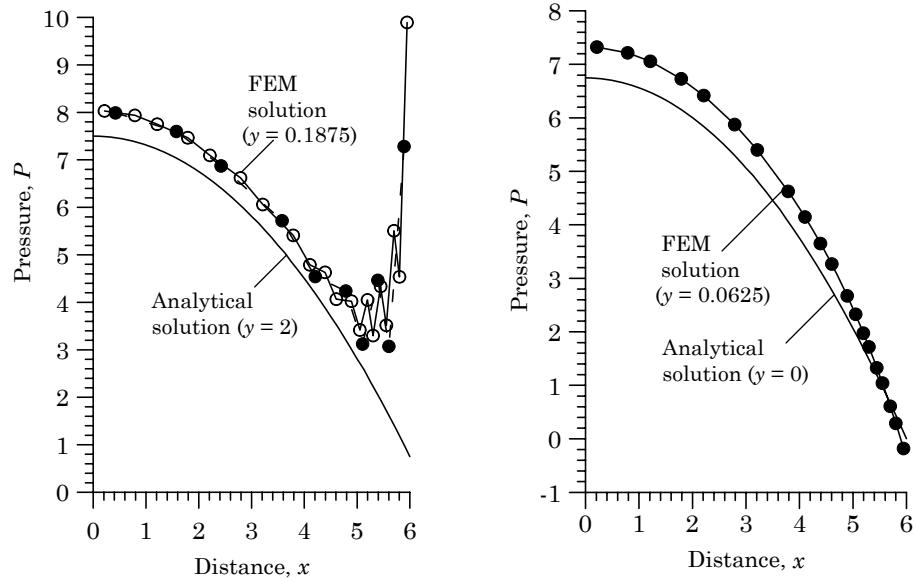


Figure 4.13.3: Pressures for fluid squeezed between parallel plates.

4.13.3 Flow of a Viscous Lubricant in a Slider Bearing

The slider (or slipper) bearing consists of a short sliding pad moving at a velocity $v_x(x, 0) = V_0$ relative to a stationary pad inclined at a small angle with respect to the stationary pad, and the small gap between the two pads is filled with a lubricant (see Figure 4.13.4a). Since the ends of the bearing are generally open, the pressure there is atmospheric, P_0 . If the upper pad is parallel to the base plate, the pressure everywhere in the gap must be atmospheric (because dP/dx is a constant for flow between parallel plates), and the bearing cannot support any transverse load. If the upper pad is inclined to the base pad, a pressure distribution (in general, a function of x and y) is set up in the gap. For large values of V_0 , the pressure generated can be of sufficient magnitude to support heavy loads normal to the base pad.

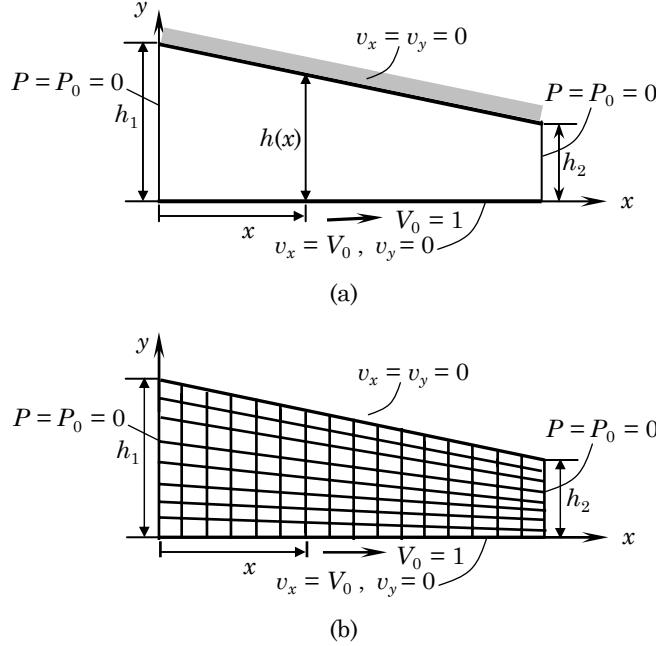


Figure 4.13.4: (a) Geometry, boundary conditions, and (b) finite element mesh (Mesh 1) for slider bearing.

Since the width of the gap and the angle of inclination are generally small, it is assumed that the pressure is not a function of y in developing an analytical solution to the problem. Assuming a two-dimensional flow and a small angle of inclination, and neglecting the normal stress gradient (in comparison with the shear stress gradient), the equations governing the flow of the lubricant between the pads can be solved to give (see Schlichting [178] and Reddy [1])

$$v_x = \left(V_0 - \frac{1}{2\mu} h^2 \frac{dP}{dx} \frac{y}{h} \right) \left(1 - \frac{y}{h} \right), \quad P = \frac{6\mu V_0 L (h_1 - h)(h - h_2)}{h^2(h_2^2 - h_1^2)} \quad (4.13.1a)$$

$$\sigma_{xy} = \mu \frac{\partial v_x}{\partial y} = \frac{dP}{dx} \left(y - \frac{h}{2} \right) - \mu \frac{V_0}{h}, \quad h(x) = h_2 + \frac{h_1 - h_2}{L} x \quad (4.13.1b)$$

In our computations, we choose

$$h_2 = 2h_1 = 8 \times 10^{-4} \text{ ft}, \quad L = 0.36 \text{ ft}, \quad \mu = 8 \times 10^{-4} \text{ lb/ft}^2, \quad V_0 = 30 \text{ ft} \quad (4.13.1c)$$

First, it should be pointed out that the assumption concerning the pressure not being a function of y is not necessary in the finite element analysis. We use a mesh (Mesh 1) of 18×8 linear quadrilateral elements to analyze the problem. The mesh and boundary conditions are shown in Figure 4.13.4b. The penalty parameter is chosen to be $\gamma = \mu \times 10^8$. Figure 4.13.5 contains plots of the horizontal velocity v_x at $x = 0$ ft, $x = 0.18$ ft, and $x = 0.36$ ft. Figure 4.13.6 contains plots of pressure and shear stress as a function of x at $y = 0$. The finite element solutions for the pressure and shear stress were computed at the center of the first row of elements along the moving block. The results are in good agreement with the analytical solutions, validating the assumptions made in the development of the analytical solution.

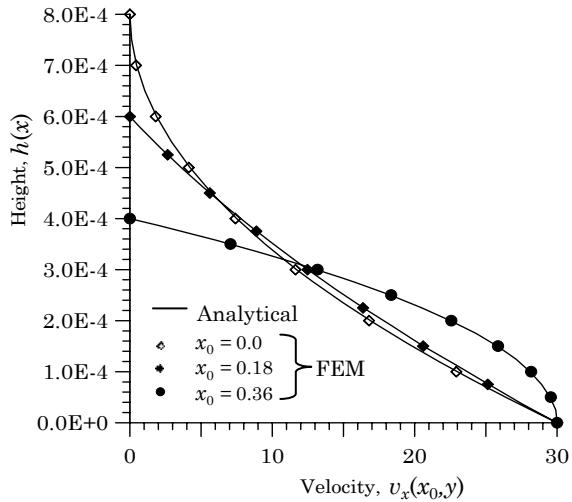


Figure 4.13.5: Pressures obtained with Mesh 1 for slider bearing.

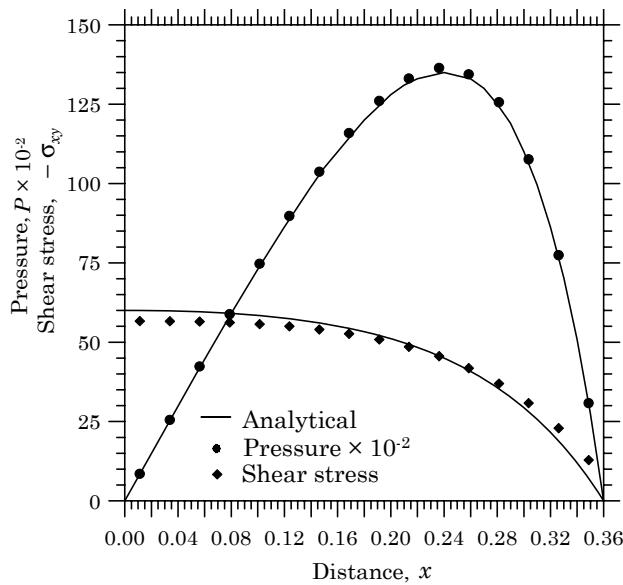


Figure 4.13.6: Velocity field, pressure, and shear stress for slider bearing.

4.13.4 Wall-Driven 2-D Cavity Flow

Consider the laminar flow of a viscous incompressible fluid in a square cavity bounded by three fixed walls and a lid moving at a constant velocity V_0 in its own plane (see Figure 4.13.7). The third dimension is assumed to be long enough to have a plane flow. Singularities exist at each corner where the moving lid meets a fixed wall. This example is one which has been extensively studied by analytical,

numerical, and experimental methods (see [179–186], among others). In solving this problem, the mesh used should be such that the boundary layer thickness is resolved. The boundary layer thickness is of the order of $Re^{-\frac{1}{2}}$, where $Re = \rho V_0 L / \mu$ and $L = 1$ is the cavity dimension. Thus the mesh should be graded to have smaller elements near the walls.

Figure 4.13.8 shows the horizontal velocity component along the vertical centerline of the cavity, i.e., plots of $v_x(0.5, y)$ vs. y , for $Re = 1, 100, 400$, and 10^3 . The results were obtained (see Reddy [185]) using a nonuniform, 14×14 mesh of linear rectangular elements. Figure 4.13.9 contains the plots of $v_x(0.5, y)$ vs. y for $Re = 400$ for various meshes. The effect of mesh refinement on the accuracy of the solution is clear.

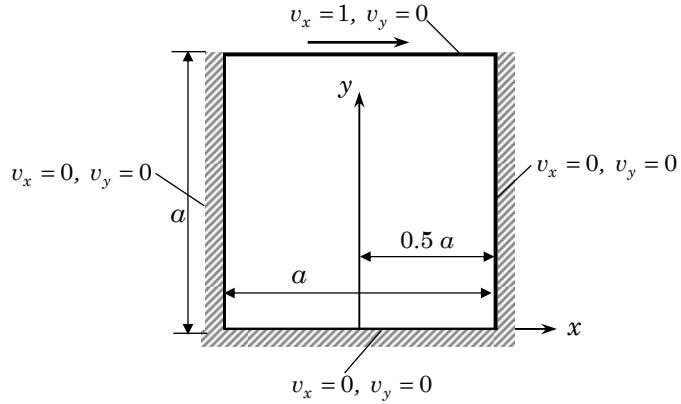


Figure 4.13.7: Geometry, boundary conditions, and finite element mesh for a wall-driven square cavity.

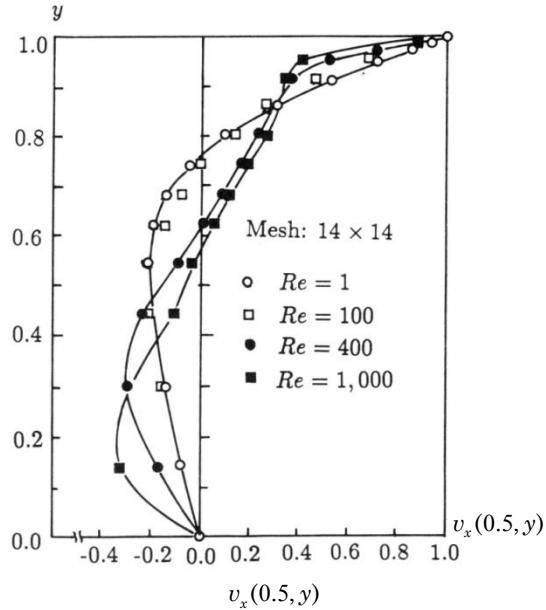


Figure 4.13.8: Horizontal velocity along the vertical centerline of the cavity for various Reynolds numbers (Mesh: 14×14).

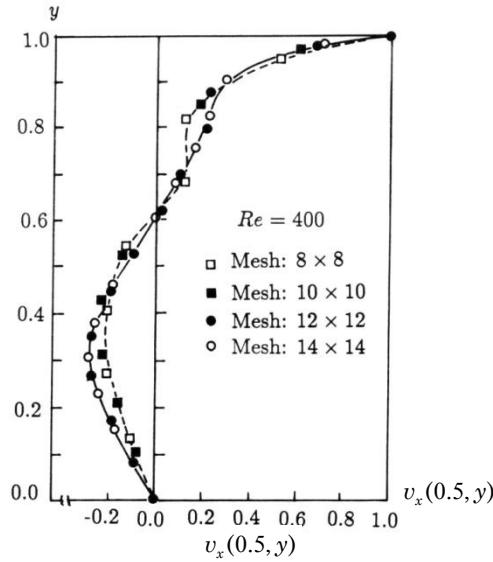


Figure 4.13.9: Horizontal velocity along the vertical centerline of the cavity for various meshes ($Re = 400$).

4.13.5 Wall-Driven 3-D Cavity Flow

The problem considered here is that of a three-dimensional cubical cavity of unit dimension [184,185,187–189]. The motion of an incompressible viscous fluid in the cavity is induced by the motion of the lid (see Figure 4.13.10). All other walls of the cavity are assumed to be stationary.

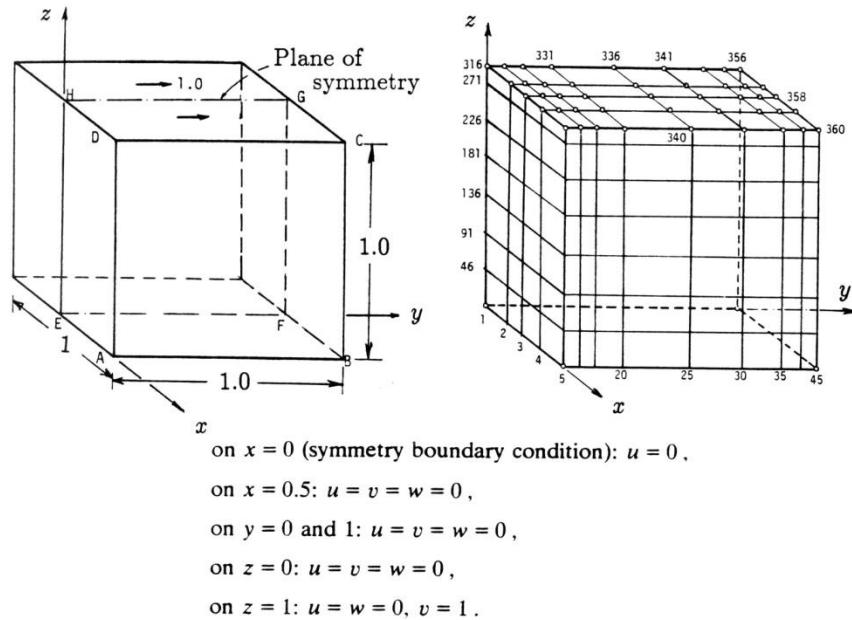


Figure 4.13.10: Geometry, mesh, and boundary conditions for a wall-driven cubical cavity.

Using the symmetry about the $x = 0$ plane, only half of the domain is modeled using a $4 \times 8 \times 7$ mesh of trilinear (i.e., eight-node brick) elements (see Reddy [189]). The problem is analyzed for $Re = 100$ and 400. Figures 4.13.11 and 4.13.12 contains the plots of velocity components in the midplane of the cavity. The present penalty finite element solution is compared with the finite difference solutions of Dennis, et al. [187] and Agarwal [188]. Dennis, et al. [187] used a second-order accurate finite difference method, whereas Agarwal [188] used a third-order accurate finite difference scheme. Both used very refined meshes. The present results agree well qualitatively with the results of Agarwal [188], although a very crude mesh was used in the finite element analysis.

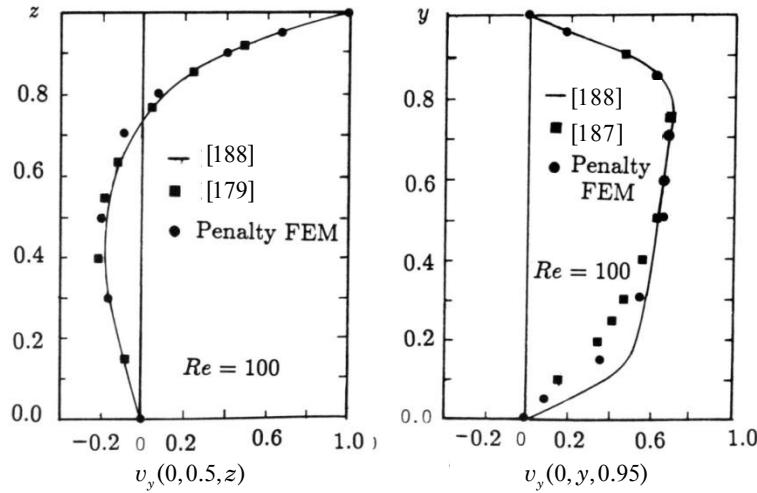


Figure 4.13.11: Velocity fields for a wall-driven cubical cavity, $Re = 100$.

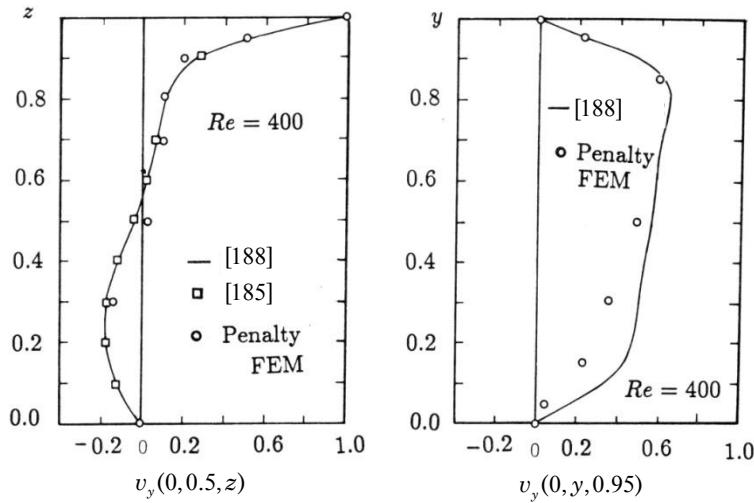


Figure 4.13.12: Velocity fields for a wall-driven cubical cavity, $Re = 400$.

4.13.6 Evaluation of the EBE Iterative Solvers

Here we use the wall-driven cavity problem to evaluate the EBE methods discussed in Appendix B and [34]. Four nonuniform meshes, Mesh 1: $5 \times 10 \times 10$, Mesh 2: $6 \times 12 \times 12$, Mesh 3: $7 \times 14 \times 14$, and Mesh 4: $8 \times 16 \times 16$ of 8-node brick elements were used. The pressure at $(0.5, 0.5, 0.0)$ is taken to be equal to zero, and it is used as an essential boundary condition for the pressure calculation according to Eq. (4.5.18). The problem was analyzed for $Re = 10$ and 100. For iterative solvers the convergence criterion for velocity and pressure calculations was taken to be $\epsilon_{vel} = 10^{-5}$ and $\epsilon_{pr} = 10^{-4}$ respectively.

It is observed that the convergence rate of the GMRES solver showed no oscillatory behavior but was slow compared to the ORTHOMIN and the ORTHORES solvers. The number of iterations required to arrive at a converged solution for $Re = 10$ for each solver are given in Table 4.13.2. The number in the parenthesis denotes the number of iterations required to solve the system of equations for each nonlinear iteration. The numbers in parenthesis in the second row for each mesh represent the number of iterations required for pressure calculations. The frontal solver took 3 iterations for each mesh. From Table 4.13.2 it is clear that the number of iterations required by the GMRES algorithm is nearly 3.5 times more than those required by the ORTHOMIN or the ORTHORES solvers for the same error tolerances. This reflects in the CPU time required to obtain a converged solution.

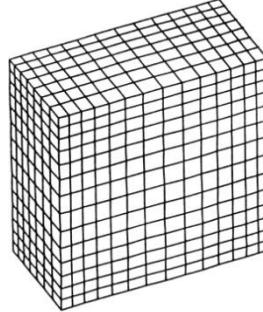


Figure 4.13.13: Finite element mesh (Mesh 3) used for the wall-driven cubical cavity.

Table 4.13.2: Number of iterations for different meshes and solvers.

Mesh	Elements	Equations	GMRES	ORTHOMIN	ORTHOLES
1	500	2,178	$2(1,580+85)$ $(19+5)$	$3(313+135+1)$ $(19+9+1)$	$3(313+137+1)$ $(19+9+1)$
2	864	3,549	$2(2,472+137)$ $(24+6)$	$3(462+243+1)$ $(24+8+1)$	$3(463+245+1)$ $(24+8+1)$
3	1,372	5,400	$2(3,383+165)$ $(35+6)$	$3(600+267+1)$ $(29+9+1)$	$3(600+269+1)$ $(29+9+1)$
4	2,048	7,803	$2(3,811+136)$ $(44+6)$	$3(789+316+1)$ $(34+10+1)$	$3(790+317+1)$ $(34+10+1)$

Figure 4.13.14 shows the CPU time requirements as a function of the number of elements for different solvers. It is clear that the frontal solver requires more CPU time. This is due to the fact that it is a direct solver and performs an elimination operation which is computationally intensive. Also the convergence of iterative solvers depends to some extent on the initial guess. Once the solution for the linear problem (Stokes) is obtained, it is used as the initial guess for the second iteration and thereon. This results in faster convergence from second the iteration onwards for iterative solvers, as shown in the Table 4.13.2. However, this is not true for the direct solvers because they require the same amount of CPU time for each iteration. Hence the CPU time requirements for direct solvers are enormous. The CPU times for ORTHOMIN were slightly less than those of ORTHORES. This is due to the fact that the GMRES algorithm needed more iterations compared to the ORTHOMIN and the ORTHORES algorithms. A similar trend was observed by [190] for the solution of convection/diffusion problems.

The storage requirements in millions of words for all four solvers for the four meshes are given in Table 4.13.3. From this table it is clear that the storage requirements for ORTHOMIN and ORTHORES are less than those for the GMRES and frontal solvers. The frontal solver requires a very large memory because the equations are stored before back substitution. The length of the equations depends on the front width. The front width for each of the meshes is given in parenthesis in Table 4.13.3.

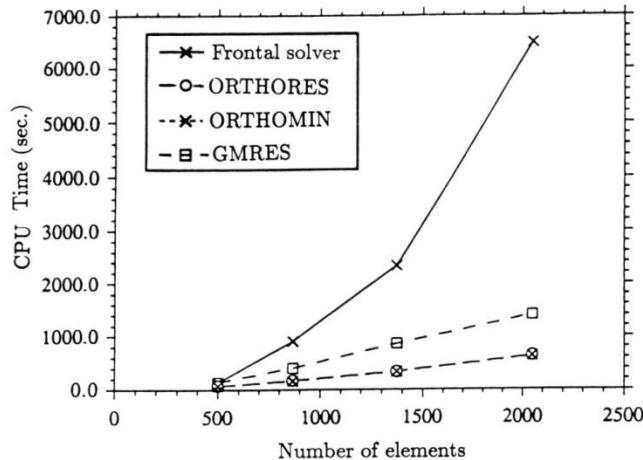


Figure 4.13.14: Comparison of CPU times for different solvers.

Table 4.13.3: Memory requirements (in millions of words) for various methods.

Mesh	Elements	Equations	Frontal	GMRES	ORTHOMIN	ORTHOLES
1	500	2,178	2.702 (270)	0.822	0.715	0.719
2	864	3,549	3.740 (354)	1.257	1.226	1.233
3	1,372	5,400	7.227 (450)	2.206	1.926	1.947
4	2,048	7,803	12.146 (558)	3.269	2.878	2.893

The solution for $Re = 100$ was obtained incrementally by using the flow field from the $Re = 10$ solution as the initial solution. The CPU times are given in Table 4.13.4 for Mesh 3 calculations. The difference in the CPU times for the $Re = 100$ and $Re = 10$ solutions gives the time used to obtain the $Re = 100$ solution. From Table 4.13.4 it is clear that the iterative solvers converge much faster and need less CPU time because of a good estimate of the initial guess for the next Reynolds number flow field calculation. It is clear that both the ORTHOMIN and the ORTHORES solvers with an element-by-element data structure are superior to the GMRES and frontal solvers.

Table 4.13.4: Comparison of CPU times (in seconds) taken by various methods.

S. No.	Re	Frontal	GMRES	ORTHOMIN	ORTHOLES
1	10	2,345	860	338	341
2	$10 \& 10^2$	5,489	1,046	619	626

4.13.7 Backward Facing Step

The 2-D laminar flow over a backward facing step is a benchmark problem for testing of outflow boundary conditions [118,119]. The geometry, boundary conditions, and a typical mesh are shown in Figure 4.13.15. The problem was solved for $Re = V_{avg}H/\nu = 800$ using zeroth-order continuation and Newton's methods. Meshes 6×120 to 40×800 were used to ensure mesh independence of the solution. The velocity field is approximated using the biquadratic interpolation functions, and linear, discontinuous, pressure approximation is used.

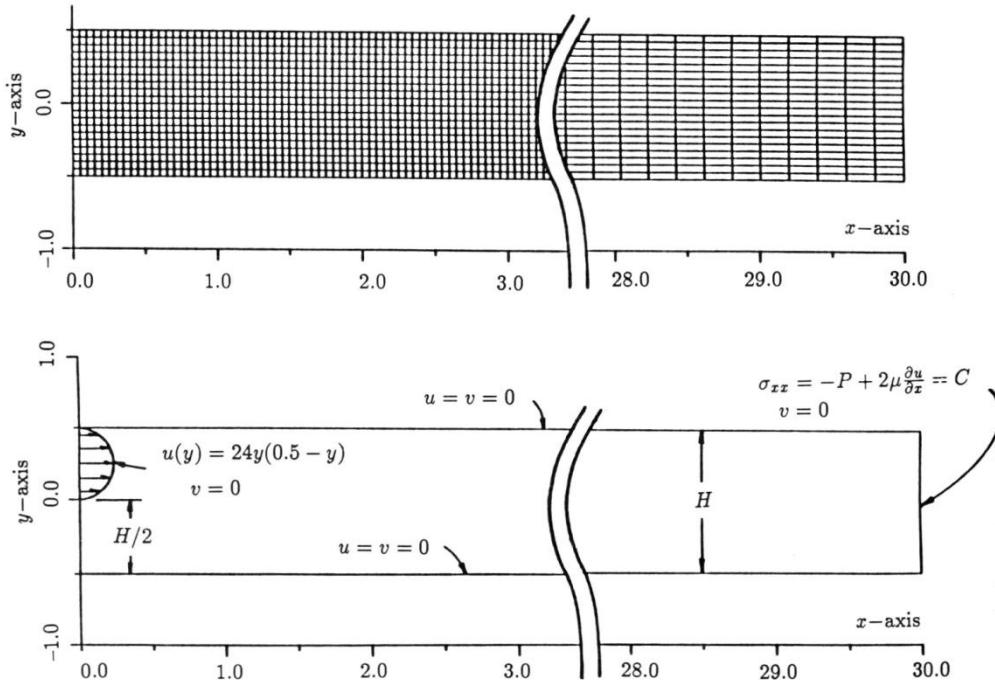


Figure 4.13.15: Schematic and mesh for backward facing step problem.

Figure 4.13.16 contains contour plots of the stream function, pressure, and vorticity obtained on a grid that extended 60 step heights downstream. Two recirculation regions are predicted with their position and intensity corresponding closely to previous numerical solutions. Other simulations were performed on computational domains that were truncated at 30 and 14 step heights downstream. In each case, the traction boundary condition shown in Figure 4.13.15 was specified at outflow.

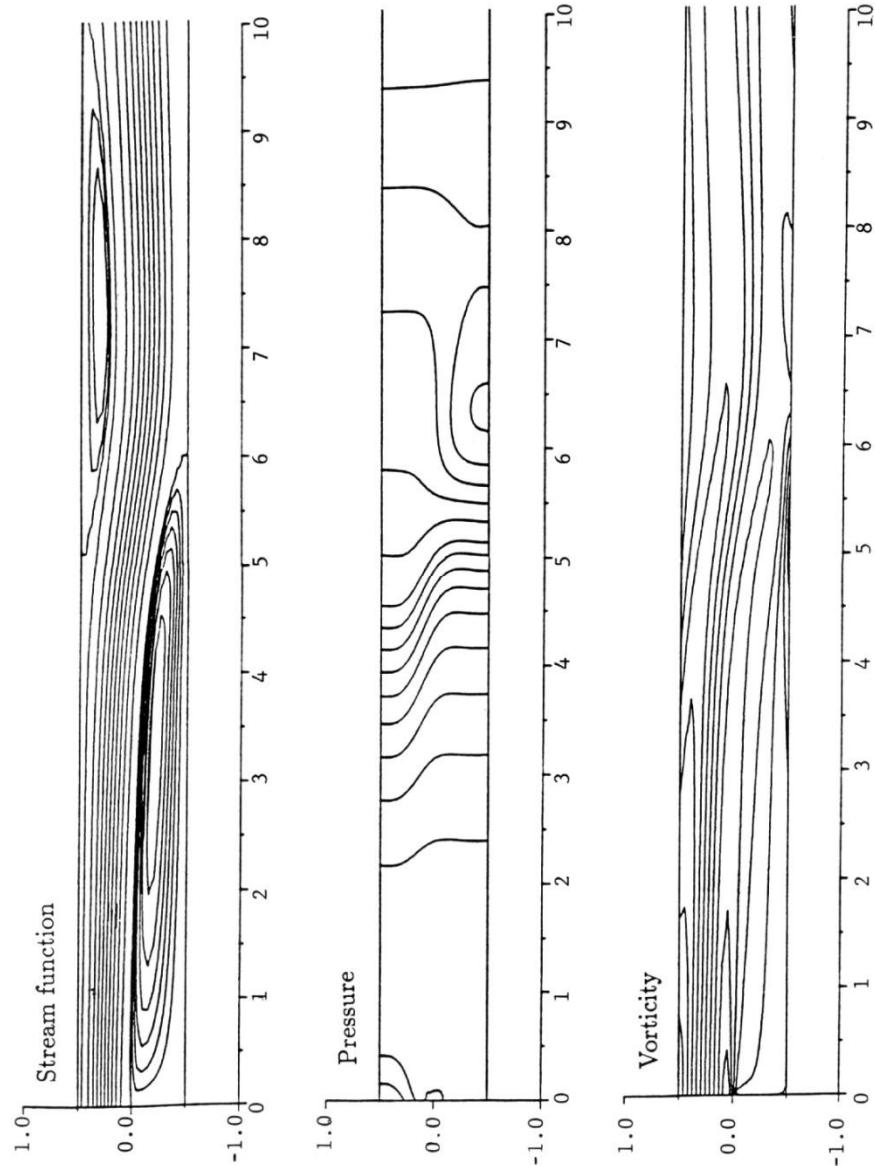


Figure 4.13.16: Contour plots for flow over a backward step; long computational domain.

Figure 4.13.17 shows a streamline plot of a result for the shortest domain; the comparison with the long domain result in Figure 4.13.16 is very good, especially considering that the upper wall recirculation zone is truncated by the outflow boundary. A comparison of axial velocities at this location between the solutions obtained with the long and short domains indicates less than 10% difference. This demonstrates the utility and accuracy of the traction boundary condition for use in short domain internal flow problems.

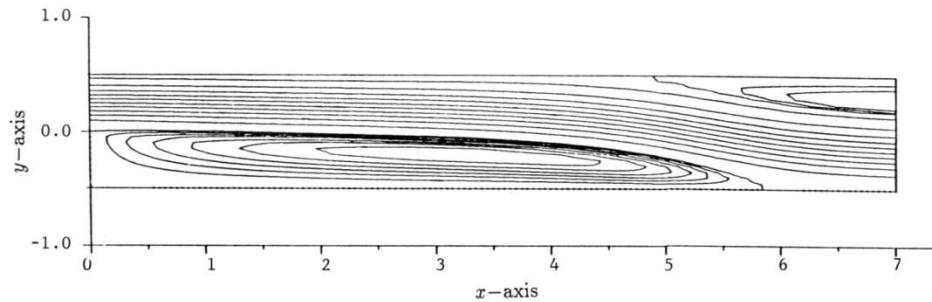


Figure 4.13.17: Contour plots for flow over a backward step; short computational domain.

4.13.8 Flow Past a Submarine

As an example of a large, three-dimensional viscous flow of engineering relevance we consider the flow past an axisymmetric submarine body with an attached fairwater or sail. This simulation was part of a benchmark exercise and consisted of predicting the steady flow past the submarine shape for $Re = 1.2 \times 10^7$. Reference [192] gives details on the entire simulation program.

Figure 4.13.18a contains several views of the surface mesh and surrounding finite element grid. Clockwise from upper left, Figure 4.13.18a illustrates the surface mesh for the complete body and enlarged views of the bow, the hull/fairwater juncture, and the stern tip, respectively. The mesh contained approximately 52,000 eight-node, hexahedral trilinear velocity elements in which the pressure was constant within an element. Figure 4.13.18b contains the three-dimensional finite element grid used for the numerical computation of steady, turbulent flow ($Re = 1.2 \times 10^7$) past the axisymmetric hull with attached fairwater. This mesh contains 57,523 nodes and 52,295 trilinear brick elements. The lower plot illustrates the relative position of the submarine within the computational domain. At the Reynolds number of interest the flow is clearly turbulent and the simulation employed a standard $k-\epsilon$ turbulence model. Wall functions were used to model the viscous sublayer along solid boundaries and provide boundary conditions for the k and ϵ variables. The equations were solved using a pressure correction method outlined in Section 4.6. The individual momentum equations, pressure Poisson equation, and $k-\epsilon$ equations were solved sequentially using an early version of the segregated solver developed for the FIDAP code [168]. Discussion of the segregated solver was presented in Section 4.6 and will be expanded in Section 5.7.3.

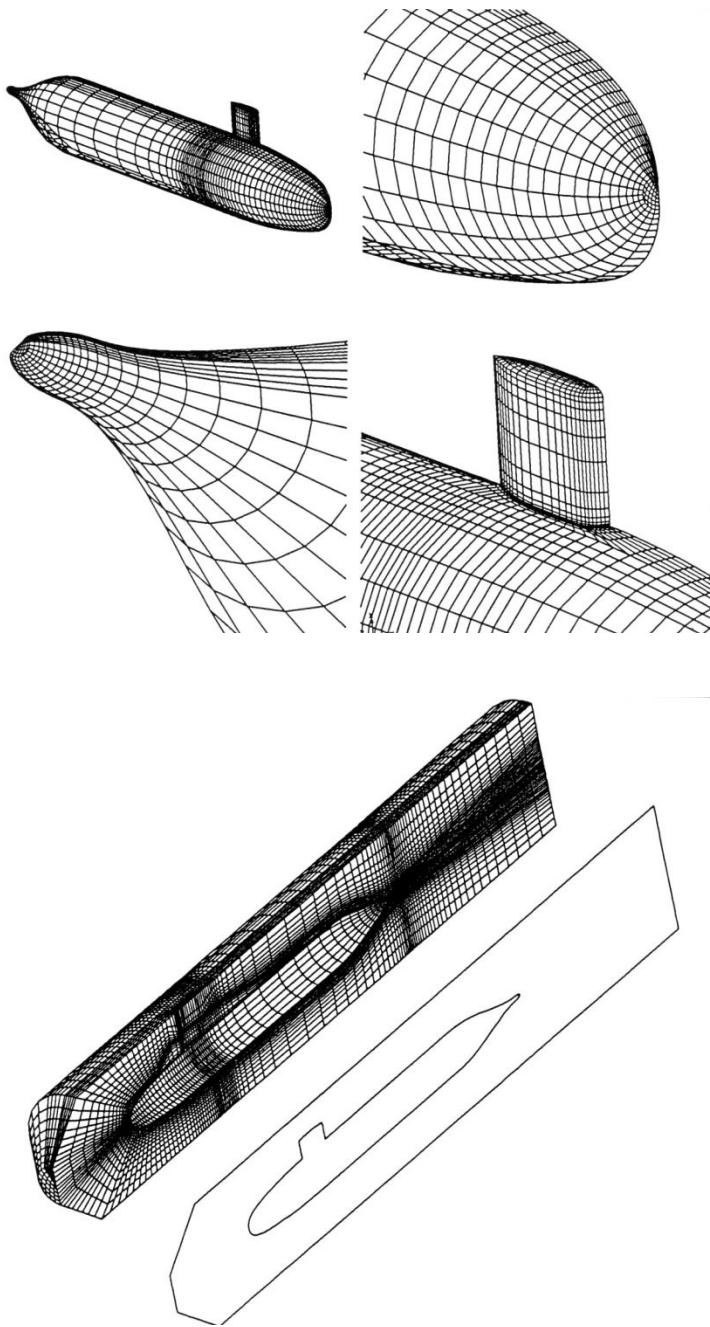


Figure 4.13.18: (a) Finite element discretization for the surface of the axisymmetric hull with attached fairwater. (b) Three-dimensional finite element grid used for the numerical computation of steady, turbulent flow ($Re = 1.2 \times 10^7$) past the axisymmetric hull with attached fairwater.

Pressure results for the submarine flow at zero angle of attack are shown as surface contours in Figure 4.13.19. A comparison of this result with an inviscid, potential flow solution showed excellent quantitative agreement. This similarity in results is to be expected since there is virtually no separated flow on the model geometry. One noticeable feature of a viscous flow is the development of secondary flows at internal corners. Shown in Figure 4.13.20 are velocity vectors in a plane normal to the longitudinal axis of the model and located at several stations downstream of the fairwater. The circulation pattern due to the horseshoe vortex that forms at the hull-fairwater junction is clearly visible. Also visible is the vortex emanating from the tip of the fairwater. Further details on this flow and the flow over other hydrodynamic bodies are available in [191].

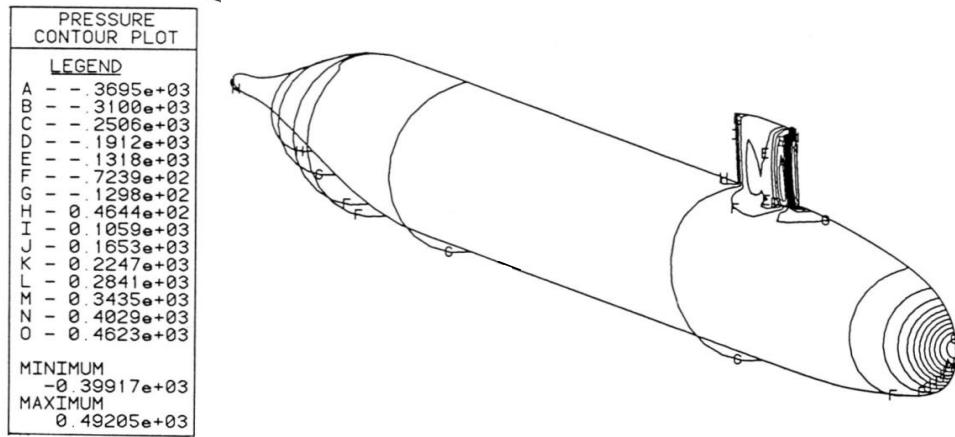


Figure 4.13.19: Pressure contour lines on the surface of the computational domain: 0° attack, 15 contour lines span the pressure extrema.

4.13.9 Crystal Growth from the Melt

As an illustration of the use of spines to compute a free surface shape, the problem of crystal growth from a melt using the Czochralski process is considered. This example is a brief excerpt from the extensive study performed by Sackinger [192] and Sackinger, et al. [193]. Figure 4.13.21 shows a schematic of the axisymmetric crystal growth model. There are two free surfaces in this problem: the melt/gas interface denoted by H_1 and the melt/crystal interface marked by H_0 . Both surfaces are parameterized with a spine technique and the location of each boundary is computed in a fully coupled manner with the momentum, continuity, and energy equations for each material region. Note that the radius of the crystal is also an unknown and was computed as part of the solution. This simulation also included the effect of Lorentz forces on the melt, rotation of the crucible and/or the crystal, and thermocapillary forces on the free surface H_1 . The steady form of this problem is well suited to the spine technique, since the movement of the nodes on each free surface is essentially one dimensional. Note that the spines associated with H_0 and H_1 are permitted to slide along the r coordinate as a result of variations in the crystal radius, R .

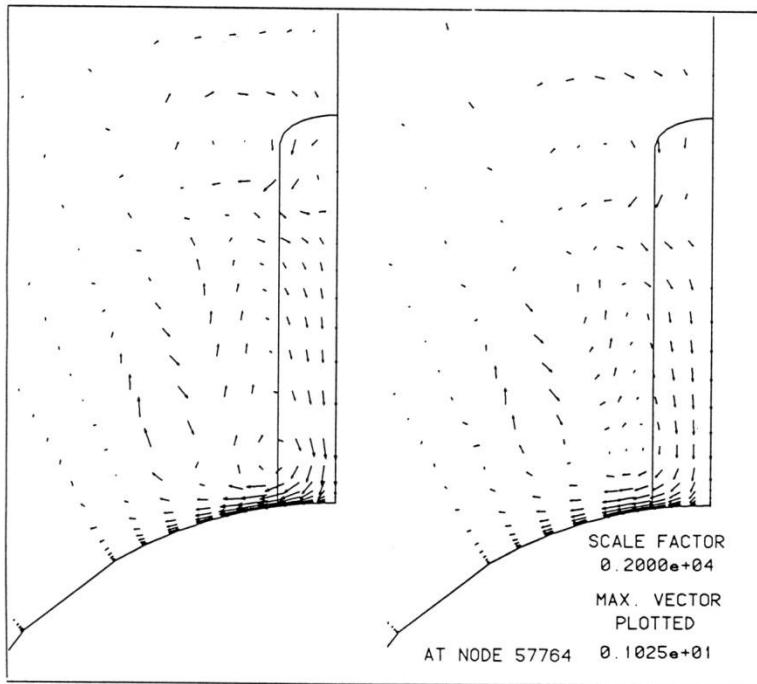


Figure 4.13.20: Velocity vectors depicting the development of the horseshoe vortex which emanates from the hull/fairwater juncture. Velocity vectors in a plane normal to the mean flow direction are illustrated for axial locations corresponding to two (left) and four (right) fairwater chord lengths downstream from the fairwater trailing edge. The mean flow velocity is graphically removed from these illustrations. The fairwater outline is superimposed upon each plot for visual perspective.

Figure 4.13.22 contains typical finite element meshes of nine-node, quadrilateral elements. Contour plots for the streamlines and isotherms for two cases with very different melt pool volumes are shown in Figures 4.13.23 and 4.13.24. The free surface shapes are quite different for these two cases, as are the temperature and flow fields.

4.13.10 Mold Filling

The filling of a cavity with a liquid presents a particularly difficult challenge for a computational algorithm. The present example considers the problem of filling a flat plate mold with a liquid metal in an experimental investment casting setup. This simulation was performed by R. C. Givler (personal communication) using the commercial finite element code, ProCAST [194]. The free surface algorithm in ProCAST is a refined version of the VOF method described in Section 4.11.3.

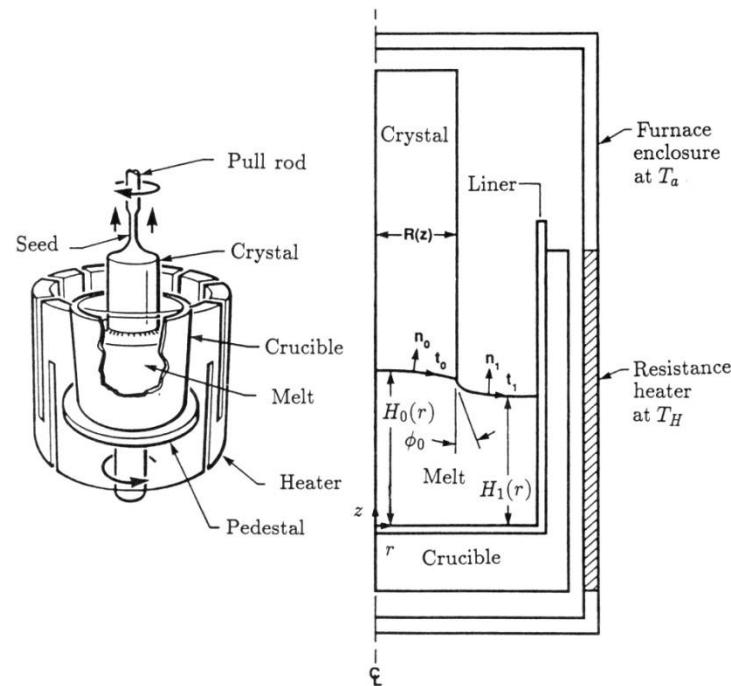


Figure 4.13.21: Schematic of the crystal growth problem.

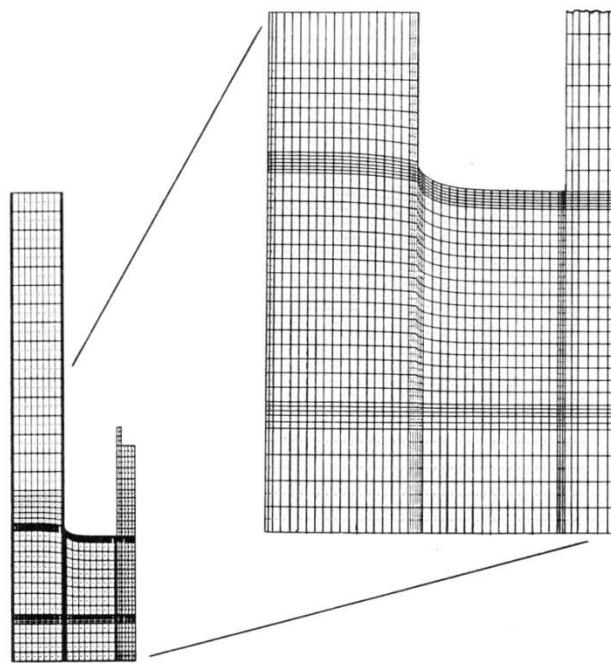


Figure 4.13.22: Typical finite element meshes for the free surface problem.

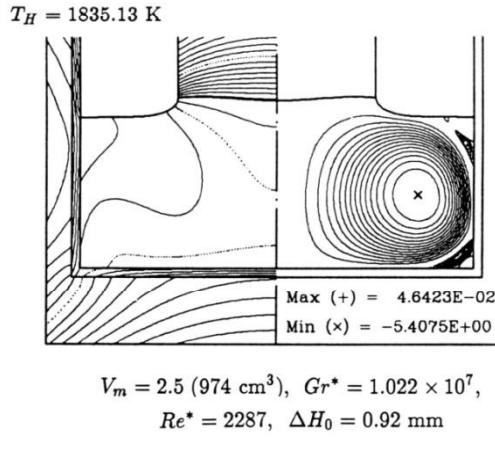


Figure 4.13.23: Isotherms and streamlines for the crystal growth problem (Melt volume $V_m = 2.5$ (974 cm³), $Gr = 1.022 \times 10^7$, $Re = 2,287$, and $\Delta H_0 = 0.92 \text{ mm}$).

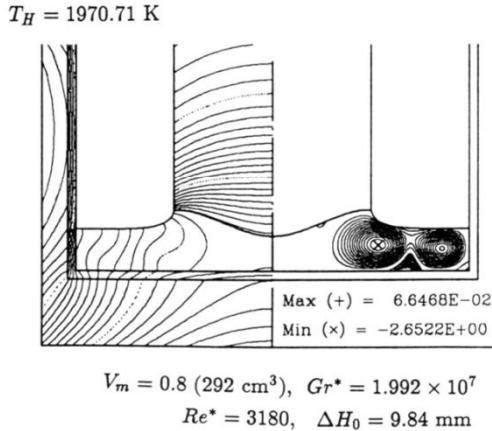


Figure 4.13.24: Isotherms and streamlines for the crystal growth problem (Melt volume $V_m = 0.8$ (292 cm³), $Gr = 1.992 \times 10^7$, $Re = 3,180$, and $\Delta H_0 = 9.84 \text{ mm}$).

Figure 4.13.25 contains the mesh used for the filling simulation. Liquid metal (stainless steel) enters the computational domain through the circular cross section at the top of the edge gate; the fluid temperature and pressure are specified at inflow. For this particular simulation the heat transfer problem is also considered in which the metal loses energy to the mold and the mold is cooled via radiation to the surroundings.

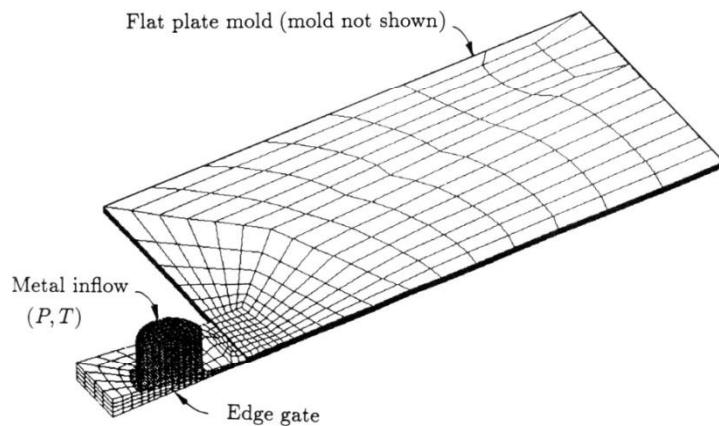


Figure 4.13.25: Schematic and mesh for mold filling (mold not shown).

Figures 4.13.26 and 4.13.27 show the pressure fields at two times during the filling process as viewed from both the top and bottom of the model. It is quite noticeable that the molten metal runs along the bottom of the mold in a film that is less than the channel thickness. Filling of the mold is completed when the fluid reaches the far end of the mold, is turned toward the inlet, and fills the upper part of the channel. Note that the gate thickness is less than the channel thickness, so this behavior is not unexpected.

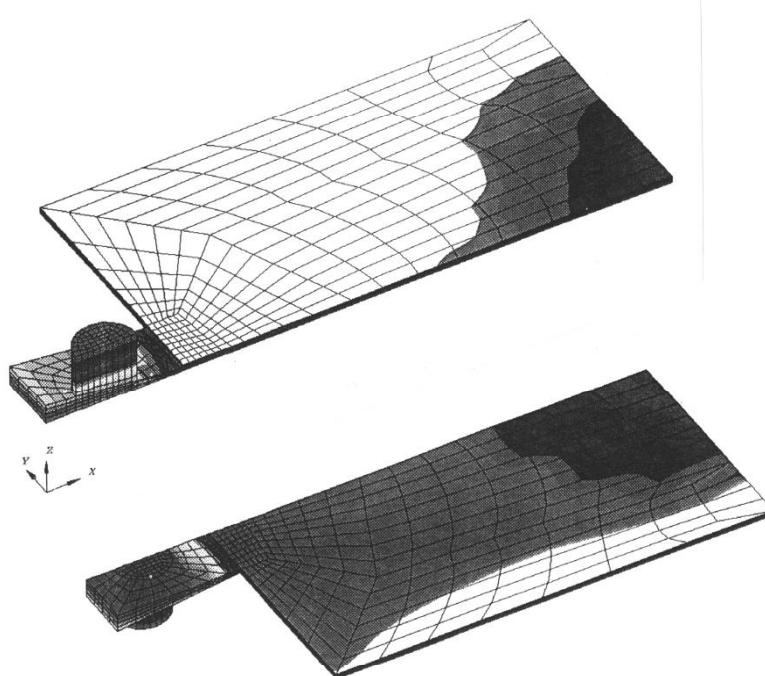


Figure 4.13.26: Top and bottom views of pressure field and free surface during filling; time, $t = 1.45$ s.

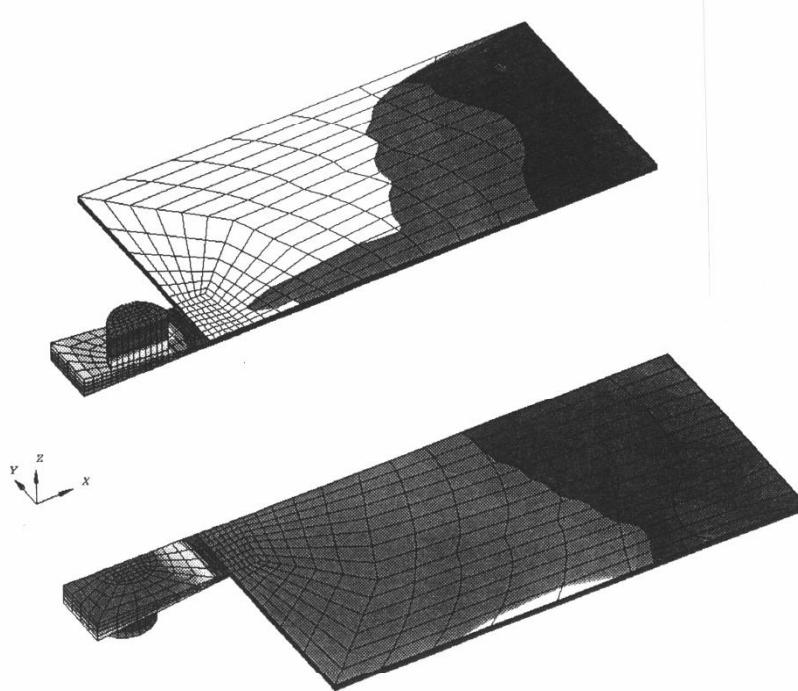


Figure 4.13.27: Top and bottom views of pressure field and free surface during filling; time, $t = 2.09$ s.

4.13.11 Examples Using Least-Squares Finite Element Models

4.13.11.1 Kovasznay flow

First we consider steady, two-dimensional, flow in a square domain $\bar{\Omega} = [-0.5, 1.5] \times [-0.5, 1.5]$. We use Kovasznay's exact solution [195] to the stationary incompressible Navier-Stokes equations to verify exponentially fast decay of the least-squares functional and error norms both measured using the L_2 norm. The solution is given by

$$v_x = 1 - e^{\lambda x} \cos(2\pi y), \quad v_y = \frac{\lambda}{2\pi} e^{\lambda x} \sin(2\pi y), \quad P = \frac{1}{2} \left(1 - e^{2\lambda x} \right) \quad (4.13.2)$$

where $\lambda = Re/2 - (Re^2/4 + 4\pi^2)^{1/2}$. Figure 4.13.28a shows v_x -velocity contours of the exact solution for $Re = 40$ and Figure 4.13.29b shows the discretization of the domain using a non-uniform mesh of 8 quadrilateral finite elements.

The exact solution is used to compute the velocity boundary conditions on Γ and pressure is specified at a point. No boundary conditions on vorticity are necessary. The resulting discrete system is solved using Newton's method with Cholesky factorization at each Newton step. Convergence is declared when the normalized norm of the residual in velocities, $\|\Delta \mathbf{v}\|/\|\mathbf{v}\|$, was less than 10^{-4} , which typically required 5 Newton iterations. A plot of the L_2 least-squares functional and L_2 error of the velocity and vorticity fields as a function of the expansion order in a logarithmic-linear scale is shown in Figure 4.13.29. Exponentially fast decay of the L_2 least-squares functional and L_2 error is observed.

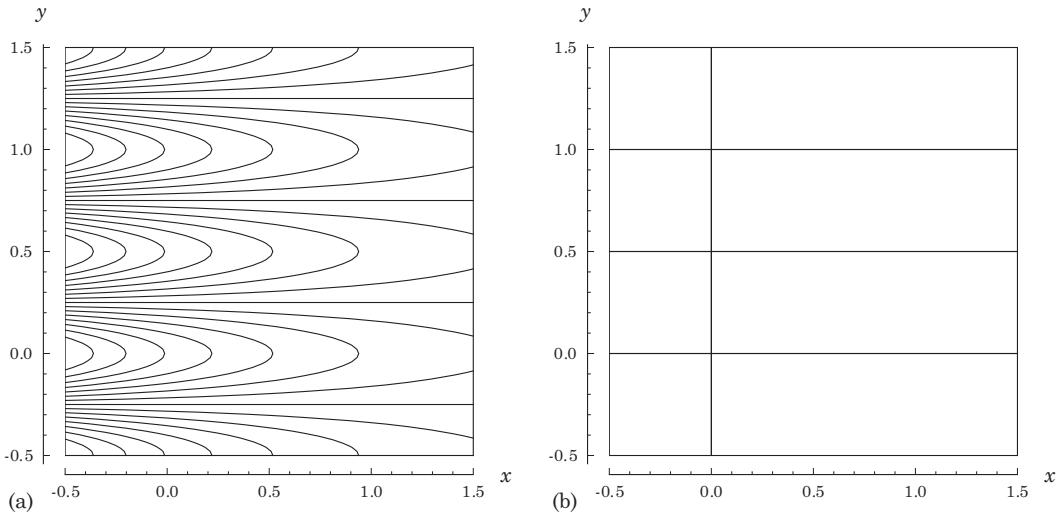


Figure 4.13.28: Kovasznay flow. (a) v_x -velocity component contours of the exact solution for $Re = 40$; (b) computational domain using 8 quadrilateral elements.

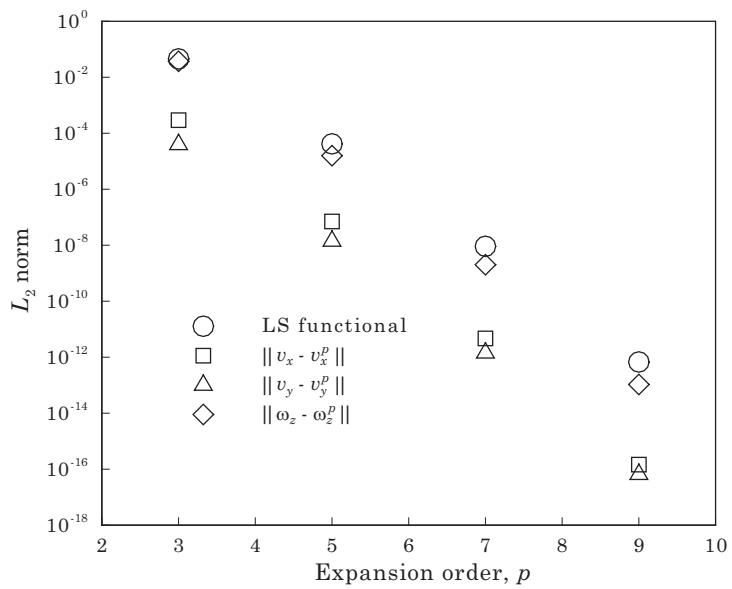


Figure 4.13.29: Decay of the least-squares functional and convergence of the velocity and vorticity fields to the exact Kovasznay solution.

4.13.11.2 Flow over a backward-facing step

Here we revisit the two-dimensional, steady flow over a backward-facing step problem of Section 4.13.7. The geometry and boundary conditions are taken from the benchmark solution of Gartling [119] and are shown in Figure 4.13.30; the standard step geometry was simplified by excluding the channel portion upstream of the step. The boundary conditions for the step geometry include the no-slip condition at all solid surfaces and a parabolic inlet velocity profile given by $v_x(y) = 24y(0.5 - y)$ for $0 \leq y \leq 0.5$. The Reynolds number is based on the mean inlet velocity.

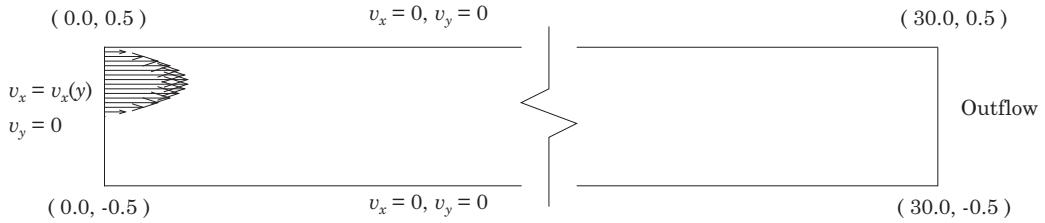


Figure 4.13.30: Geometry and boundary conditions for flow over a backward-facing step.

Instead of imposing an outflow boundary condition in a strong sense, we impose it in a weak sense through the least-squares functional. For example, if we use the vorticity based first-order system the L_2 least-squares functional is given by

$$\begin{aligned} I(\mathbf{v}, P, \omega) = & \frac{1}{2} \left(\| (\mathbf{v} \cdot \nabla) \mathbf{v} + \nabla P + \frac{1}{Re} \nabla \times \omega - \tilde{\mathbf{f}} \|^2_0 + \| \omega - \nabla \times \mathbf{v} \|^2_0 \right. \\ & \left. + \| \nabla \cdot \mathbf{v} \|^2_0 + \| \hat{\mathbf{n}} \cdot \tilde{\sigma} - \tilde{\mathbf{f}}^s \|^2_{0, \Gamma_{\text{outflow}}} \right) \end{aligned} \quad (4.13.3)$$

where $\tilde{\sigma}$ is a pseudo-stress (see Gartling [119]), $\tilde{\sigma} = -P\mathbf{I} + (1/Re) \nabla \mathbf{v}$, and $\tilde{\mathbf{f}}^s$ are the prescribed pseudo-tractions, typically taken to be zero at an outflow boundary.

The domain, $\bar{\Omega} = [0, 30] \times [-0.5, 0.5]$, is discretized using 20 finite elements: two elements along the height of the channel and 10 uniform elements along the length of the channel. The numerical simulation is performed using the two-dimensional incompressible Navier–Stokes equations with the velocity gradient based first-order form (see Pontaza and Reddy [8,10]). An 11th order *modal* expansion is used in each element and the resulting discrete system is solved using Newton’s method. At each Newton step, the linear system of algebraic equations is solved using the conjugate gradient method with a symmetric Gauss–Seidel preconditioner. Convergence of the conjugate gradient method was declared when the norm of the residual was less than 10^{-6} . Nonlinear convergence was declared when the normalized norm of the residual in velocities, $\|\Delta \mathbf{v}\|/\|\mathbf{v}\|$, was less than 10^{-4} , which typically required four Newton iterations. The analysis starts with $Re = 100$ and steps to $Re = 800$ using

a solution continuation technique with increments of $Re = 100$. Away from the corner of the step at $(x, y) = (0, 0)$, the least-squares functional remained below 10^{-5} through the Reynolds number stepping.

Figure 4.13.31 shows the streamlines, the vector velocity field, and pressure contours for $0 \leq x \leq 10$, where most of the interesting flow structures occur. The flow separates at the step corner and forms a large recirculation region with a reattachment point on the lower wall of the channel at approximately $x = 6$. A second recirculation region forms on the upper wall of the channel beginning near $x = 5$ with a reattachment point at approximately $x = 10.5$.

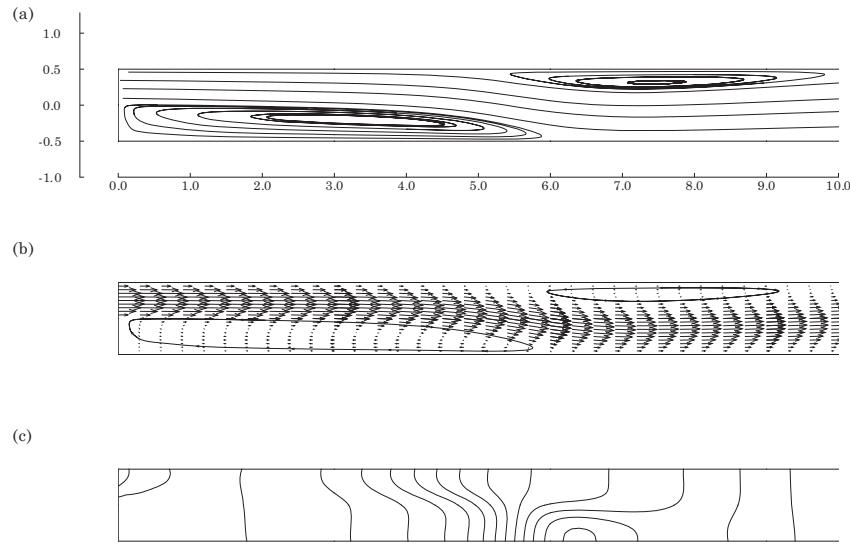


Figure 4.13.31: Flow over a backward-facing step at $Re = 800$: (a) streamlines, (b) vector velocity field, and (c) pressure field.

Figure 4.13.32 shows v_x -velocity profiles along the channel height at $x = 7$ and $x = 15$. We compare with tabulated values from the benchmark solution of Gartling [119] (also see Section 4.13.7) and find excellent agreement. Gartling's benchmark solution is based on a mixed Galerkin formulation using grid systems ranging from 6×120 to 40×800 biquadratic elements. Figure 4.13.33 shows pressure profiles along the length of the channel walls. The slopes of the pressure profiles become constant near the exit plane, meaning that the flow has recovered to fully developed conditions at the exit.

4.13.11.3 Flow past a circular cylinder at low Reynolds number

The last example deals with the two-dimensional flow of an incompressible fluid past a circular cylinder [8]. The cylinder is of unit diameter and is placed in the finite region $\bar{\Omega} = [-15.5, 30.5] \times [-20.5, 20.5]$. The center of the cylinder lies at $(x, y) = (0, 0)$, so that the inflow boundary is located at 10.5 cylinder diameters left (or in front) of the center of the cylinder and the outflow boundary is at 30.5

cylinder diameters downstream of the center of the cylinder. The top and bottom boundaries are located each at 20.5 cylinder diameters above and below the center of the cylinder. Having considered a large computational domain allows us to impose free-stream boundary conditions at the top and bottom of the domain without noticeably affecting the solution.

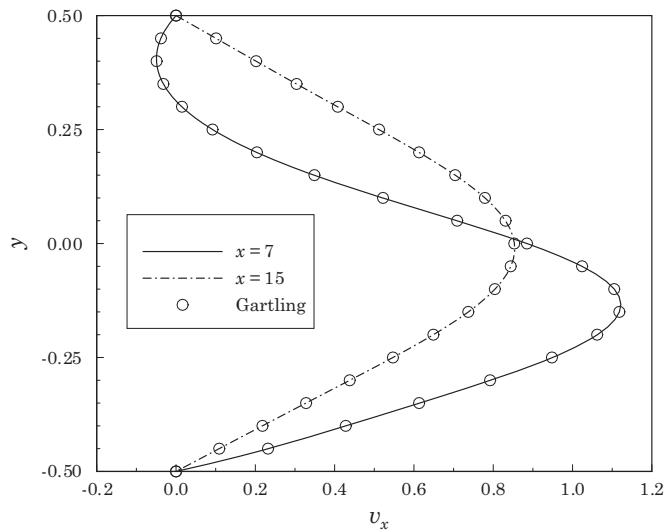


Figure 4.13.32: Horizontal velocity profiles along the height of the channel at $x = 7$ and $x = 15$ for flow over a backward-facing step problem at $Re = 800$.

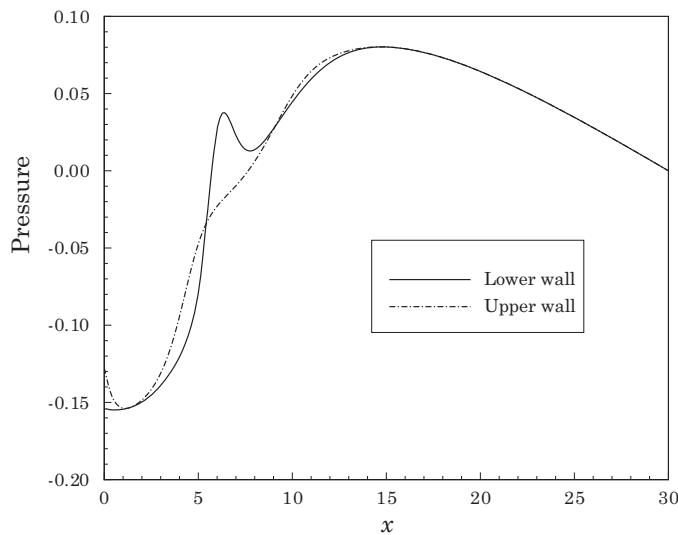


Figure 4.13.33: Pressure profiles along lower and upper walls of the channel for flow over a backward-facing step at $Re = 800$.

The boundary conditions include a specified value of 1.0 for the x -component of velocity at the inflow, top, and bottom boundaries; that is, the free-stream velocity u_∞ is specified to be unity. At these boundaries the y -component of velocity is set to zero. The outflow boundary conditions are imposed in a weak sense through the least-squares functional. The Reynolds numbers considered here are 20 and 40, for which a steady-state solution exists. The Reynolds number is based on the free-stream velocity and cylinder diameter.

The vorticity based finite element model in Eq. (4.9.23) is used with sixth-order nodal expansions in each element. The finite element mesh consists of 501 finite elements (see Figure 4.13.34), where a close-up view of the geometric discretization around the circular cylinder is also shown. To accurately represent the circular arc, the same approximation for the geometry and the solution (i.e. isoparametric formulation) is used. The total number of degrees of freedom for the problem is 73,344. The storage of the assembled system of equations in banded or in compressed sparse row/column format for such large size problems is prohibitively expensive in terms of computer memory. Therefore matrix-free techniques, also known as element-by-element solution algorithms, are implemented in a matrix-free version of the conjugate gradient method with a Jacobi preconditioner.

At each Newton step the linear system of equations is solved using the matrix-free conjugate gradient algorithm with a Jacobi preconditioner and convergence tolerance for the norm of the residual to be 10^{-6} . Nonlinear convergence is declared when the relative norm of the residual in velocities between two consecutive iterations was less than 10^{-4} , which required less than six Newton iterations.

Figure 4.13.35 shows the computed surface pressure coefficient distributions along the cylinder surface for $Re = 20$ and 40, together with experimental measurements of Grove et al. [107] for $Re = 40$. The finite element results are in good agreement with the experimental measurements. The computed drag coefficients for $Re = 20$

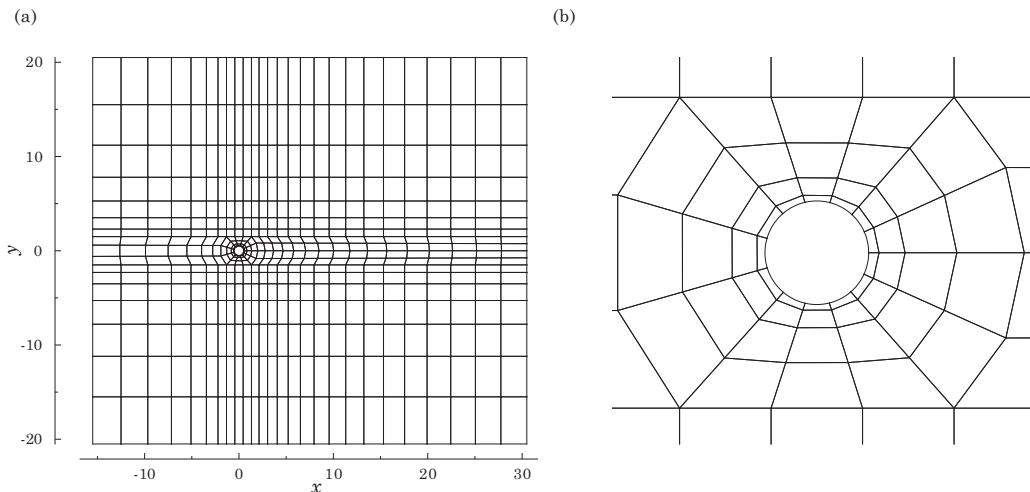


Figure 4.13.34: Computational domain and mesh for flow past a circular cylinder.
 (a) Computational mesh. (b) Close-up view of the geometric discretization around the circular cylinder.

and $Re = 40$ are $C_D = 2.0862$ and $C_D = 1.5537$, respectively. Good agreement is found between the computed drag coefficients and the experimental mean curve of Tritton [196], where the corresponding values are $C_D = 2.05$ and $C_D = 1.56$.

Figure 4.13.36 shows computed pressure contours and streamlines in the wake region for $Re = 20$ and $Re = 40$. The predicted wake extends 1.86 and 4.55 cylinder radii measured from the back of the cylinder. The values for the wake lengths are in good agreement with the numerical solution of Dennis and Chang [197], whose computed wake lengths for $Re = 20$ and 40 were reported as 1.88 and 4.69 cylinder radii, respectively. Better agreement for the case $Re = 40$ is found with the numerical solution of Kawaguti and Jain [198], who reported a computed wake length of 4.50 cylinder radii.

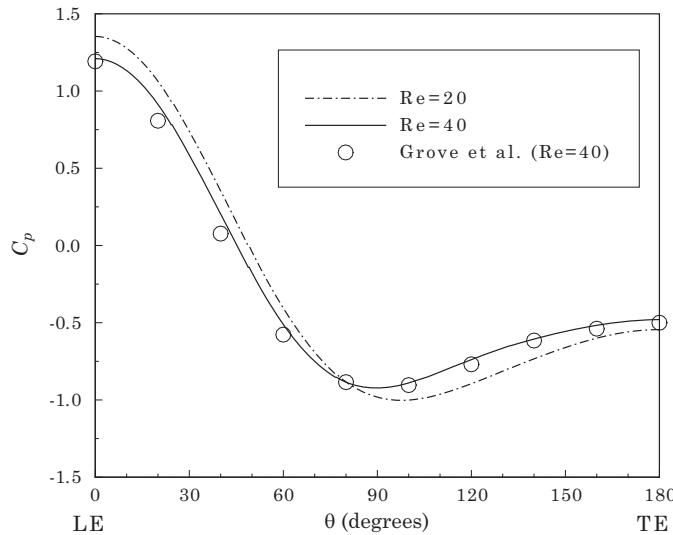


Figure 4.13.35: Comparison of the computed pressure coefficient distributions along the cylinder surface with experimental results of Grove et al. [107] for $Re = 40$.

Problems

4.1 For axisymmetric viscous incompressible flows (i.e., the flow field is independent of the θ coordinate), and when the convective terms are neglected, the governing equations are

$$\frac{1}{r} \frac{\partial}{\partial r} (r\sigma_r) - \frac{\sigma_\theta}{r} + \frac{\partial\sigma_{rz}}{\partial z} + f_r = \rho \frac{\partial u}{\partial t} \quad (1)$$

$$\frac{1}{r} \frac{\partial}{\partial r} (r\sigma_{rz}) + \frac{\partial\sigma_{zz}}{\partial z} + f_z = \rho \frac{\partial w}{\partial t} \quad (2)$$

$$\frac{1}{r} \frac{\partial}{\partial r} (ru) + \frac{\partial w}{\partial z} = 0 \quad (3)$$

where

$$\sigma_r = -P + 2\mu \frac{\partial u}{\partial r}, \quad \sigma_\theta = -P + 2\mu \frac{u}{r} \quad (4)$$

$$\sigma_z = -P + 2\mu \frac{\partial w}{\partial z}, \quad \sigma_{rz} = \mu \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial r} \right) \quad (5)$$

Develop the semidiscrete mixed finite element model of the equations.

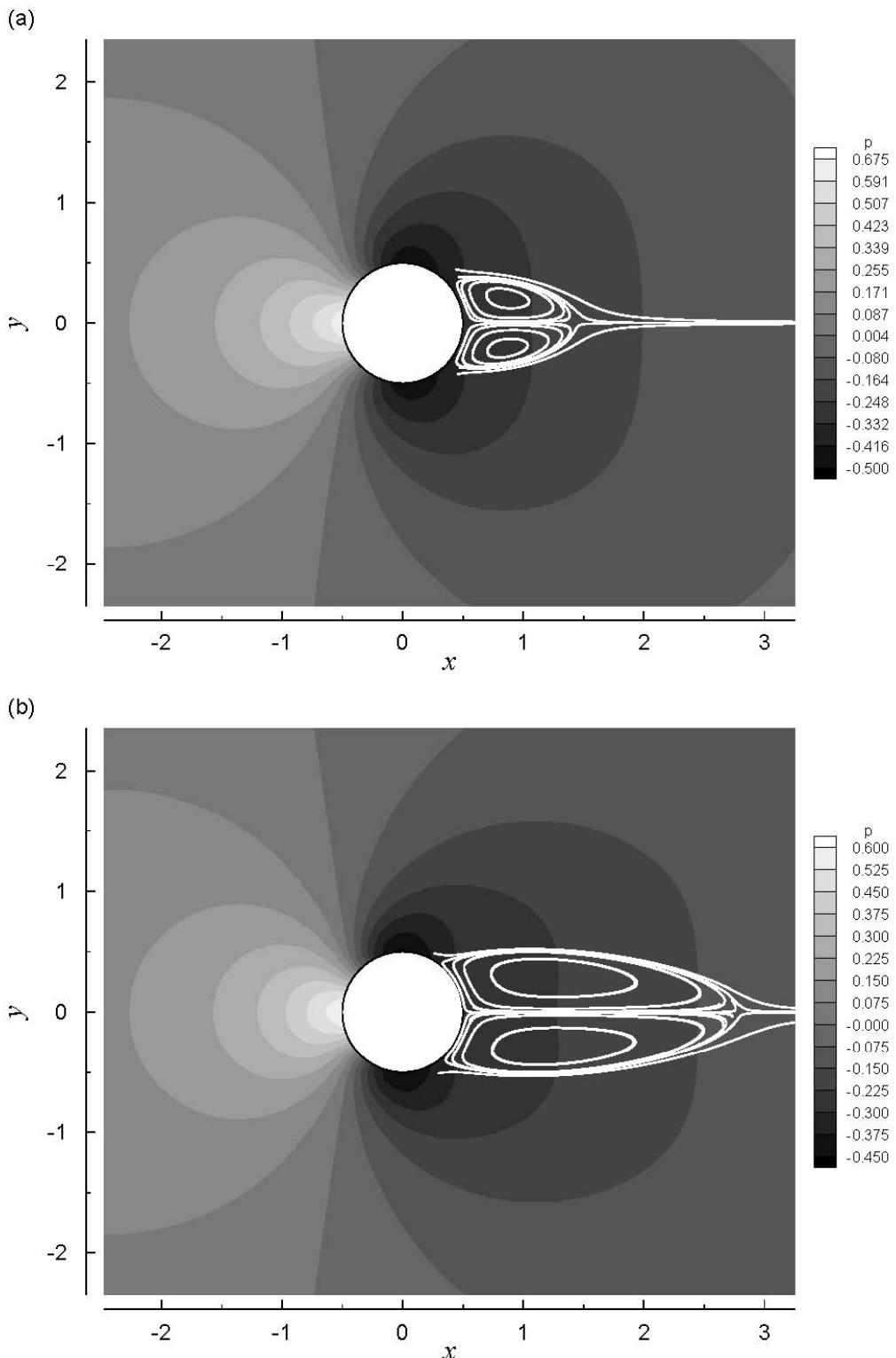


Figure 4.13.36: Flow past a circular cylinder at (a) $Re = 20$ and (b) $Re = 40$: pressure contours and streamlines in the wake region.

4.2 Develop the semidiscrete penalty finite element model of the equations in Problem 4.2.

4.3 The equations governing unsteady slow flow of viscous, incompressible fluids in the xy -plane can be written in terms of vorticity ω and stream function ψ :

$$\rho \frac{\partial \omega}{\partial t} - \mu \nabla^2 \omega = 0$$

$$-\omega - \nabla^2 \psi = 0$$

Develop the semidiscrete finite element model of the equations. Discuss the meaning of the secondary variables. Use the α -family of approximation to reduce the ordinary differential equations to algebraic equations.

4.4 Compute the tangent coefficient matrix for the penalty finite element model in Eq. (4.3.18).

4.5 Analyze the *wall-driven cavity* problem for $Re=5,000$ using the 16×20 mesh of bilinear elements and the penalty finite element model. The following increments [$DX(I)$ and $DY(I)$] along x and y are suggested:

$$\begin{aligned} & 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \\ & 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \\ & 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \\ & 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.0625 \quad 0.03125 \quad 0.03125 \quad 0.03125 \\ & \quad \quad \quad \quad \quad 0.03125 \quad 0.03125 \quad 0.03125 \end{aligned}$$

You may experiment with the acceleration parameter (for convergence); use a convergence tolerance of 10^{-2} . Plot the horizontal velocity v_x along the vertical centerline of the cavity, y for the linear and nonlinear cases. Note that you must pick ρ and μ to obtain the desired Reynolds number.

References for Additional Reading

1. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).
2. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis*, Oxford University Press, Oxford, UK (2004).
3. J. N. Reddy, “On Penalty Function Methods in the Finite Element Analysis of Flow Problems,” *International Journal for Numerical Methods in Fluids*, **2**, 151–171 (1982).
4. J. T. Oden, “RIP Methods for Stokesian Flows,” in R. H. Gallagher, O. C. Zienkiewicz, J. T. Oden, and D. Norrie (eds.), *Finite Element Method in Flow Problems*, Vol. IV, John Wiley & Sons, London (1982).
5. B. Q. Li, *Discontinuous Finite Element Methods in Fluid Mechanics and Heat Transfer*, Springer, London (2006).
6. Bo-Nan Jiang, *The Least-Squares Finite Element Method. Theory and Applications in Computational Fluid Dynamics and Electromagnetics*, Springer-Verlag, New York (1998).
7. K. S. Surana, A. R. Ahmadi and J. N. Reddy, “The k -Version of Finite Element Method for Non-Self-Adjoint Operators in BVP,” *International Journal of Computational Engineering Science*, **4**(4), 737–812 (2003).
8. J. P. Pontaza and J. N. Reddy, “Spectral/hp Least-Squares Finite Element Formulation for the Navier-Stokes Equations,” *Journal of Computational Physics*, **190**(2), 523–549 (2003).
9. K. S. Surana, A. R. Ahmadi and J. N. Reddy, “The k -Version of Finite Element Method for Nonlinear Operators in BVP,” *International Journal of Computational Engineering Science*, **5**(1), 133–207 (2004).
10. J. P. Pontaza and J. N. Reddy, “Space-Time Coupled Spectral/hp Least-Squares Finite Element Formulation for the Incompressible Navier-Stokes Equations,” *Journal of Computational Physics*, **197**(2), 418–459 (2004).

11. J. P. Pontaza and J. N. Reddy, "Least-squares finite element formulations for viscous compressible and incompressible fluid flows," *Computer Methods in Applied Mechanics and Engineering*, **195**, 2454–2494 (2006).
12. V. Prabhakar and J. N. Reddy, "Spectral/hp Penalty Least-Squares Finite Element Formulation for the Steady Incompressible Navier–Stokes Equations," *Journal of Computational Physics*, **215**(1), 274–297 (2006).
13. V. Prabhakar and J. N. Reddy, "A Stress-based Least-Squares Finite-element Model for Incompressible Navier–Stokes Equations," *International Journal for Numerical Methods in Fluids*, **54**(11), 1369–1385 (2007).
14. K. S. Surana, S. Bhola, J. N. Reddy, and P. W. TenPas, " k -Version of finite element method in 2D-polymer flows: Upper convected Maxwell model," *Computers and Structures*, **86**(17–18), 1782–1808 (2008).
15. P. B. Bochev and M. D. Gunzburger, *Least-Squares Finite Element Methods*, Springer, New York (2009).
16. J. T. Oden and G. F. Carey, *Finite Elements, Mathematical Aspects*, Vol. IV, Prentice Hall, Englewood Cliffs, New Jersey (1983).
17. F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer–Verlag, Berlin (1991).
18. F. Brezzi and K. J. Bathe, "The inf-sup Condition, Equivalent Forms and Applications," in *Reliability of Methods for Engineering Analysis*, K. J. Bathe and D. R. J. Owen (eds.), Pineridge Press, Swansea, U.K. (1986).
19. D. S. Malkus and E. T. Olsen, "Obtaining Error Estimates for Optimally Constrained Incompressible Finite Elements," *Computer Methods in Applied Mechanics and Engineering*, **45**, 331–353 (1984).
20. P. Le Tallec and V. Ruas, "On the Convergence of the Bilinear-Velocity Constant-Pressure Finite Element Method in Viscous Flow," *Computer Methods in Applied Mechanics and Engineering*, **54**, 235–243 (1986).
21. D. Chapelle and K. J. Bathe, "The inf-sup Test," *Computers & Structures*, **47**(4/5), 537–545 (1993).
22. J. N. Reddy, "On the Accuracy and Existence of Solutions to Primitive Variable Models of Viscous Incompressible Fluids," *International Journal of Engineering Science*, **16**, 921–929 (1978).
23. J. N. Reddy, "On the Finite Element Method with Penalty for Incompressible Fluid Flow Problems," in *The Mathematics of Finite Elements and Applications III*, J. R. Whiteman (ed.), Academic Press, New York (1979).
24. R. Temam, *Theory and Numerical Analysis of the Navier–Stokes Equations*, North-Holland, Amsterdam (1977).
25. R. L. Sani, P. M. Gresho, R. L. Lee, and D. F. Griffiths, "The Cause and Cure (?) of the Spurious Pressures Generated by Certain FEM Solutions of the Incompressible Navier–Stokes Equations: Part 1," *International Journal for Numerical Methods in Fluids*, **1**, 17–43 (1981).
26. R. L. Sani, P. M. Gresho, R. L. Lee, and D. F. Griffiths, "The Cause and Cure (?) of the Spurious Pressures Generated by Certain FEM Solutions of the Incompressible Navier–Stokes Equations: Part 2," *International Journal for Numerical Methods in Fluids*, **1**, 171–204 (1981).
27. T. J. R. Hughes, R. L. Taylor, and J. F. Levy, "A Finite Element Method for Incompressible Viscous Flows," in *Proceedings of the Second International Symposium on Finite Element Methods in Flow Problems*, S. Margherita, Italy (1976).
28. T. J. R. Hughes, W. K. Liu, and A. Brooks, "Review of Finite Element Analysis of Incompressible Viscous Flows by Penalty Function Formulation," *Journal of Computational Physics*, **30**, 1–60 (1979).
29. M. Bercovier and M. Engelman, "A Finite Element for the Numerical Solution of Viscous Incompressible Flows," *Journal of Computational Physics*, **30**, 181–201 (1979).
30. M. Engelman, R. Sani, P. M. Gresho, and M. Bercovier, "Consistent vs. Reduced Integration Penalty Methods for Incompressible Media Using Several Old and New Elements," *International Journal for Numerical Methods in Fluids*, **2**, 25–42 (1982).

31. R. S. Marshall, J. C. Heinrich, and O. C. Zienkiewicz, "Natural Convection in a Square Enclosure by a Finite Element, Penalty Function Method, Using Primitive Fluid Variables," *Numerical Heat Transfer*, **1**, 315–330 (1978).
32. J. C. Heinrich and M. Strada, "Penalty Finite Element Analysis of Natural Convection at High Rayleigh Numbers," in *Finite Elements in Fluids*, Vol. 4, R. H. Gallagher, D. Norrie, J. T. Oden and O. C. Zienkiewicz (eds.), John Wiley, New York (1982).
33. M. P. Reddy and J. N. Reddy, "Finite-Element Analysis of Flows of Non-Newtonian Fluids in Three-Dimensional Enclosures," *International Journal of Non-Linear Mechanics*, **27**, 9–26 (1992).
34. M. P. Reddy, J. N. Reddy, and H. U. Akay, "Penalty Finite Element Analysis of Incompressible Flows Using Element by Element Solution Algorithms," *Computer Methods in Appl. Mech. and Engng.*, **100**, 169–205 (1992).
35. D. K. Gartling, C. E. Hickox and R. C. Givler, "Simulation of Viscous and Porous Flow Problems," *Computational Fluid Dynamics*, **7**, 23–48 (1996).
36. P. M. Gresho, "Some Current CFD Issues Relevant to the Incompressible Navier–Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, **87**, 201–252 (1991).
37. P. M. Gresho, "Incompressible Fluid Dynamics: Some Fundamental Formulation Issues," *Annual Review of Fluid Mechanics*, **23**, 413–453 (1991).
38. P. M. Gresho, "Some Interesting Issues in Incompressible Fluid Dynamics, Both in the Continuum and in Numerical Simulation," in *Advances in Applied Mechanics*, J. W. Hutchinson and T. Y. Wu (eds.), **28**, 46–140 (1992).
39. P. M. Gresho and R. L. Sani, *Incompressible Flow and the Finite Element Method*, John Wiley & Sons, London (1998).
40. C. Taylor and P. Hood, "A Numerical Solution of the Navier–Stokes Equations Using FEM Techniques," *Computers and Fluids*, **1**, 73–100 (1973).
41. T. J. R. Hughes, L. P. Franca, and M. Balestra, "A New Finite Element Formulation for Computational Fluid Dynamics, V. Circumventing the Babuska–Brezzi Condition: A Stable Petrov–Galerkin Formulation for the Stokes Problem Accommodating Equal-Order Interpolations," *Computer Methods in Applied Mechanics and Engineering*, **59**, 85–99 (1986).
42. T. E. Tezduyar, "Stabilized Finite Element Formulations for Incompressible Flow Computations," in *Advances in Applied Mechanics*, J. W. Hutchinson and T. Y. Wu (eds.), **28**, 1–44 (1992).
43. L. P. Franca, T. J. R. Hughes and R. Stenberg, "Stabilized Finite Element Methods," in *Incompressible Computational Fluid Dynamics*, M. D. Gunzburger and R. A. Nicolaides (eds), Cambridge University Press, Cambridge, U. K., 87–107 (1993).
44. B.-N Jiang, T. L. Lin and L. A. Povinelli, "Large-Scale Computation of Incompressible Viscous Flow by the Least-Squares Finite Element Method," *Computer Methods in Applied Mechanics and Engineering*, **114**, 213–231 (1994).
45. G. F. Carey, A. I. Pehlivanov, Y. Shen, A. Bose and K. C. Wang, "Least-Squares Finite Elements for Fluid Flow and Transport," *International Journal for Numerical Methods in Fluids*, **27**, 97–107 (1998).
46. C. R. Dohrmann and P. B. Bochev, "A Stabilized Finite Element Method for the Stokes Problem Based on Polynomial Pressure Projection," *International Journal for Numerical Methods in Fluids*, **46**, 183–201 (2004).
47. M. Fortin and S. Boivin, "Iterative Stabilization of the Bilinear Velocity - Constant Pressure Element," *International Journal for Numerical Methods in Fluids*, **10**, 125–140 (1990).
48. J. Qin and S. Zhang, "On the Selective Local Stabilization of the Mixed Q1 - P0 Element," *International Journal for Numerical Methods in Fluids*, **55**, 1121–1141 (2007).
49. E. M. Salonen and J. Aalto, "A Pressure Determination Scheme," *Proceedings of the Fifth International Conference on Numerical Methods in Laminar and Turbulent Flow*, C. Taylor, M. D. Olson, P. M. Gresho, and W. G. Habashi (eds.), Pineridge Press, Swansea, U.K. (1985).
50. T. Shiojima and Y. Shimazaki, "A Pressure-Smoothing Scheme for Incompressible Flow Problems," *International Journal for Numerical Methods in Fluids*, **9**, 557–567 (1989).

51. M. Engelman, R. L. Sani, and P. M. Gresho, "The Implementation of Normal and/or Tangential Boundary Conditions in Finite Element Codes for Incompressible Fluid Flow," *International Journal for Numerical Methods in Fluids*, **2**, 225–238 (1982).
52. J. A. Scott, "A Parallel Frontal Solver for Finite Element Applications," *International Journal for Numerical Methods in Engineering*, **50**, 1131–1144 (2001).
53. H. C. Elman, D. J. Silvester and A. J. Walther, *Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics*, Oxford University Press, Oxford (2005).
54. C. R. Dohrmann and O. B. Widlund, "An Overlapping Schwartz Algorithm for Almost Incompressible Elasticity," Courant Institute Report No. TR2008-912 (2008).
55. G. C. Broyden, "A Class of Methods for Solving Nonlinear Simultaneous Equations," *Mathematics of Computation*, **19**, 577–593 (1965).
56. M. S. Engelman, G. Strang, and K. J. Bathe, "The Application of Quasi-Newton Methods in Fluid Mechanics," *International Journal for Numerical Methods in Engineering*, **17**, 707–718 (1981).
57. P. Underwood, "Dynamic Relaxation," in *Computational Methods for Transient Analysis*, T. Belytschko and T. J. R. Hughes (eds.), North-Holland, Amsterdam, 245–265 (1983).
58. C. P. Jackson and K. H. Winters, "A Finite Element Study of the Bernard Problem Using Parameter-Stepping and Bifurcation Search," *International Journal of Numerical Methods in Fluids*, **4**, 127–145 (1984).
59. K. H. Winters, K. A. Cliffe, and C. P. Jackson, "The Prediction of Instabilities Using Bifurcation Theory," in *Numerical Methods for Transient and Coupled Problems*, R. W. Lewis, et al. (eds.), John Wiley, New York, 179–198 (1987).
60. K. H. Winters and R. O. Jack, "Anomalous Convection at Low Prandtl Number," *Communications in Applied Numerical Methods*, **5**, 401–404 (1989).
61. K. Winters, "Bifurcation and Stability: A Computational Approach," *Computational Physics Communications*, **65**, 299–309 (1991).
62. W. F. Schmidt, "Adaptive Step Size Selection for Use with the Continuation Method," *International Journal for Numerical Methods in Engineering*, **12**, 677–694 (1978).
63. W. C. Rheinboldt, "On the Solution of Some Nonlinear Equations Arising in the Application of Finite Element Methods," in *Proceedings of the Conference on The Mathematics of Finite Elements and Applications*, J. R. Whiteman (ed.), Brunel University, Academic Press, London (1975).
64. W. C. Rheinboldt, "Numerical Analysis of Continuation Methods for Nonlinear Structural Problems," *Computers and Structures*, **13**, 103–113 (1981).
65. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, McGraw-Hill, New York (1980).
66. V. Haroutunian, M. S. Engelman, and I. Hasbani, "Segregated Finite Element Algorithms for the Numerical Solution of Large-Scale Incompressible Flow Problems," *International Journal for Numerical Methods in Fluids*, **17**, 323–348 (1993).
67. A. C. Benim and W. Zinser, "A Segregated Formulation of Navier–Stokes Equations with Finite Elements," *Computer Methods in Applied Mechanics and Engineering*, **57**, 223–237 (1986).
68. J. G. Rice and R. J. Schnipke, "An Equal-Order Velocity-Pressure Formulation that Does Not Exhibit Spurious Pressure Modes," *Computer Methods in Applied Mechanics and Engineering*, **58**, 135–149 (1986).
69. P. M. Gresho, S. T. Chan, R. L. Lee, and C. D. Upson, "A Modified Finite Element Method for Solving the Time-Dependent, Incompressible Navier–Stokes Equations. Part 1: Theory," *International Journal for Numerical Methods in Fluids*, **4**, 557–598 (1984).
70. P. M. Gresho and S. T. Chan, "On the Theory of Semi-Implicit Projection Methods for Viscous Incompressible Flow and Its Implementation Via a Finite Element Method That Introduces a Nearly Consistent Mass Matrix, Part 1—Theory," *International Journal for Numerical Methods in Fluids*, **11**, 587–620 (1990).

71. P. M. Gresho, R. L. Lee, S. T. Chan, and R. L. Sani, "Solution of the Time-Dependent Incompressible Navier-Stokes and Boussinesq Equations Using the Galerkin Finite Element Method," in *Proc. IUTAM Symposium on Approximation Methods for Navier-Stokes Problems*, Paderborn, W. Germany, Springer-Verlag, Berlin (1979).
72. P. M. Gresho, D. F. Griffiths and D. J. Silvester, "Adaptive Time Stepping for Incompressible Flows, Part I: Scalar Advection-Diffusion," *SIAM Journal on Scientific Computing*, **30**, 2018–2054 (2008).
73. M. A. Christon, "A Domain Decomposition Message-Passing Approach to Transient Viscous Incompressible Flow Using Explicit Time Integration," *Computer Methods in Applied Mechanics and Engineering*, **148**, 329–352 (1997).
74. J. C. Heinrich, P. S. Huyakorn, O. C. Zienkiewicz and A. R. Mitchell, "An 'Upwind' Finite Element Scheme for Two-Dimensional Convective Transport Equation," *International Journal for Numerical Methods in Engineering*, **11**, 131–143 (1977).
75. J. C. Heinrich and O. C. Zienkiewicz, "The Finite Element Method and 'Upwinding' Techniques in the Numerical Solutions of Convection Dominated Flow Problems," in *Finite Element Methods for Convection Dominated Flows*, T. J. R. Hughes (ed.), ASME, AMD **34**, New York, 105–136 (1979).
76. I. Christie, D. F. Griffiths, A. R. Mitchell and O. C. Zienkiewicz, "Finite Element Methods for Second Order Differential Equations with Significant First Derivatives," *International Journal for Numerical Methods in Engineering*, **10**, 1389–1396 (1976).
77. T. J. R. Hughes and A. N. Brooks, "A Multi-Dimensional Upwind Scheme with No Crosswind Diffusion," in *Finite Element Methods for Convection Dominated Flows*, T. J. R. Hughes (ed.), ASME, AMD **34**, NY, 19–35 (1979).
78. A. N. Brooks and T. J. R. Hughes, "Streamline-Upwind/Petrov-Galerkin Formulations for Convection Dominated Flows with Particular Emphasis on Incompressible Navier-Stokes Equation," *Computer Methods in Applied Mechanics and Engineering*, **32**, 199–259 (1982).
79. T. J. R. Hughes and T. E. Tezduyar, "Finite Element Methods for First-Order Hyperbolic Systems with Particular Emphasis on the Compressible Euler Equations," *Computer Methods in Applied Mechanics and Engineering*, **45**, 217–284 (1982).
80. T. E. Tezduyar, S. Mittal, S. E. Ray and R. Shih, "Incompressible Flow Computations with Stabilized Bilinear and Linear Equal-Order-Interpolation Velocity-Pressure Elements," *Computer Methods in Applied Mechanics and Engineering*, **95**, 221–242 (1992).
81. T. J. R. Hughes, "Multiscale Phenomena: Green's Functions, the Dirichlet-to-Neumann Formulation, Subgrid Scale Models, Bubbles and the Origin of Stabilized Methods," *Computer Methods in Applied Mechanics and Engineering*, **127**, 387–401 (1995).
82. T. J. R. Hughes, G. R. Feijóo, L. Mazzi and J-B. Quincy, "The Variational Multiscale Method - A Paradigm for Computational Mechanics," **166**, 3–24 (1998).
83. T. J. R. Hughes, L. Mazzi and K. E. Jansen, "Large Eddy Simulation and the Variational Multiscale Method," *Computing and Visualization in Science*, **3**, 47–59 (2000).
84. T. J. R. Hughes, L. Mazzi, A. Oberai and A. Wray, "The Multiscale Formulation of Large Eddy Simulation: Decay of Homogeneous Isotropic Turbulence," *Physics of Fluids*, **13**, 505–512 (2000).
85. T. J. R. Hughes and L. P. Franca, "A New Finite Element Formulation for Computational Fluid Dynamics: VII. The Stokes Problem with Various Well-posed Boundary Conditions: Symmetric Formulations that Converge for All Velocity/Pressure Spaces," *Computer Methods in Applied Mechanics and Engineering*, **65**, 85–96 (1987).
86. T. J. R. Hughes, L. P. Franca and G. M. Hulbert, "A New Finite Element Formulation for Computational Fluid Dynamics: VIII. The Galerkin/Least-squares Method for Advective-Diffusive Equations," *Computer Methods in Applied Mechanics and Engineering*, **73**, 173–189 (1989).
87. F. Shakib, "Finite Element Analysis of the Compressible Euler and Navier-Stokes Equations," Ph.D. Thesis, Stanford University, Stanford (1988).
88. D. K. Gartling and C. R. Dohrmann, "Quadratic Finite Elements and Incompressible Viscous Flows," *Computer Methods in Applied Mechanics and Engineering*, **95**, 1692–1708 (2006).

89. G. Wagner and W. K. Liu, "Coupling of Atomistic and Continuum Simulations Using a Bridging Scale Decomposition," *Journal of Computational Physics*, **190**, 249–274 (2003).
90. G. Wagner, R. E. Jones, J. A. Templeton and M. L. Parks, "An Atomistic-to-Continuum Coupling Method for Heat Transfer in Solids," *Computer Methods in Applied Mechanics and Engineering*, **197**, 3351–3365 (2008).
91. T. J. R. Hughes, G. Scovazzi and L. Franca, "Chapter 2 - Multiscale and Stabilized Methods," in E. Stein, R. de Borst and T. J. R. Hughes (eds.), *Encyclopedia of Computational Mechanics, Volume 3: Fluids*, John Wiley & Sons, London (2007).
92. R. Codina and J. Blasco, "Analysis of a Pressure-Stabilized Finite Element Approximation of the Stationary Navier–Stokes Equations," *Numerische Matematik*, **87**, 59–81 (2000).
93. A. Masud and R. Khurram, "A Multiscale Finite Element Method for the Incompressible Navier–Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, **195**, 1750–1777 (2006).
94. F. Brezzi, L. Marini and E. Süli, "Residual-Free Bubbles for Advection-Diffusion Equations," *Numerische Matematik*, **85**, 31–47 (2000).
95. R. Codina, "Stabilized Finite Element Approximation of Transient Incompressible Flows Using Orthogonal Subscales," *Computer Methods in Applied Mechanics and Engineering*, **191**, 4295–4321 (2002).
96. C. H. Whiting and K. E. Jansen, "A Stabilized Finite Element Method for the Incompressible Navier–Stokes Equations Using a Hierarchical Basis," *International Journal for Numerical Methods in Fluids*, **35**, 93–116 (2001).
97. L. P. Franca and A. Nesliturk, "On a Two-Level Finite Element Method for the Incompressible Navier–Stokes Equations," *International Journal for Numerical Methods in Engineering*, **52**, 433–453 (2001).
98. V. Gravemeier, W. A. Wall and E. Ramm, "A Three-Level Finite Element Method for the Instationary Incompressible Navier–Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, **193**, 1323–1366 (2004).
99. C. R. Dohrmann, "Advection Stabilization," *Private Communication*, Sandia National Laboratories, Albuquerque NM (2008).
100. J. N. Reddy, *Applied Functional Analysis and Variational Methods in Engineering*, McGraw-Hill, New York (1986); reprinted by Krieger Publishing Company, Malabar, FL (1991).
101. P. B. Bochev, "Analysis of Least-Squares Finite Element Methods for the Navier–Stokes Equations," *SIAM Journal of Numerical Analysis*, **34**, 1817–1844 (1997).
102. P. B. Bochev and M. D. Gunzburger, "Finite Element Methods of Least-Squares Type," *SIAM Review* **40**, 789–837 (1998).
103. Z. Cai, T. A. Manteuffel, and S. F. McCormick, "First-Order System Least-Squares for the Stokes Equations, with Applications to Linear Elasticity," *SIAM Journal of Numerical Analysis*, **34**, 1727–1741 (1997).
104. M. M. J. Proot and M. I. Gerritsma, "Least-Squares Spectral Elements Applied to the Stokes Problem," *Journal of Computational Physics*, **181**, 454–477 (2002).
105. B. N. Jiang, "A Least-Squares Finite Element Method for Incompressible Navier–Stokes Problems," *International Journal for Numerical Methods in Fluids*, **14**, 843–859 (1992).
106. L. Q. Tang and T. H. Tsang, "Temporal, Spatial and Thermal Features of 3-D Rayleigh–Benard Convection by a Least-Squares Finite Element Method," *Computer Methods in Applied Mechanics and Engineering*, **140**, 201–219 (1997).
107. P. B. Bochev, Z. Cai, T. A. Manteuffel, and S. F. McCormick, "Analysis of Velocity-Flux First-Order System Least-Squares Principles for the Navier–Stokes Equations, Part I," *SIAM Journal of Numerical Analysis*, **35**, 990–1009 (1998).
108. B. N. Jiang and V. Sonnad, "Least-Squares Solution of Incompressible Navier–Stokes Equations with the p -version of Finite Elements," *Computational Mechanics* **15**, 129–136 (1994).
109. D. Winterscheidt and K. S. Surana, " p -version Least-Squares Finite Element Formulation for Two-Dimensional Incompressible Fluid Flow," *International Journal of Numerical Methods Fluids*, **18**, 43–69 (1994).

110. M. Bagheri and K. S. Surana, “ p -version Least-Squares Finite Element Formulation for Steady-State Two-Dimensional Turbulent Flows using the k - ε Model of Turbulence,” *Communications in Numerical Methods in Engineering*, **16**, 97–120 (2000).
111. V. Prabhakar, J. Pontaza, and J. N. Reddy, “A Collocation Penalty Least-squares Finite Element Formulation for Incompressible Flows,” *Computer Methods in Applied Mechanics and Engineering*, **197**, 449–463 (2008).
112. V. Prabhakar and J. N. Reddy, “Spectral/ hp Penalty Least-squares Finite Element Formulation for Unsteady Incompressible Flows,” *International Journal of Numerical Methods for Fluids*, **58**(3), 287–306 (2008).
113. B. C. Bell and K. S. Surana, “A Space-Time Coupled p -version Least-Squares Finite Element Formulation for Unsteady Fluid Dynamics Problems,” *International Journal for Numerical Methods in Engineering*, **37**, 3545–3569 (1994).
114. B. C. Bell and K. S. Surana, “A Space-Time Coupled p -version Least-Squares Finite Element Formulation for Unsteady Two-Dimensional Navier–Stokes Equations,” *International Journal for Numerical Methods in Engineering*, **39**, 2593–2618 (1996).
115. T. C. Warburton, S. J. Sherwin, and G. E. Karniadakis, “Basis Functions for Triangular and Quadrilateral High-Order Elements,” *SIAM Journal of Scientific Computing*, **20**, 1671–1695 (1999).
116. G. E. Karniadakis and S. J. Sherwin, *Spectral/ hp Element Methods for CFD*, Oxford University Press, Oxford (1999).
117. A. S. Grove, F. H. Shair, E. E. Petersen, and A. Acrivos, “An Experimental Investigation of the Steady Separated Flow Past a Circular Cylinder,” *Journal of Fluid Mechanics*, **19**, 60–80 (1964).
118. R. L. Sani and P. M. Gresho, “Resume and Remarks on the Open Boundary Condition Minisymposium,” *International Journal of Numerical Methods in Fluids*, **18**, 983–1008 (1994).
119. D. K. Gartling, “A Test Problem for Outflow Boundary Conditions - Flow over a Backward-Facing Step,” *International Journal for Numerical Methods in Fluids*, **11**, 953–967 (1990).
120. D. J. Naylor, “Stresses in Nearly Incompressible Materials by Finite Elements with Application to the Calculation of Excess Pore Pressures,” *International Journal for Numerical Methods in Engineering*, **8**, 443–460 (1974).
121. E. Hinton, F. C. Scott, and R. E. Ricketts, “Local Least Squares Stress Smoothing for Parabolic Isoparametric Elements,” *International Journal for Numerical Methods in Engineering*, **9**, 235–256 (1975).
122. J. -M. Hervouet, *Hydrodynamics of Free Surface Flows, Modeling with the Finite Element Method*, John Wiley & Sons, London (2007).
123. R. I. Tanner, R. E. Nickell, and R. W. Bilger, “Finite Element Methods for the Solution of Some Incompressible Non-Newtonian Fluid Mechanics Problems with Free Surfaces,” *Computer Methods in Applied Mechanics and Engineering*, **6**, 155–174 (1975).
124. K. R. Reddy and R. I. Tanner, “Finite Element Solution of Viscous Jet Flows with Surface Tension,” *Computers and Fluids*, **6**, 83–91 (1978).
125. O. C. Zienkiewicz, P. C. Jain, and E. Oñate, “Flow of Solids During Forming and Extrusion: Some Aspects of Numerical Solutions,” *International Journal of Solids and Structures*, **14**, 15–38 (1978).
126. H. Saito and L. E. Scriven, “Study of Coating Flow by the Finite Element Method,” *Journal of Computational Physics*, **42**, 53–76 (1981).
127. L. E. Scriven and S. F. Kistler, “Coating Flow Theory by Finite Element and Asymptotic Analysis of the Navier–Stokes System,” in *Finite Element Flow Analysis*, T. Kawai (ed.), University of Tokyo Press, Tokyo, 503–510 (1982).
128. M. Engelman, R. L. Sani, and P. M. Gresho, “The Implementation of Normal and/or Tangential Boundary Conditions in Finite Element Codes for Incompressible Fluid Flow,” *International Journal for Numerical Methods in Fluids*, **2**, 225–238 (1982).
129. K. N. Christodoulou and L. E. Scriven, “Discretization of Free Surface Flows and Other Moving Boundary Problems,” *Journal of Computational Physics*, **99**, 39–55 (1992).

130. P. M. Schweizer and S. F. Kistler (eds.), *Liquid Film Coating, Scientific Principles and Their Technical Implications*, Van Nostrand Reinhold, New York (1994).
131. D. R. Lynch and W. G. Gray, "Finite Element Simulation of Flow in Deforming Regions," *Journal of Computational Physics*, **36**, 135–153 (1980).
132. D. R. Lynch, "Unified Approach to Simulation on Deforming Elements with Application to Phase Change Problems," *Journal of Computational Physics*, **47**, 387–411 (1982).
133. P. R. Schunk, P. A. Sackinger, R. R. Rao, K. S. Chen, R. A. Cairncross, T. A. Baer and D. A. Labreche, "GOMA 2.0 – A Full-Newton Finite Element Program for Free and Moving Boundary Problems with Coupled Fluid/Solid Momentum, Energy, Mass and Chemical Species Transport: User's Guide," Sandia National Laboratories Report, SAND97-2404, Albuquerque, New Mexico (1998).
134. A. A. Johnson and T. E. Tezduyar, "Mesh Update Strategies in Parallel Finite Element Computations of Flow Problems with Moving Boundaries and Interfaces," *Computer Meth. App. Mech. and Engng.*, **119**, 73–94 (1994).
135. G. Chiandussi, G. Bugeda and E. Oñate, "A Simple Method for Automatic Updating of Finite Element Meshes," *Communications in Numerical Methods in Engineering*, **16**, 1–19 (2000).
136. D. K. Gartling, "MERLIN II – A Computer Program to Transfer Solution Data Between Finite Element Meshes," Sandia National Laboratories Report, SAND89-2989, Albuquerque, New Mexico (1991).
137. K. Jansen, F. Shakib and T. J. R. Hughes, "Fast Projection Algorithms for Unstructured Meshes," in *Computational Nonlinear Mechanics in Aerospace Engineering*, S. Atluri (ed.), AIAA, New York, 175–204 (1992).
138. R. L. Sani, P. M. Gresho, D. R. Tuerpe and R. L. Lee, "The Imposition of Incompressibility Constraints via Variational Adjustment of Velocity Fields," in *Proceedings of the First International Conference on Numerical Methods in Laminar and Turbulent Flow*, C. Taylor, K. Morgan and C. A. Brebbia (eds.), Pentech Press, London, 983–994 (1978).
139. W. J. Rider and D. B. Kothe, "Reconstructing Volume Tracking," *Journal of Computational Physics*, **141**, 112–152 (1998).
140. FIDAP Manual, Ver. 7.5, Fluid Dynamics International, Evanston, IL (1995).
141. J. A. Sethian, *Level Set Methods and Fast Marching Methods*, Second Edition, Cambridge University Press, Cambridge, United Kingdom (1999).
142. D. Adalsteinsson and J. A. Sethian, "The Fast Construction of Extensional Velocities in Level Set Methods," *Journal of Computational Physics*, **148**, 2–22 (1999).
143. M. Sussman, P. Smereka and S. Osher, "A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow," *Journal of Computational Physics*, **114**, 146–159 (1994).
144. J. O. Hinze, *Turbulence*, McGraw-Hill, New York (1975).
145. D. C. Leslie, *Developments in the Theory of Turbulence*, Oxford University Press, Oxford (1984).
146. P. Bradshaw (ed.), *Turbulence*, Second Edition, Springer-Verlag, Berlin (1978).
147. G. K. Batchelor, *The Theory of Homogeneous Turbulence*, Cambridge University Press, Cambridge, U.K. (1953).
148. A. S. Monin and A. M. Yaglom, *Statistical Fluid Mechanics*, MIT Press, Cambridge, MA (1973).
149. J. H. Ferziger, "Simulation of Incompressible Turbulent Flows," *Journal of Computational Physics*, **69**, 1–48 (1987).
150. J. M. Ferziger, "Higher Level Simulations of Turbulent Flows," in *Computational Methods for Turbulent, Transonic and Viscous Flows*, J. A. Essers (ed.), 93–182, Hemisphere Publishing, Washington, D.C. (1983).
151. P. T. Harsha, "Kinetic Energy Methods," in *Handbook of Turbulence*, W. Frost and T. Moulden (ed.), 187–236, Plenum Press, New York (1977).
152. J. D. Murphy, "Turbulence Modeling," in *Encyclopedia of Fluid Mechanics*, N. Chermisinoff (Ed.), **6**, 1131–1151, Gulf Publishing, Houston, Texas (1988).
153. M. Lesieur, *Turbulence in Fluids*, Martinus Nijhoff, Dordrecht, The Netherlands (1987).
154. A. Yoshizawa, "Large Eddy Simulation of Turbulent Flows," in *Encyclopedia of Fluid Mechanics*, N. Chermisinoff (ed.), **6**, 1277–1297, Gulf Publishing, Houston, Texas (1968).

155. R. S. Rogallo and P. Moin, "Numerical Simulation of Turbulent Flows," in *Annual Review of Fluid Mechanics*, **16**, 99–137 (1984).
156. P. Sagaut, *Large Eddy Simulation for Incompressible Flows, An Introduction*, Second Edition, Springer-Verlag, Berlin (2002).
157. V. John, *Large Eddy Simulation of Turbulent Incompressible Flows*, Springer-Verlag, Berlin (2004).
158. W. Rodi, "Turbulence Models and Their Applications to Hydraulics," IAHR Report, Delft, The Netherlands (1984).
159. T. Cebeci and A. M. O. Smith, "Analysis of Turbulent Boundary Layers," *Applied Mathematics and Mechanics*, Academic Press, New York (1974).
160. V. Haroutunian, "Turbulent Flows with FIDAP," Fluid Dynamics International Seminar Notes, Evanston, Illinois (1988).
161. B. Mohammadi and O. Pironneau, *Analysis of the K-Epsilon Turbulence Model*, John Wiley & Sons, New York (1994).
162. M. Nallasamy, "Turbulence Models and Their Applications to the Prediction of Internal Flows: A Review," *Computers and Fluids*, **15**, 151–194(1987).
163. R. M. Smith, "On Finite Element Calculations of Turbulent Flow Using the $k-\epsilon$ Model," *International Journal for Numerical Methods in Fluids*, **4**, 303–319 (1984).
164. R. M. Smith, "A Practical Method of Two Equation Modelling Using Finite Elements," *International Journal for Numerical Methods in Fluids*, **4**, 321–336 (1984).
165. A. G. Hutton, R. M. Smith and S. Hickmott, "The Computation of Turbulent Flows of Industrial Complexity by Finite Element Methods-Progress and Prospects," in *Finite Elements in Fluids*, Vol. 7, R. H. Gallagher, et al. (eds.), John Wiley & Sons, New York (1988).
166. B. E. Launder and D. B. Spalding, *Lectures in Mathematical Models of Turbulence*, Academic Press, London, U.K. (1972).
167. B. E. Launder and D. B. Spalding, "The Numerical Computation of Turbulent Flow," *Computer Methods in Applied Mechanics and Engineering*, **23**, 249–270 (1974).
168. FIDAP Manual, Version 6.0, Fluid Dynamics International, Evanston, Illinois (1991).
169. A. G. Hutton, "Finite Element Boundary Techniques for Improved Performance in Computing Navier–Stokes and Related Heat Transfer Problems," in *Finite Elements in Fluids*, Vol. 4, R. H. Gallagher, et al. (eds.) John Wiley & Sons, New York (1982).
170. B. Koobus and C. Farhat, "A Variational Multiscale Method for the Large Eddy Simulation of Compressible Turbulent Flows on Unstructured Meshes - Application to Vortex Shedding," *Computer Methods in Applied Mechanics and Engineering*, **193**, 1367–1383 (2004).
171. T. J. R. Hughes, A. Oberai and L. Mazzie, "Large Eddy Simulation of Turbulent Channel Flows by the Variational Multiscale Method," *Physics of Fluids*, **13**, 1784–1799 (2001).
172. T. J. R. Hughes, G. Wells and A. Wray, "Energy Transfer and Spectral Eddy Viscosity in Large Eddy Simulations of Homogeneous Isotropic Turbulence," *Physics of Fluids*, **16**, 4044–4052 (2004).
173. V. Gravemeier, W. Wall and E. Ramm, "Large Eddy Simulation of Turbulent Incompressible Flows by a Three-Level Finite Element Method," *International Journal for Numerical Methods in Fluids*, **48**, 1067–1099 (2005).
174. Y. Bazilevs, V. M. Calo, J. A. Cottrell, T. J. R. Hughes, A. Reali and G. Scovazzi, "Variational Multiscale Residual-Based Turbulence Modeling for Large Eddy Simulation of Incompressible Flows," *Computer Methods in Applied Mechanics and Engineering*, **197**, 173–201 (2007).
175. Y. Bazilevs, C. Michler, V. M. Calo and T. J. R. Hughes, "Weak Dirichlet Boundary Conditions for Wall-Bounded Turbulent Flows," *Computer Methods in Applied Mechanics and Engineering*, **196**, 4853–4862 (2007).
176. D. K. Gartling, "NACHOS II – A Finite Element Computer Program for Incompressible Flow Problems," Sandia National Laboratories Report, SAND86-1816 and SAND86-1817, Albuquerque, New Mexico (1987).
177. A. Nadai, *Theory of Flow and Fracture of Solids, Vol. II*, McGraw-Hill, New York (1963).

178. H. Schlichting, *Boundary-Layer Theory* (translated by J. Kestin), Seventh Edition, McGraw-Hill, New York (1979).
179. O. R. Burggraf, "Analytical and Numerical Studies of the Structure of Steady Separated Flows," *Journal of Fluid Mechanics*, **24**(1), 113–151 (1966).
180. F. Pan and A. Acrivos, "Steady Flow in Rectangular Cavities," *Journal of Fluid Mechanics*, **28**, 643–655 (1967).
181. R. T. Cheng, "Numerical Solution of the Navier-Stokes Equations by the Finite Element Method," *Physics of Fluids*, **15** (12), 2098–2105 (1972).
182. G. D. Mallison and G. de Vahl Davis, "The Method of False Transient for the Solution of Coupled Elliptic Equations," *Journal of Computational Physics*, **12**, 435–461 (1967).
183. G. de Vahl Davis and G. D. Mallison, "An Evaluation of Upwind and Central Difference Approximations by a Study of Recirculating Flow," *Computers and Fluids*, **4**, 29–43 (1976).
184. H. Takami and K. Kuwahara, "Numerical Study of Three-Dimensional Flow Within a Cubic Cavity," *Journal of Physical Society of Japan*, **37**, 1695–1698 (1974).
185. J. N. Reddy, "Penalty Finite Element Methods for the Solution of Advection and Free Convection Flows," *Finite Element Methods in Engineering*, A. P. Kabaila and V. A. Pulmano (eds.), University of New South Wales, Sydney, Australia, 583–598 (1979).
186. U. Ghia, K. N. Ghia, and C. T. Shin, "High-Re Solution for Incompressible Flow using the Navier-Stokes Equations and the Multigrid Method," *Journal of Computational Physics*, **48**, 387–411 (1982).
187. S. C. R. Dennis, D. B. Ingham, and R. N. Cook, "Finite Difference Methods for Calculating Steady Incompressible Flows in Three Dimensions," *Journal of Computational Physics*, **33**, 325–339 (1979).
188. R. K. Agarwal, "A Third-Order Accurate Upwind Scheme for Navier-Stokes Solutions in Three Dimensions," McDonnell Douglas Research Laboratory, Report No. MDRL 81–20, St. Louis, Missouri (1981).
189. J. N. Reddy, "Penalty-Finite-Element Analysis of 3-D Navier-Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, **35**, 87–106 (1982).
190. H. P. Langtangen and A. Tevito, "A Numerical Comparison of Conjugate Gradient-Like Methods," *Communications in Applied Numerical Methods*, **34**, 793–798 (1988).
191. R. C. Givler, D. K. Gartling, M. S. Engelman, and V. Haroutunian, "Navier-Stokes Simulations of Flow Past Three-Dimensional Submarine Models," *Computer Methods in Applied Mechanics and Engineering*, **87**, 175–200 (1991).
192. P. A. Sackinger, "Flows and Transitions During Solidification: Simulation of Hydrodynamics, Heat Transfer and Free Boundaries in Czochralski Growth," Ph.D. Dissertation, Dept. of Chem. Engr., MIT, Cambridge, MA (1989).
193. P. A. Sackinger, R. A. Brown, and J. J. Derby, "A Finite Element Method for Analysis of Fluid Flow, Heat Transfer and Free Interfaces in Czochralski Crystal Growth," *International Journal for Numerical Methods in Fluids*, **9**, 453–492 (1989).
194. ProCAST User's Manual, Version 2.0, *UES, Inc.*, Dayton, Ohio (1992).
195. L. S. G. Kovasznay, "Laminar flow behind a two-dimensional grid," *Proceedings of Cambridge Philosophical Society*, **44** 58–62 (1948).
196. Tritton, D. J., "Experiments on the Flow Past a Circular Cylinder at Low Reynolds Numbers," *Journal of Fluid Mechanics*, **6**, 547–567 (1959).
197. Dennis, S. C. R. and Chang, G. Z., "Numerical Solutions for Steady Flow Past a Circular Cylinder at Reynolds Numbers up to 100," *Journal of Fluid Mechanics*, **42**, 471–489 (1970).
198. Kawaguti, M. and Jain, P., "Numerical Study of a Viscous Fluid Past a Circular Cylinder," *Journal of the Physical Society of Japan*, **21**, 2055–2063 (1966).

Coupled Fluid Flow and Heat Transfer

5.1 Introduction

The general problem area of convective heat transfer is a particularly challenging one since it represents a situation involving inherently coupled problems, i.e., problems involving multiple physical phenomena. Indeed, the subject itself represents a joining of two classical areas of applied mechanics, namely, fluid mechanics and heat transfer.

Convection problems are generally divided into two major categories depending primarily on the forces that are responsible for the fluid motion. In *forced convection*, the fluid motion is due to the application of pressure or viscous forces on the fluid boundary. *Free* or *natural convection* problems are characterized by the fluid motion which is produced by temperature-induced buoyancy forces. This distinction between the types of convection is not always possible because the two types of driving forces (surface and volume) may appear in varying degrees and combinations.

Another categorization of convection problems may be defined based on the compressibility of the fluid. The most widely studied model for convection utilizes the Boussinesq approximation which assumes an incompressible flow and limits density variations to be small and isolated to the body force term. In a strict Boussinesq model, thermophysical properties are treated as constant, work done by pressure and viscous dissipation are neglected, and the density in the body force term is a linear function of the temperature. The Boussinesq model is computationally convenient because the incompressible viscous flow equations only require minimal alteration for the inclusion of nonisothermal effects. However, the Boussinesq approximation is limited to flows with relatively small temperature differences as outlined in Section 1.7, and this limitation has forced consideration of other low-speed convection models.

Low-speed compressible flows or non-Boussinesq models form another category of convection problems. The standard model in this regime involves the so-called acoustically filtered equations which admit large density (temperature) variations but suppress the propagation of acoustic waves. The resolution of small amplitude (acoustic) pressure disturbances is computationally troublesome and one of the major difficulties in using fully compressible flow formulations at the low-speed or low Mach number limit. The acoustically filtered model may be derived by any of several formal methods [1–3] and these were outlined in Section 1.7. A comparison and comments on the various non-Boussinesq models can also be found in [3,4]. Unfortunately, the finite element methods for this model are not quite the immediate extension of the incompressible methods from the previous chapter.

Regardless of the categorization, the convective heat transfer process depends directly on the motion of a fluid medium. The Navier–Stokes equations from Chapter 4 (or their extension to the compressible case) as well as the advection-diffusion equation that describes thermal energy transport must be considered in the development of a computational scheme. These equations consist of a set of coupled partial differential equations in terms of the velocity field, pressure, and temperature.

In this chapter, finite element models based on the weak formulation of the governing equations of viscous incompressible fluids in the presence of buoyancy forces and inertial effects are developed. Finite element models for nonisothermal porous flow and low-speed compressible flows are also formulated. The finite element model development for this coupled case is completely analogous to the developments for heat conduction in Chapter 3 and viscous incompressible flows in Chapter 4. Therefore, detailed discussions of the actual model development will be less extensive in the present chapter.

5.2 Nonisothermal Incompressible Flows

5.2.1 Governing Equations

The laws describing the nonisothermal flow of an incompressible Newtonian fluid were presented in Chapter 1 and are summarized here for ready reference. The equations are written for a fluid region Ω_f using a Cartesian coordinate system x_i in a Eulerian reference frame, with the index i taking the values $i = 1, 2, 3$ for the three-dimensional case and $i = 1, 2$ for the two-dimensional case; the usual summation convention on repeated indices is used [see Eqs. (1.5.7)–(1.5.9)]; time is denoted by t . An extended form of the Boussinesq approximation is used, which allows the fluid properties to be functions of the thermodynamic state (e.g., pressure P and temperature T) and the density ρ to vary with temperature T according to the relation

$$\rho = \rho_0[1 - \beta(T - T_0)] \quad (5.2.1)$$

where β is the coefficient of thermal expansion and the subscript zero indicates a reference condition. The variation of density as given in (5.2.1) is permitted only in the description of the body force; the density in all other situations is assumed to be that of the reference state, ρ_0 . The governing equations of convective heat transfer are summarized below.

Conservation of Mass

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (5.2.2)$$

Conservation of Momentum

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) - \frac{\partial}{\partial x_j} \left[-P\delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 g_i \beta(T - T_0) = 0 \quad (5.2.3)$$

Conservation of Energy

$$\rho_0 C_v \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} \right) - \frac{\partial}{\partial x_i} \left(k_{ij} \frac{\partial T}{\partial x_j} \right) - Q - \Phi = 0 \quad (5.2.4)$$

where v_i denote the velocity components, P the pressure, ρ_0 the density, g_i the gravitational force components, T the temperature, C_v the specific heat of the fluid at constant volume, Q the rate of internal heat generation, μ the shear viscosity of the fluid, k_{ij} the components of the thermal conductivity tensor, and Φ is the viscous dissipation in the fluid

$$\Phi = 2\mu D_{ij}D_{ij}; \quad D_{ij} = \frac{1}{2} \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \quad (5.2.5)$$

Recall that Eq. (5.2.2) is also known as the continuity equation or divergence-free condition on the velocity field and relations in Eq. (5.2.3) are known as the Navier–Stokes equations. The equations in (5.2.3) and (5.2.4) are written in advective form.

The advective form of the Boussinesq equations is the standard form for finite element computation. Other discretization methods (particularly finite difference and finite volume methods) have advocated the use of the conservation forms of the equations that conserve various quantities such as kinetic energy, enstrophy, temperature, and temperature squared. It has been observed that the conservation form minimizes aliasing errors and improves temporal stability for these computational schemes. Global conservation of these types of quantities is often useful in proving boundedness and stability of the continuum equations when such proofs are available. Though the theory is incomplete for the continuum, it is often assumed that if the discrete method shares the same conservation properties as the continuum, the discrete solution will behave properly and remain a good approximation to the partial differential equations. The need for the conservation form in a finite element method is uncertain. Gresho, et al. [5] and Cliffe [6] conducted finite element studies of the inviscid Boussinesq equations and determined what forms conserved which quantities and produced the more stable forms. Note, however, that there are limits on what combinations of linear and quadratic quantities may be conserved within any given scheme; discrete systems are limited to conservation of quadratic quantities. Also, because physical dissipation is normally included in convection problems, the need for the conservation form remains unclear. Here a general form of the advection operator (introduced in Section 1.4.7) is used to allow an easy transition from advection to conservation to an “absolute” conservation form. Using a form of (1.4.31) for the incompressible, Boussinesq equations, the balance equations can be written as

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (5.2.6)$$

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} + \alpha_v v_i \frac{\partial v_j}{\partial x_j} \right) - \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 g_i \beta (T - T_0) = 0 \quad (5.2.7)$$

$$\rho_0 C_v \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} + \alpha_T T \frac{\partial v_j}{\partial x_j} \right) - \frac{\partial}{\partial x_i} \left(k_{ij} \frac{\partial T}{\partial x_j} \right) - Q - \Phi = 0 \quad (5.2.8)$$

The parameters α_v and α_T may have values of 0, 1, and 1/2 which produce the advective, conservation, and absolute conservation forms of the momentum and energy equations, respectively. Note that the difference between the advective and

conservative equations differ by the occurrence of a moment of the incompressibility constraint. In the continuum, the continuity equation is satisfied exactly and the advective and conservation forms are identical. Because the continuity equation is only satisfied approximately in a computational scheme, the two forms differ in any numerical method. Also, because the α parameters may be selected independently, a number of possible combinations of advective and conservation forms are possible. This also implies that the quantities conserved by the discrete method will differ with choices of α .

Though the demonstration of what quantities are conserved with particular selections of α is not difficult, it need not be repeated here; both references [5] and [6] have extensive discussions of the process, assumptions and results. As we are interested in Boussinesq systems, only a few results will be quoted here for standard mixed finite element methods using a coupled method. For $\alpha_v = \alpha_T = 0$, advective forms are used in both equations and nothing is conserved. If $\alpha_v = \alpha_T = 1$, then the conservation (divergence) form is employed and the global momentum and temperature are conserved. The combination of $\alpha_T = 1/2$ with any allowed value of α_v gives a mixed equation form and ensures conservation of the temperature squared; kinetic energy is conserved for an isothermal flow if $\alpha_v = 1/2$. Many other combinations are possible and additional considerations may be introduced if special pressure interpolation functions are introduced as in [6]. It is the general consensus that conservation of temperature squared is the most important quantity with regard to stability of the inviscid Boussinesq system and therefore $\alpha_v = \alpha_T = 1/2$ are the preferred parameters. However, the benefit of using the absolute conservation form when dissipation is incorporated appears to be minimal.

5.2.2 Boundary Conditions

The boundary conditions are given by

$$v_i = f_i^v(s_k, t) \quad \text{on } \Gamma_v \quad (5.2.9a)$$

$$T_i \equiv \sigma_{ij}(s_k, t) n_j(s_k) = f_i^T(s_k, t) \quad \text{on } \Gamma_T \quad (5.2.9b)$$

for the fluid mechanics part of the problem, and

$$T = f^T(s_k, t) \quad \text{on } \Gamma_T \quad (5.2.10a)$$

$$-\left(k_{ij}\frac{\partial T}{\partial x_j}\right)n_i \equiv q_i n_i = q_c + q_r = q_a = f^q(s_k, t) \quad \text{on } \Gamma_q \quad (5.2.10b)$$

for the heat transfer part of the problem. In Eq. (5.2.9b), σ_{ij} denote the components of the total stress tensor (i.e., viscous and hydrostatic)

$$\sigma_{ij} = \mu\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right) - P\delta_{ij} \quad (5.2.11)$$

We will not explicitly consider the energy equation for solid regions in this chapter because it was considered in detail in Chapter 3. However, it should be recognized that the inclusion of solid body conduction regions in any convection problem (i.e., the so-called conjugate problem) is a relatively straightforward procedure. Also, as

mentioned in Chapter 1, convection problems may require the addition of auxiliary variables to describe all of the physical phenomena of interest. We will not consider this case explicitly since the formulation and solution methods for transport equations follow exactly the treatment of the energy equation. Note, however, that inclusion of auxiliary transport equations may significantly increase the magnitude of the computational effort.

There are five partial differential equations expressed in terms of five unknowns (v_1, v_2, v_3, P, T) for a three-dimensional problem. Following the developments of Chapters 3 and 4, we shall consider two finite element models of Eqs. (5.2.1)–(5.2.4). The first one is the *velocity-pressure-temperature model* or *mixed model*. The second model is the *penalty-finite element model*. These models were discussed in Chapter 4 for isothermal flows. Extension of these models to convective heat transfer problems is straightforward, as will be shown in this chapter. Note also that the general conservation form of the Boussinesq system, as given in (5.2.6)–(5.2.8), may also be used in the development of a finite element model with the addition of two terms.

5.3 Mixed Finite Element Model

Let us denote expressions on the left side of the equalities in (5.2.2)–(5.2.4) by f_1, \mathbf{f}_2 , and f_3 , respectively. As discussed in Section 4.2, the weighted-integral statements of the three equations over a typical element Ω^e are given by

$$\int_{\Omega^e} w_1 f_1 d\mathbf{x} = 0, \quad \int_{\Omega^e} \mathbf{w}_2 \cdot \mathbf{f}_2 d\mathbf{x} = 0, \quad \int_{\Omega^e} w_3 f_3 d\mathbf{x} = 0 \quad (5.3.1)$$

where (w_1, \mathbf{w}_2, w_3) are the weight functions, which will be equated to the interpolation functions used for (P, \mathbf{v}, T) , respectively. It should be recalled from Chapters 3 and 4 that the weighted-integral statements in Eq. (5.3.1) are reduced to weak statements by integrating the viscous and diffusion parts. All other terms are kept as they are.

Suppose that the dependent variables (T, v_i, P) are approximated by expansions of the form

$$T(\mathbf{x}, t) = \sum_{m=1}^M \theta_m(\mathbf{x}) T_m(t) = \boldsymbol{\Theta}^T \mathbf{T} \quad (5.3.2a)$$

$$v_i(\mathbf{x}, t) = \sum_{n=1}^N \psi_n(\mathbf{x}) v_i^n(t) = \boldsymbol{\Psi}^T \mathbf{v}_i \quad (5.3.2b)$$

$$P(\mathbf{x}, t) = \sum_{l=1}^L \phi_l(\mathbf{x}) P_l(t) = \boldsymbol{\Phi}^T \mathbf{P} \quad (5.3.2c)$$

where $\boldsymbol{\Theta}$, $\boldsymbol{\Psi}$, and $\boldsymbol{\Phi}$ are vectors of interpolation (or shape) functions, \mathbf{T} , \mathbf{v}_i , and \mathbf{P} are vectors of nodal values of temperature, velocity components, and pressure, respectively, and superscript $(\cdot)^T$ denotes a transpose of the enclosed vector or matrix. The weight functions (w_1, \mathbf{w}_2, w_3) have the following correspondence (see Reddy [7] for further details)

$$w_1 \approx \phi_l, \quad \mathbf{w}_2 \approx \boldsymbol{\psi}_n, \quad w_3 \approx \theta_m \quad (5.3.3)$$

Substitution of Eqs. (5.3.2) and (5.3.3) into the weak forms associated with Eq. (5.3.1) results in the following finite element equations:

Continuity

$$-\left[\int_{\Omega^e} \Phi \frac{\partial \Psi^T}{\partial x_i} dx \right] \mathbf{v}_i = \mathbf{0} \quad (5.3.4)$$

Momentum

$$\begin{aligned} & \left[\int_{\Omega^e} \rho_0 \Psi \Psi^T dx \right] \dot{\mathbf{v}}_i + \left[\int_{\Omega^e} \rho_0 \Psi (\Psi^T \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} dx \right] \mathbf{v}_i + \left[\int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_j} dx \right] \mathbf{v}_i \\ & + \left[\int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_i} dx \right] \mathbf{v}_j - \left[\int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T dx \right] \mathbf{P} \\ & = - \left[\int_{\Omega^e} \rho_0 g_i \beta \Psi \Theta^T dx \right] \mathbf{T} + \left\{ \int_{\Omega^e} \rho_0 g_i \beta \Psi T_0 dx \right\} + \left\{ \oint_{\Gamma^e} \Psi \mathcal{T}_i ds \right\} \end{aligned} \quad (5.3.5)$$

Energy

$$\begin{aligned} & \left[\int_{\Omega^e} \rho_0 C_v \Theta \Theta^T dx \right] \dot{\mathbf{T}} + \left[\int_{\Omega^e} \rho_0 C_v \Theta (\Psi^T \mathbf{v}_j) \frac{\partial \Theta^T}{\partial x_j} dx \right] \mathbf{T} + \left[\int_{\Omega^e} k_{ij} \frac{\partial \Theta}{\partial x_i} \frac{\partial \Theta^T}{\partial x_j} dx \right] \mathbf{T} \\ & = \left\{ \int_{\Omega^e} \Theta Q dx \right\} + \left\{ \int_{\Omega^e} \Theta \Phi dx \right\} + \left\{ \oint_{\Gamma^e} \Theta q ds \right\} \end{aligned} \quad (5.3.6)$$

or

$$\begin{aligned} & \begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}}_1 \\ \dot{\mathbf{v}}_2 \\ \dot{\mathbf{v}}_3 \\ \dot{\mathbf{P}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P} \end{Bmatrix} + \\ & \begin{bmatrix} \hat{\mathbf{K}}_{11} & \mathbf{K}_{21} & \mathbf{K}_{31} & -\mathbf{Q}_1 \\ \mathbf{K}_{12} & \hat{\mathbf{K}}_{22} & \mathbf{K}_{32} & -\mathbf{Q}_2 \\ \mathbf{K}_{13} & \mathbf{K}_{23} & \hat{\mathbf{K}}_{33} & -\mathbf{Q}_3 \\ -\mathbf{Q}_1^T & -\mathbf{Q}_2^T & -\mathbf{Q}_3^T & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_1(\mathbf{T}) \\ \mathbf{F}_2(\mathbf{T}) \\ \mathbf{F}_3(\mathbf{T}) \\ \mathbf{0} \end{Bmatrix} \end{aligned} \quad (5.3.7)$$

$$\mathbf{N} \dot{\mathbf{T}} + \mathbf{D}(\mathbf{v}) \mathbf{T} + \mathbf{L} \mathbf{T} = \mathbf{G}(\mathbf{T}) \quad (5.3.8)$$

The coefficient matrices shown in Eqs. (5.3.7) and (5.3.8) are defined by

$$\begin{aligned} \hat{\mathbf{K}}_{11} &= 2\mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} \\ \hat{\mathbf{K}}_{22} &= \mathbf{K}_{11} + 2\mathbf{K}_{22} + \mathbf{K}_{33} \\ \hat{\mathbf{K}}_{33} &= \mathbf{K}_{11} + \mathbf{K}_{22} + 2\mathbf{K}_{33} \end{aligned} \quad (5.3.9a)$$

$$\begin{aligned} \mathbf{M} &= \int_{\Omega^e} \rho_0 \Psi \Psi^T dx ; \quad \mathbf{C}(\mathbf{v}) = \int_{\Omega^e} \rho_0 \Psi (\Psi^T \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} dx \\ \mathbf{K}_{ij} &= \int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} dx ; \quad \mathbf{Q}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T dx \\ \mathbf{F}_i(\mathbf{T}) &= - \int_{\Omega^e} \rho_0 g_i \beta \Psi \Psi^T \mathbf{T} dx + \int_{\Omega^e} \rho_0 g_i \beta \Psi T_0 dx + \oint_{\Gamma^e} \Psi \mathcal{T}_i ds \end{aligned}$$

$$\begin{aligned}\mathbf{D}(\mathbf{v}) &= \int_{\Omega^e} \rho_0 C \Theta (\Psi^T \mathbf{v}_j) \frac{\partial \Theta^T}{\partial x_j} d\mathbf{x} \\ \mathbf{N} &= \int_{\Omega^e} \rho_0 C_v \Theta \Theta^T d\mathbf{x}; \quad \mathbf{L} = \int_{\Omega^e} k \frac{\partial \Theta}{\partial x_i} \frac{\partial \Theta^T}{\partial x_i} d\mathbf{x} \\ \mathbf{G} &= \int_{\Omega^e} \Theta Q d\mathbf{x} + \int_{\Omega^e} \Theta \Phi d\mathbf{x} + \oint_{\Gamma^e} \Theta q ds\end{aligned}\quad (5.3.9b)$$

where summation on repeated indices is implied. The above equations can be written symbolically as follows.

Continuity

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (5.3.10)$$

Momentum

$$\mathbf{M} \dot{\mathbf{v}} + \mathbf{Cv} + \mathbf{Kv} - \mathbf{QP} + \mathbf{BT} = \mathbf{F} \quad (5.3.11)$$

Energy

$$\mathbf{NT} + \mathbf{DT} + \mathbf{LT} = \mathbf{G} \quad (5.3.12)$$

where the superposed dot represents a time derivative and $\mathbf{v}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T\}$. In writing Eq. (5.3.11) the buoyancy term \mathbf{B} has been separated from the general force vector \mathbf{F} . The expression \mathbf{BT} has the meaning

$$\mathbf{BT} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{B}_3 \end{bmatrix} \begin{Bmatrix} \mathbf{T} \\ \mathbf{T} \\ \mathbf{T} \end{Bmatrix} \quad (5.3.13a)$$

where

$$\mathbf{B}_i = \int_{\Omega^e} \rho_0 \beta g_i \Psi^T \Theta d\mathbf{x} \quad (5.3.13b)$$

Note that the solid body conduction equation of Chapter 3 is obtained from (5.3.12) by neglecting the transport term \mathbf{D} .

Equations (5.3.10)–(5.3.12) can be combined into a single matrix equation

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{N} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}} \\ \dot{\mathbf{P}} \\ \dot{\mathbf{T}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{v}) + \mathbf{K}(\mathbf{T}) & -\mathbf{Q} & \mathbf{B}(\mathbf{T}) \\ -\mathbf{Q}^T & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}(\mathbf{v}) + \mathbf{L}(\mathbf{T}) \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \\ \mathbf{T} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}(\mathbf{T}) \\ \mathbf{0} \\ \mathbf{G}(\mathbf{T}, \mathbf{u}) \end{Bmatrix} \quad (5.3.14)$$

The viscous and diffusion terms are indicated as being functions of temperature to accommodate the case of temperature dependent viscosity and conductivity; other functional dependencies are also possible. In a more symbolic format, Eq. (5.3.14) can be expressed as

$$\bar{\mathbf{M}} \dot{\mathbf{U}} + \bar{\mathbf{K}} \mathbf{U} = \bar{\mathbf{F}} \quad (5.3.15a)$$

where

$$\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{P}^T, \mathbf{T}^T\} \quad (5.3.15b)$$

Note that the general form of Eq. (5.2.15a) is the same as the nonlinear diffusion Eq. (3.6.3) or the incompressible flow system in Eq. (4.2.15a). Therefore, the time-approximations discussed in Chapters 3 and 4 are readily applicable to the ordinary differential equations in (5.2.15a).

The derivation to this point has considered the advective form of the equations. If conservation forms for the momentum and energy equations are desired, Eqs. (5.2.6)–(5.2.8) may be used for the GFEM. The result is the same as the above equations with additional convection terms $\mathbf{C}^\alpha \mathbf{v}$ and $\mathbf{D}^\alpha \mathbf{v}$ occurring in the momentum and energy equations. Using the above notation, these additional matrices are defined by

$$\mathbf{C}^\alpha \mathbf{v} = \begin{bmatrix} \mathbf{C}^\alpha_1(\mathbf{v}_1) & \mathbf{C}^\alpha_2(\mathbf{v}_1) & \mathbf{C}^\alpha_3(\mathbf{v}_1) \\ \mathbf{C}^\alpha_1(\mathbf{v}_2) & \mathbf{C}^\alpha_2(\mathbf{v}_2) & \mathbf{C}^\alpha_3(\mathbf{v}_2) \\ \mathbf{C}^\alpha_1(\mathbf{v}_3) & \mathbf{C}^\alpha_2(\mathbf{v}_3) & \mathbf{C}^\alpha_3(\mathbf{v}_3) \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{Bmatrix} \quad (5.3.16a)$$

where

$$\mathbf{C}_j^\alpha(\mathbf{v}_i) = \alpha_v \int_{\Omega^e} \rho_0 \mathbf{\Psi} (\mathbf{\Psi}^T \mathbf{v}_i) \frac{\partial \mathbf{\Psi}^T}{\partial x_j} d\mathbf{x} \quad (5.3.16b)$$

and

$$\mathbf{D}^\alpha \mathbf{v} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{D}_2 & \mathbf{D}_3 \\ \mathbf{D}_1 & \mathbf{D}_2 & \mathbf{D}_3 \\ \mathbf{D}_1 & \mathbf{D}_2 & \mathbf{D}_3 \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{Bmatrix} \quad (5.3.17a)$$

where

$$\mathbf{D}_j^\alpha(\mathbf{T}) = \alpha_T \int_{\Omega^e} \rho_0 C \mathbf{\Theta} (\mathbf{\Theta}^T \mathbf{T}) \frac{\partial \mathbf{\Theta}^T}{\partial x_j} d\mathbf{x} \quad (5.3.17b)$$

These forms provide entries for each velocity component \mathbf{v}_j in the momentum and energy equations, respectively. These terms are easily constructed and add little cost to the overall formulation. The α terms may complicate slightly the solution procedure as they add some cross-coupling matrices in the Jacobian of a Newton procedure as outlined in Section 5.7.

The three sets of interpolation functions used in (5.3.2) should be of the Lagrange type, i.e., derived by interpolating only the values of the functions, and not their derivatives. There are three different finite elements associated with the three field variables (v_i , P , T). Although, in principle, one can use different interpolation functions for the velocity field and temperature, it is practical to use the same type of interpolation for the two field variables (T, v_i). The choice of interpolation functions used for the pressure variable (see Section 4.5.2) in the mixed finite element model is constrained by the special role the pressure plays in incompressible flows. In order to prevent an over-constrained system of discrete equations, the interpolation used for pressure must be at least one order lower than that used for the velocity field (i.e., unequal interpolation). Further, pressure need not be made continuous across elements because the pressure variable does not constitute a primary variable of the mixed model.

For two-dimensional flows of viscous incompressible fluids, often the Lagrange quadratic interpolation is used for the velocity components. Two different pressure approximations are available when the velocities are approximated by quadratic Lagrange functions. The first is a continuous-bilinear approximation, in which the pressure is defined at the corner nodes of the element and is made continuous across

element boundaries. The second pressure approximation involves a discontinuous, linear variation defined on the element by

$$\Phi = \begin{Bmatrix} 1 \\ x \\ y \end{Bmatrix} \quad (5.3.18)$$

Here the unknowns are not nodal point values of the pressure but correspond to the coefficients in $P = a \cdot 1 + b \cdot x + c \cdot y$. A standard element used in the analysis of three-dimensional viscous flow problems is the eight-node, hexahedron (brick) element, where the velocity components are approximated using the trilinear Lagrange functions and pressure is a constant (and obviously discontinuous between elements).

5.4 Penalty Finite Element Model

5.4.1 Preliminary Comments

Recall that the velocity field of viscous incompressible flows must satisfy the momentum Eqs. (5.2.3) and in addition the continuity Eq. (5.2.2). Equation (5.2.2) is treated as a constraint on the velocity field. In the penalty function method, the constrained problem is reformulated as an unconstrained problem. The discussion presented in Section 4.3 is applicable here with the body force components f_i in equations (4.3.16a-c) replaced with the buoyancy forces, $g_i\beta(T - T_0)$. The weak statements associated with the penalty formulation of convective heat transfer are as follows:

Continuity and Momentum

$$\begin{aligned} 0 &= \int_{\Omega^e} \rho_0 \delta v_1 \left[\frac{\partial v_1}{\partial t} + \left(v_1 \frac{\partial v_1}{\partial x_1} + v_2 \frac{\partial v_1}{\partial x_2} + v_3 \frac{\partial v_1}{\partial x_3} \right) - g_1 \beta(T - T_0) \right] d\mathbf{x} \\ &\quad + \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_1}{\partial x_1} \frac{\partial v_1}{\partial x_1} + \mu \frac{\partial \delta v_1}{\partial x_2} \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) + \mu \frac{\partial \delta v_1}{\partial x_3} \left(\frac{\partial v_1}{\partial x_3} + \frac{\partial v_3}{\partial x_1} \right) \right] d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_1 T_1 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_1}{\partial x_1} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) d\mathbf{x} \end{aligned} \quad (5.4.1a)$$

$$\begin{aligned} 0 &= \int_{\Omega^e} \rho_0 \delta v_2 \left[\frac{\partial v_2}{\partial t} + \left(v_1 \frac{\partial v_2}{\partial x_1} + v_2 \frac{\partial v_2}{\partial x_2} + v_3 \frac{\partial v_2}{\partial x_3} \right) - g_2 \beta(T - T_0) \right] d\mathbf{x} \\ &\quad + \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_2}{\partial x_2} \frac{\partial v_2}{\partial x_2} + \mu \frac{\partial \delta v_2}{\partial x_1} \left(\frac{\partial v_1}{\partial x_2} + \frac{\partial v_2}{\partial x_1} \right) + \mu \frac{\partial \delta v_2}{\partial x_3} \left(\frac{\partial v_2}{\partial x_3} + \frac{\partial v_3}{\partial x_2} \right) \right] d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_2 T_2 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_2}{\partial x_2} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) d\mathbf{x} \end{aligned} \quad (5.4.1b)$$

$$\begin{aligned} 0 &= \int_{\Omega^e} \rho_0 \delta v_3 \left[\frac{\partial v_3}{\partial t} + \left(v_1 \frac{\partial v_3}{\partial x_1} + v_2 \frac{\partial v_3}{\partial x_2} + v_3 \frac{\partial v_3}{\partial x_3} \right) - g_3 \beta(T - T_0) \right] d\mathbf{x} \\ &\quad + \int_{\Omega^e} \left[2\mu \frac{\partial \delta v_3}{\partial x_3} \frac{\partial v_3}{\partial x_3} + \mu \frac{\partial \delta v_3}{\partial x_1} \left(\frac{\partial v_1}{\partial x_3} + \frac{\partial v_3}{\partial x_1} \right) + \mu \frac{\partial \delta v_3}{\partial x_2} \left(\frac{\partial v_2}{\partial x_3} + \frac{\partial v_3}{\partial x_2} \right) \right] d\mathbf{x} \\ &\quad - \oint_{\Gamma^e} \delta v_3 T_3 ds + \int_{\Omega^e} \gamma_e \frac{\partial \delta v_3}{\partial x_3} \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) d\mathbf{x} \end{aligned} \quad (5.4.1c)$$

Energy

$$0 = \int_{\Omega^e} \rho_0 C_v \delta T \left[\frac{\partial T}{\partial t} + \left(v_1 \frac{\partial T}{\partial x_1} + v_2 \frac{\partial T}{\partial x_2} + v_3 \frac{\partial T}{\partial x_3} \right) - Q \right] d\mathbf{x} \\ + \int_{\Omega^e} k_{ij} \frac{\partial \delta T}{\partial x_i} \frac{\partial T}{\partial x_j} d\mathbf{x} - \oint_{\Gamma^e} \delta T q ds \quad (5.4.2)$$

where γ_e is the penalty parameter. We note that the pressure does not appear explicitly in the weak forms (5.4.1) and (5.4.2), although it is a part of the boundary stresses, \mathcal{T}_i [see Eqs. (5.2.9b) and (5.2.11)]. An approximation for the pressure can be post-computed from the relation [see Eq. (4.3.20)]

$$P = -\gamma_e \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3} \right) = -\gamma_e \frac{\partial v_i}{\partial x_i} \quad (5.4.3)$$

5.4.2 Reduced Integration Penalty Model

The penalty finite element model is obtained from Eq. (5.4.1) by substituting finite element interpolation (5.3.2b) for the velocity field and $\delta v_i = \Psi^T$:

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}}_1 \\ \dot{\mathbf{v}}_2 \\ \dot{\mathbf{v}}_3 \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}(\mathbf{v}) \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{Bmatrix} + \\ \begin{bmatrix} 2\mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{21} & \mathbf{K}_{31} \\ \mathbf{K}_{12} & \mathbf{K}_{11} + 2\mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{32} \\ \mathbf{K}_{13} & \mathbf{K}_{23} & \mathbf{K}_{11} + \mathbf{K}_{22} + 2\mathbf{K}_{33} \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{Bmatrix} \\ + \begin{bmatrix} \hat{\mathbf{K}}_{11} & \hat{\mathbf{K}}_{12} & \hat{\mathbf{K}}_{13} \\ \hat{\mathbf{K}}_{21} & \hat{\mathbf{K}}_{22} & \hat{\mathbf{K}}_{23} \\ \hat{\mathbf{K}}_{31} & \hat{\mathbf{K}}_{32} & \hat{\mathbf{K}}_{33} \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \mathbf{F}_3 \end{Bmatrix} \quad (5.4.4)$$

where \mathbf{M} , $\mathbf{C}(\mathbf{v})$, \mathbf{K}_{ij} , and \mathbf{F}_i are the same as those defined in Eq. (5.3.9), and

$$\hat{\mathbf{K}}_{ij} = \int_{\Omega^e} \gamma_e \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \quad (5.4.5)$$

The finite element model of the energy Eq. (5.3.8) or (5.3.12) remains unchanged.

The comments made in Section 4.5.3 concerning the evaluation of the integrals in $\hat{\mathbf{K}}$ of Eq. (5.4.4) should be recalled here. These coefficients (i.e., penalty terms) should be evaluated using a numerical integration rule of an order less than that required to integrate them exactly. For example, if a linear rectangular element is used to approximate the velocity field for two-dimensional problems, all matrix coefficients except the penalty terms should be evaluated using the 2×2 Gauss quadrature, and the penalty terms should be evaluated using the one-point (1×1) Gauss quadrature. When a quadratic rectangular element is used, the 3×3 Gauss quadrature is used to evaluate the non-penalty terms, and the 2×2 Gauss quadrature is used to evaluate the penalty terms. Similar comments apply to three-dimensional elements.

Concerning the post-computation of pressure in the penalty model, in general, the pressure computed from Eq. (5.4.3) at the integration points is not always reliable and accurate, and one is advised to use either the Poisson equation for pressure [see Eq. (4.5.11)] or the scheme suggested in Section 4.5.4 [see Eq. (4.5.10)].

5.4.3 Consistent Penalty Model

In this model, the finite element model of Eq. (5.4.3) is constructed first. We have

$$\left[\int_{\Omega^e} \Phi \Phi^T d\mathbf{x} \right] \mathbf{P} = - \left[\int_{\Omega^e} \gamma_e \Phi \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_i \quad (5.4.6a)$$

or in matrix notation

$$\mathbf{M}_p \mathbf{P} = -\gamma_e \mathbf{Q}^T \mathbf{v} \quad (5.4.6b)$$

Inverting Eq. (5.4.6b) for \mathbf{P} , we obtain

$$\mathbf{P} = -\gamma_e \mathbf{M}_p^{-1} \mathbf{Q}^T \mathbf{v} \quad (5.4.7)$$

where $\mathbf{v}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T\}$, $\mathbf{Q}^T = \{\mathbf{Q}_1^T, \mathbf{Q}_2^T, \mathbf{Q}_3^T\}$, and \mathbf{Q}_i and \mathbf{M}_p are given by

$$\mathbf{Q}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x}; \quad \mathbf{M}_p = \int_{\Omega^e} \Phi \Phi^T d\mathbf{x} \quad (5.4.8)$$

Next, Eq. (5.4.7) is substituted for the pressure into the finite element model (5.3.11) to obtain

$$\mathbf{M} \dot{\mathbf{v}} + (\mathbf{C}(\mathbf{v}) + \mathbf{K} + \mathbf{K}_p) \mathbf{v} = \mathbf{F} \quad (5.4.9a)$$

with

$$\mathbf{K}_p = \gamma_e \mathbf{Q} \mathbf{M}_p^{-1} \mathbf{Q}^T \quad (5.4.9b)$$

It should be noted that reduced integration is not used to evaluate \mathbf{K}_p . For straight-sided elements, one can show the equivalence of matrix $\hat{\mathbf{K}}$ in Eq. (5.4.4) and the matrix \mathbf{K}_p . The post-computation of the pressure P using Eq. (5.4.7) in the consistent penalty method yields very good results.

5.5 Finite Element Models of Porous Flow

The derivation of the finite element model for the nonisothermal, incompressible porous flow problem follows the developments of Section 4.4 in a completely analogous manner. Note that the Boussinesq approximation is utilized in this formulation. The density in the body force term is given by (5.2.1) and is considered a constant at ρ_0 in all other terms. For completeness, we record the mixed finite element model including the buoyancy term:

Continuity

$$-\left[\int_{\Omega^e} \Phi \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_i = \mathbf{0} \quad (5.5.1)$$

Momentum

$$\left[\int_{\Omega^e} \frac{\rho_0}{\phi} \Psi \Psi^T d\mathbf{x} \right] \dot{\mathbf{v}}_i + \left[\int_{\Omega^e} \frac{\rho_0 \hat{c}}{\sqrt{\kappa}} \Psi (\Psi^T \|\mathbf{v}\|) \Psi^T d\mathbf{x} \right] \mathbf{v}_i + \left[\int_{\Omega^e} \frac{\mu}{\kappa} \Psi \Psi^T d\mathbf{x} \right] \dot{\mathbf{v}}_i$$

$$+ \left[\int_{\Omega^e} \mu_e \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_j} dx \right] \mathbf{v}_i + \left[\int_{\Omega^e} \mu_e \frac{\partial \Psi}{\partial x_j} \frac{\partial \Psi^T}{\partial x_i} dx \right] \mathbf{v}_j - \left[\int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T dx \right] \mathbf{P} = \\ - \left[\int_{\Omega^e} \rho_0 g_i \beta \Psi \Theta dx \right] \mathbf{T} + \left\{ \int_{\Omega^e} \rho_0 g_i \beta \Psi T_0 dx \right\} + \left\{ \oint_{\Gamma^e} \Psi \mathcal{T}_i ds \right\} \quad (5.5.2)$$

Energy

$$\left[\int_{\Omega^e} (\rho C)_e \Theta \Theta^T dx \right] \dot{\mathbf{T}} + \left[\int_{\Omega^e} \rho_0 C \Theta (\Psi^T \mathbf{u}_j) \frac{\partial \Theta^T}{\partial x_j} dx \right] \mathbf{T} + \\ \left[\int_{\Omega^e} k_e \frac{\partial \Theta}{\partial x_i} \frac{\partial \Theta^T}{\partial x_i} dx \right] \mathbf{T} = \left\{ \int_{\Omega^e} \Theta Q dx \right\} + \left\{ \oint_{\Gamma^e} \Theta q_n ds \right\} \quad (5.5.3)$$

Note that the dissipation term is omitted from the energy equation, and the acceleration tensor is taken to be $1/\phi$. The above equations can be written in matrix form as given below.

Continuity

$$-\tilde{\mathbf{Q}}^T \mathbf{v} = \mathbf{0} \quad (5.5.4)$$

Momentum

$$\tilde{\mathbf{M}} \dot{\mathbf{v}} + \tilde{\mathbf{C}} \mathbf{v} + \tilde{\mathbf{K}} \mathbf{v} - \tilde{\mathbf{Q}} \mathbf{P} + \tilde{\mathbf{B}} \mathbf{T} = \tilde{\mathbf{F}} \quad (5.5.5)$$

Energy

$$\tilde{\mathbf{N}} \dot{\mathbf{T}} + \tilde{\mathbf{D}} \mathbf{T} + \tilde{\mathbf{L}} \mathbf{T} = \tilde{\mathbf{G}} \quad (5.5.6)$$

where $\mathbf{v}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T\}$ and

$$\begin{aligned} \tilde{\mathbf{M}} &= \int_{\Omega^e} \frac{\rho_0}{\phi} \Psi \Psi^T dx, \quad \tilde{\mathbf{C}}(\mathbf{v}) = \int_{\Omega^e} \frac{\rho_0 \hat{c}}{\sqrt{\kappa}} \Psi \Psi^T \|\mathbf{v}\| \Psi^T dx \\ \tilde{\mathbf{A}} &= \int_{\Omega^e} \frac{\mu}{\kappa} \Psi \Psi^T dx, \quad \tilde{\mathbf{K}}_{ij} = \int_{\Omega^e} \mu_e \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} dx, \quad \tilde{\mathbf{Q}}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T dx \\ \tilde{\mathbf{F}}_i(\mathbf{T}) &= - \int_{\Omega^e} \rho_0 g_i \beta \Psi \Theta dx + \int_{\Omega^e} \rho_0 g_i \beta \Psi T_0 dx + \oint_{\Gamma^e} \Psi \mathcal{T}_i ds \quad (5.5.7) \\ \tilde{\mathbf{D}}(\mathbf{v}) &= \int_{\Omega^e} \rho_0 C \Theta (\Psi^T \mathbf{v}_j) \frac{\partial \Theta^T}{\partial x_j} dx, \quad \tilde{\mathbf{N}} = \int_{\Omega^e} (\rho C)_e \Theta \Theta^T dx \\ \tilde{\mathbf{L}} &= \int_{\Omega^e} k_e \frac{\partial \Theta}{\partial x_i} \frac{\partial \Theta^T}{\partial x_i} dx, \quad \tilde{\mathbf{G}} = \int_{\Omega^e} \Theta Q dx + \oint_{\Gamma^e} \Theta q_n ds \end{aligned}$$

where summation on repeated indices is implied. The porous flow Eqs. (5.5.4)–(5.5.6) can be arranged into a single matrix equation of the same form as given previously for the flow equations:

$$\begin{bmatrix} \tilde{\mathbf{M}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \tilde{\mathbf{N}} \\ \mathbf{0} & \mathbf{0} & \tilde{\mathbf{N}} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}} \\ \dot{\mathbf{P}} \\ \dot{\mathbf{T}} \end{Bmatrix} + \begin{bmatrix} \tilde{\mathbf{C}}(\mathbf{v}) + \tilde{\mathbf{A}} + \tilde{\mathbf{K}} & -\tilde{\mathbf{Q}} & \tilde{\mathbf{B}}(\mathbf{T}) \\ -\tilde{\mathbf{Q}}^T & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \tilde{\mathbf{D}}(\mathbf{v}) + \tilde{\mathbf{L}}(\mathbf{T}) \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \\ \mathbf{T} \end{Bmatrix} = \begin{Bmatrix} \tilde{\mathbf{F}}(\mathbf{T}) \\ \mathbf{0} \\ \tilde{\mathbf{G}}(\mathbf{T}, \mathbf{v}) \end{Bmatrix} \quad (5.5.8)$$

5.6 Nonisothermal, Low-Speed, Compressible Flows

5.6.1 Governing Equations

The equations describing nonisothermal, low-speed compressible flow of a Newtonian fluid were outlined in Chapter 1 and are summarized here. The equations are written in advective form using the standard acoustically filtered approximation. The equation of state and conservation equations are presented below.

Equation of State

$$\rho = \rho(T, P_0) \quad (5.6.1)$$

Conservation of Mass

$$\frac{\partial \rho v_i}{\partial x_i} = -\frac{\partial \rho}{\partial t} \quad (5.6.2)$$

Conservation of Momentum

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) - \frac{\partial}{\partial x_j} \left[-P' \delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \frac{2}{3} \frac{\partial}{\partial x_i} \left[\mu \left(\frac{\partial v_k}{\partial x_k} \right) \right] - \rho g_i = 0 \quad (5.6.3)$$

Conservation of Energy

$$\rho C_v \left(\frac{\partial T}{\partial t} + v_i \frac{\partial T}{\partial x_i} \right) - \frac{\partial}{\partial x_i} \left(k \frac{\partial T}{\partial x_i} \right) - Q - \Phi - \beta T \frac{dP_0}{dt} = 0 \quad (5.6.4)$$

The variables and parameters used in (5.6.1)–(5.6.4) have the standard meaning; the viscous dissipation definition is altered because the stress tensor is expanded to include the velocity divergence term. In the above system of equations, the total pressure P is composed of two parts

$$P(x_i, t) = P_0(t) + P'(x_i, t) \quad (5.6.5)$$

where P_0 is a spatially uniform, time-dependent background (thermodynamic) pressure and P' is a dynamical (mechanical) pressure. The absence of the dynamical pressure from the equation of state precludes acoustic waves and gives the fluid model its name. Formal derivation and theoretical discussion of this system is available in [1-3].

The system in (5.6.2)–(5.6.5) is observed to be quite similar to the Boussinesq system (5.2.2)–(5.2.5) if the density is considered a variable parameter that can be evaluated as needed using (5.6.1). The additional “source” terms in (5.6.2) and (5.6.4) pose little difficulty in evaluation and the inclusion of the added stress term is straightforward. This similarity with the Boussinesq system provides guidance for the development of a finite element method for low-speed compressible flows as outlined in a following section.

For completeness, the conservation form of the non-Boussinesq equations will also be summarized. The conservation form is significantly more complex than the advective form, with a number of additional terms that complicate both the finite element model and associated solution procedures. Though the differences between numerical models for the advection and conservation forms of the Boussinesq equations have been judged to be generally insignificant, the same conclusion is not available

for the non-Boussinesq (acoustically filtered) equations; comparison of the models requires additional evaluation. The generalized advection operator, used previously with the Boussinesq equations, could also be employed here to arrive at the proper conservation form. However, an alternate approach will be used to demonstrate a limiting form of the fully compressible equations. In this approach, the all-Mach number formulation, as defined by Hauke and Hughes [8,9], is the starting point. As noted in [8,9], the fully compressible equations will not have a well defined incompressible limit when density is used as a variable. The most convenient variables are therefore the pressure, primitive variables (P, v_i, T) . Using these variables, the conservation form of the fully compressible flow equations includes an equation of state. The equation of state and conservation equations are given below.

Equation of State

$$\rho = \rho(T, P) \quad (5.6.6)$$

Conservation of Mass

$$\rho\kappa\frac{DP}{Dt} - \rho\beta\frac{DT}{Dt} + \rho\frac{\partial v_i}{\partial x_i} = 0 \quad (5.6.7)$$

Conservation of Momentum

$$\rho\frac{Dv_i}{Dt} + \rho\kappa v_i\frac{DP}{Dt} - \rho\beta v_i\frac{DT}{Dt} + \rho v_i\frac{\partial v_k}{\partial x_k} + \delta_{ij}\frac{\partial P}{\partial x_j} - \frac{\partial\sigma_{ij}}{\partial x_j} - \rho g_i = 0 \quad (5.6.8)$$

Conservation of Energy

$$\rho\frac{De^t}{Dt} + \rho\kappa e^t\frac{DP}{Dt} - \rho\beta e^t\frac{DT}{Dt} + \rho e^t\frac{\partial v_k}{\partial x_k} + \frac{\partial Pv_i}{\partial x_i} - \frac{\partial\sigma_{ij}v_j}{\partial x_i} + \frac{\partial q_i}{\partial x_i} - \rho g_i v_i - Q = 0 \quad (5.6.9)$$

where the additional parameters and variables are the total energy $e^t = e + 1/2v_k v_k$, defined as the sum of the internal and kinetic energies, the isothermal compressibility coefficient, κ , and the coefficient of thermal expansion (at constant pressure) β , given by

$$\kappa = \frac{1}{\rho}\frac{\partial\rho}{\partial P}\Big|_T, \quad \beta = -\frac{1}{\rho}\frac{\partial\rho}{\partial T}\Big|_P \quad (5.6.10)$$

The viscous stress tensor σ_{ij} again has the compressible form which includes the divergence of the velocity. Note also that the substantial derivative of the density is

$$\frac{D\rho}{Dt} = \rho\kappa\frac{DP}{Dt} - \rho\beta\frac{DT}{Dt} \quad (5.6.11)$$

A standard scaling of Eqs. (5.6.7)–(5.6.9) allows the low Mach number form of the equations to be derived. In this limit, the kinetic energy terms and dissipation terms in the energy equation are neglected (small Eckert number). The resulting system of equations is valid for low speed flows without restrictions on the equation of state. If a perfect gas is assumed for convenience, the resulting equations are (5.6.7), (5.6.8), and a simplified energy equation of the form

$$\rho\frac{DC_vT}{Dt} + \rho\kappa C_v T\frac{DP}{Dt} - \rho\beta C_v T\frac{DT}{Dt} + \rho C_v T\frac{\partial v_k}{\partial x_k} + \frac{\partial q_i}{\partial x_i} - Q = 0 \quad (5.6.12)$$

where the relations $P = \rho RT$ and $e = C_v T$ have been defined. This system is not acoustically filtered because the density is still a function of the total (thermodynamic) pressure. However, the system is a valid form for nonisothermal, low-speed flows and could possibly be used for developing finite element models for this class of heat transfer problem. The problem of acoustic waves may cause problems in this system for certain flows and the further reduction to an acoustically filtered form is desirable.

When the isothermal compressibility parameter κ is set to zero, the density relation in Eq. (5.6.11) is simplified and the conservation equations reduce to a conservation form analogous to the non-Boussinesq equations in (5.6.2)–(5.6.4). This system can be expressed as follows.

Conservation of Mass

$$\rho \frac{\partial v_i}{\partial x_i} - \rho \beta \frac{DT}{Dt} = 0 \quad (5.6.13)$$

Conservation of Momentum

$$\rho \frac{Dv_i}{Dt} + v_i \left(\rho \frac{\partial v_k}{\partial x_k} - \rho \beta \frac{DT}{Dt} \right) + \delta_{ij} \frac{\partial P}{\partial x_j} - \frac{\partial \sigma_{ij}}{\partial x_j} + \rho g_i = 0 \quad (5.6.14)$$

Conservation of Energy

$$\rho C_v \frac{DT}{Dt} + C_v T \left(\rho \frac{\partial v_k}{\partial x_k} - \rho \beta \frac{DT}{Dt} \right) + \frac{\partial q_i}{\partial x_i} - Q = 0 \quad (5.6.15)$$

If the mass conservation equation is used to simplify (5.6.14) and (5.6.15), the resulting system is the advective form of the acoustically filtered equations. Though not shown here, it is straightforward to incorporate a spatially uniform, background pressure in the development. Also, if the thermal expansion coefficient is set to zero, the conservation form of the Boussinesq equations are produced though some assumption regarding the body force term is still required.

Both the advective and conservation forms of the acoustically filtered, non-Boussinesq equations may be used for developing finite element models for heat transfer applications. A comparison of the Boussinesq and non-Boussinesq formulations is illustrated in the examples of Section 5.13.

5.6.2 Boundary Conditions

The boundary conditions for the (acoustically filtered) non-Boussinesq equations are formally the same as given previously in Section 5.2.2 for the Boussinesq system. Typically, the velocity and/or tractions are specified on various parts of the fluid mechanics boundary. Also, the temperature and/or heat flux is specified for the thermal boundary of the problem.

5.6.3 Mixed Finite Element Model

The derivation of a finite element model for the acoustically filtered equations follows closely the development of the Boussinesq system. The advective form of the non-Boussinesq system will be considered in detail; the derivation for the conservation form need only be outlined as the models differ by only a few terms. Because

the dynamical part of the pressure P' does not appear in the equation of state, it plays the same role as the pressure in an incompressible flow. That is, the dynamical pressure is a Lagrange multiplier that enforces satisfaction of the continuity constraint. Note that in the time dependent case, the continuity equation has a source term that is not present in the Boussinesq system and will require evaluation based on the equation of state. Overall, the density is generally treated as a variable parameter in the problem that is evaluated from the temperature field through the equation of state.

Spatial discretization of the acoustically filtered Eqs. (5.6.2)–(5.6.4) may be accomplished using the Galerkin form of the method of weighted residuals. The specific approach considered here was designed by Martinez [10] to be a simple extension of the Boussinesq equations. The formulation differs only slightly from other published procedures (see, for example, [11,12]). The weighted integral statements for the conservation of mass, momentum and energy are the same as for the Boussinesq equations and are given in (5.3.1). Let the dependent variables be approximated by expansions of the form

$$T(\mathbf{x}, t) = \sum_{m=1}^M \theta_m(\mathbf{x}) T_m(t) = \boldsymbol{\Theta}^T \mathbf{T} \quad (5.6.16a)$$

$$v_i(\mathbf{x}, t) = \sum_{n=1}^N \psi_n(\mathbf{x}) v_i^n(t) = \boldsymbol{\Psi}^T \mathbf{v}_i \quad (5.6.16b)$$

$$P(\mathbf{x}, t) = \sum_{l=1}^L \phi_l(\mathbf{x}) P_l(t) = \boldsymbol{\Phi}^T \mathbf{P} \quad (5.6.16c)$$

$$\rho(\mathbf{x}, t) = \sum_{m=1}^M \theta_m(\mathbf{x}) \rho_m(t) = \boldsymbol{\Theta}^T \rho \quad (5.6.16d)$$

where the temperature and density variables are assumed to have the same interpolation functions. In general, the interpolation functions for the velocity will also be the same as the temperature functions with the pressure being one order lower than the velocity.

Substitution of (5.6.16) into the weak form associated with (5.3.1) and identification of the appropriate weighting functions as in (5.3.3), produces the following matrix equations.

Continuity

$$\mathbf{H}\dot{\rho} + \mathbf{Q}^T \mathbf{R}\mathbf{v} = \mathbf{0} \quad (5.6.17)$$

Momentum

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{C}(\mathbf{R}\mathbf{v})\mathbf{v} + \tilde{\mathbf{K}}\mathbf{v} - \mathbf{Q}\mathbf{P} + \mathbf{B} = \mathbf{F} \quad (5.6.18)$$

Energy

$$\mathbf{N}\dot{\mathbf{T}} + \mathbf{D}(\mathbf{R}\mathbf{v})\mathbf{T} + \mathbf{L}\mathbf{T} = \mathbf{G} \quad (5.6.19)$$

State

$$\rho = \rho(\mathbf{T}, P_0) \quad (5.6.20)$$

where the superposed dot represents a time derivative and $\mathbf{v}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T\}$. This system is analogous to the Boussinesq equations displayed in (5.3.10)–(5.3.12). The individual matrix terms are defined by

$$\begin{aligned}
 \mathbf{H} &= \int_{\Omega^e} \Phi \Theta^T d\mathbf{x} \\
 \mathbf{M} &= \int_{\Omega^e} \rho \Psi \Psi^T d\mathbf{x}; \quad \mathbf{C}(\mathbf{R}\mathbf{v}) = \int_{\Omega^e} \Psi (\Psi^T \mathbf{R} \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \\
 \mathbf{K}_{ij} &= \int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x}; \quad \mathbf{Q}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x} \\
 \mathbf{B}_i &= \int_{\Omega^e} \Psi \Theta^T \rho g_i d\mathbf{x}; \quad \mathbf{F}_i = \oint_{\Gamma^e} \Psi \mathcal{T}_i ds \quad (5.6.21a) \\
 \mathbf{D}(\mathbf{R}\mathbf{v}) &= \int_{\Omega^e} C \Theta (\Psi^T \mathbf{R} \mathbf{v}_j) \frac{\partial \Theta^T}{\partial x_j} d\mathbf{x} \\
 \mathbf{N} &= \int_{\Omega^e} \rho C_v \Theta \Theta^T d\mathbf{x}; \quad \mathbf{L} = \int_{\Omega^e} k \frac{\partial \Theta}{\partial x_i} \frac{\partial \Theta^T}{\partial x_i} d\mathbf{x} \\
 \mathbf{G} &= \int_{\Omega^e} \Theta Q d\mathbf{x} + \oint_{\Gamma^e} \Theta q ds \\
 \tilde{\mathbf{K}}_{11} &= \frac{4}{3} \mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} \\
 \tilde{\mathbf{K}}_{12} &= \mathbf{K}_{21} - \frac{2}{3} \mathbf{K}_{12} \\
 \tilde{\mathbf{K}}_{13} &= \mathbf{K}_{31} - \frac{2}{3} \mathbf{K}_{13} \\
 \tilde{\mathbf{K}}_{21} &= \mathbf{K}_{12} - \frac{2}{3} \mathbf{K}_{21} \\
 \tilde{\mathbf{K}}_{22} &= \mathbf{K}_{11} + \frac{4}{3} \mathbf{K}_{22} + \mathbf{K}_{33} \quad (5.6.21b) \\
 \tilde{\mathbf{K}}_{23} &= \mathbf{K}_{32} - \frac{2}{3} \mathbf{K}_{23} \\
 \tilde{\mathbf{K}}_{13} &= \mathbf{K}_{31} - \frac{2}{3} \mathbf{K}_{13} \\
 \tilde{\mathbf{K}}_{23} &= \mathbf{K}_{32} - \frac{2}{3} \mathbf{K}_{23} \\
 \tilde{\mathbf{K}}_{33} &= \mathbf{K}_{11} + \mathbf{K}_{22} + \frac{4}{3} \mathbf{K}_{33}
 \end{aligned}$$

Equations (5.6.17)–(5.6.20) represent coupled, nonlinear ordinary differential equations for the nodal point unknowns of velocity \mathbf{v}_i , mechanical pressure \mathbf{P} , temperature \mathbf{T} and density ρ . In this formulation the matrix $\mathbf{R} = \text{dia}[\rho]$ is used to include the variable density in the convective terms. In essence, the convective terms have the term ρv_i interpolated as a product rather than a product of interpolants. This manipulation is not mandatory and has not been used in other implementations. Compared to the standard Boussinesq formulation, only the \mathbf{H} matrix is new, though the diffusion matrix $\tilde{\mathbf{K}}$ is altered by the compressible form of the stress tensor and the body force term \mathbf{B} is not explicit in the temperature.

As noted in previous sections, the separate Eqs. (5.6.17)–(5.6.19) can be combined into a single matrix equation as

$$\begin{aligned} \begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{N} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}} \\ \dot{\mathbf{P}} \\ \dot{\mathbf{T}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{R}\mathbf{v}) + \tilde{\mathbf{K}}(\mathbf{T}) & -\mathbf{Q} & \mathbf{B} \\ -\mathbf{Q}^T & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}(\mathbf{R}\mathbf{v}) + \mathbf{L}(\mathbf{T}) \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \\ \mathbf{T} \end{Bmatrix} \\ = \begin{Bmatrix} \mathbf{F} \\ \mathbf{H}\dot{\rho} \\ \mathbf{G}(\mathbf{T}, \mathbf{v}) \end{Bmatrix} \end{aligned} \quad (5.6.22)$$

In a more symbolic format, Eq. (5.6.22) can be expressed as

$$\bar{\mathbf{M}}\dot{\mathbf{U}} + \bar{\mathbf{K}}\mathbf{U} = \bar{\mathbf{F}} \quad (5.6.23a)$$

where again

$$\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{P}^T, \mathbf{T}^T\} \quad (5.6.23b)$$

In the non-Boussinesq system, the density is treated as a temperature dependent property based on (5.6.20); the background pressure is computed as needed from an initial condition or an energy balance on a closed system. The density rate term in (5.6.17) may be evaluated by either of two methods. In the simplest case, the rate term $\mathbf{H}\dot{\rho}$ is moved to the right-hand side of the equation (as shown above) and time-differenced appropriately. Previous values of the density and density rate may be stored or recomputed from the equation of state as needed. In a more implicit method, the density rate is replaced by a temperature rate using

$$\dot{\rho} = \left(\frac{\partial \rho}{\partial T} \right) \dot{T} \quad (5.6.24)$$

An appropriate recursion relation is employed for \dot{T} and the term remains on the left-hand side of the equation. Both methods have been used in practice with the more implicit scheme being somewhat more robust. General solution methods for the non-Boussinesq equations are similar to the Boussinesq system and are outlined in the following section. Further discussion of the non-Boussinesq formulation and application can be found in [4,11,12] and in the examples in a later section.

5.7 Solution Methods

5.7.1 General Discussion

The finite element models developed in the previous sections describe a very large group of heat transfer and fluid mechanics problems ranging in complexity from simple solid body conduction through forced and free convection, to mixed mode convection/conduction. The solution methods used depend on the model, computational resources, the nonlinearity of the system, and the strength of the coupling between equations (see Section 4.6).

The solution procedures described in Section 4.6 were all variants of the basic fixed point method (see Appendix C) and were generally designed to operate concurrently on all of the equations present in a finite element model. The strong coupling

between equations (e.g., momentum and energy) that is characteristic of convective heat transfer problems makes these combined equation methods optimal from the standpoint of convergence rate. The disadvantage, of course, is that a very large and computationally expensive matrix problem must be treated at each iteration. The requirement to perform larger (more elements and higher dimensionality) and more complex (physical phenomena) simulations has reached the point where the usual direct matrix methods for combined equations are prohibitively expensive. A natural choice to make the matrix problem more affordable (while retaining the standard fixed point schemes) is to switch from the Gauss elimination type methods to the iterative methods, such as the preconditioned conjugate gradient (PCG) method. Previously, the development of iterative methods for combined equation sets was slow and has been handicapped by the lack of good preconditioner techniques that can adequately treat the dominating effect of the incompressibility constraint. Various methods now exist that can solve an increasing range of combined equation problems. Though still not as robust or generally applicable as required by the spectrum of applications, there is an increasing use of iterative methods. However, for some demanding applications, the combined equation approach may be sacrificed for alternative formulations of the discrete equations.

In the following sections we first review Newton's method for convective heat transfer problems, a combined equation technique that is heavily used for two-dimensional problems and modest three-dimensional applications. Various segregated solution methods are then reviewed. Throughout this section we will only consider the time-independent equations. From the discussion in Section 4.6 it should be recognized that any of the standard time integration procedures can be applied to the convective heat transfer equations. The resulting nonlinear algebraic equations for each time step are then analogous to the steady state problem, and they can be treated with the algorithms described in the following sections.

5.7.2 Newton's Method

The application of Newton's method to the combined equations for convective heat transfer (5.3.14), (5.5.8) or (5.6.22) follows directly from the isothermal case described in Section 4.6.2. The iterative procedure is given by

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \mathcal{J}^{-1}(\mathbf{U}^n)\mathbf{R}(\mathbf{U}^n) \quad (5.7.1)$$

where the vector of unknowns is $\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{P}^T, \mathbf{T}^T\}$ for the mixed finite element model and $\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{T}^T\}$ for the penalty finite element model. The definition of the Jacobian matrix \mathcal{J} is expanded to include the variations in the equations with respect to the vector of nodal temperatures, \mathbf{T} . Thus for the two-dimensional mixed finite element model of a Boussinesq system we have

$$\mathcal{J} = \frac{\partial \mathbf{R}}{\partial \mathbf{U}} = \begin{bmatrix} \frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_1}{\partial \mathbf{P}} & \frac{\partial \mathbf{R}_1}{\partial \mathbf{T}} \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{P}} & \frac{\partial \mathbf{R}_2}{\partial \mathbf{T}} \\ \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{P}} & \frac{\partial \mathbf{R}_3}{\partial \mathbf{T}} \\ \frac{\partial \mathbf{R}_4}{\partial \mathbf{v}_1} & \frac{\partial \mathbf{R}_4}{\partial \mathbf{v}_2} & \frac{\partial \mathbf{R}_4}{\partial \mathbf{P}} & \frac{\partial \mathbf{R}_4}{\partial \mathbf{T}} \end{bmatrix} \quad (5.7.2)$$

where

$$\begin{aligned}\mathbf{R}_1 &= \mathbf{C}_1(\mathbf{v}_1)\mathbf{v}_1 + \mathbf{C}_2(\mathbf{v}_2)\mathbf{v}_1 + (2\mathbf{K}_{11} + \mathbf{K}_{22})\mathbf{v}_1 + \mathbf{K}_{12}\mathbf{v}_2 - \mathbf{Q}_1\mathbf{P} + \mathbf{B}_1\mathbf{T} - \mathbf{F}_1 \\ \mathbf{R}_2 &= \mathbf{C}_1(\mathbf{v}_1)\mathbf{v}_2 + \mathbf{C}_2(\mathbf{v}_2)\mathbf{v}_2 + \mathbf{K}_{21}\mathbf{v}_1 + (\mathbf{K}_{11} + 2\mathbf{K}_{22})\mathbf{v}_2 - \mathbf{Q}_2\mathbf{P} + \mathbf{B}_2\mathbf{T} - \mathbf{F}_2 \\ \mathbf{R}_3 &= -\mathbf{Q}_1^T\mathbf{v}_1 - \mathbf{Q}_2^T\mathbf{v}_2 \\ \mathbf{R}_4 &= \mathbf{D}_1(\mathbf{v}_1)\mathbf{T} + \mathbf{D}_2(\mathbf{v}_2)\mathbf{T} + (\mathbf{L}_{11} + \mathbf{L}_{22})\mathbf{T} - \mathbf{G}\end{aligned}\quad (5.7.3)$$

which correspond to the two momentum equations, the incompressibility constraint, and the energy equation. The matrices appearing in Eq. (5.7.3) are defined in Eqs. (5.3.9a,b). The components of \mathcal{J} are computed directly from the definitions in (5.7.3) and yield the following values:

$$\begin{aligned}\frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_1} &= \mathbf{C}_1(\mathbf{v}_1) + \mathbf{C}_2(\mathbf{v}_2) + \mathbf{C}_1(1)\mathbf{v}_1 + 2\mathbf{K}_{11} + \mathbf{K}_{22} \\ \frac{\partial \mathbf{R}_1}{\partial \mathbf{v}_2} &= \mathbf{C}_2(1)\mathbf{v}_1 + \mathbf{K}_{21}; \quad \frac{\partial \mathbf{R}_1}{\partial \mathbf{P}} = -\mathbf{Q}_1; \quad \frac{\partial \mathbf{R}_1}{\partial \mathbf{T}} = \mathbf{B}_1 \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_1} &= \mathbf{C}_1(1)\mathbf{v}_2 + \mathbf{K}_{12}; \quad \frac{\partial \mathbf{R}_2}{\partial \mathbf{P}} = -\mathbf{Q}_2 \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{v}_2} &= \mathbf{C}_1(\mathbf{v}_1) + \mathbf{C}_2(\mathbf{v}_2) + \mathbf{C}_2(1)\mathbf{v}_2 + \mathbf{K}_{11} + 2\mathbf{K}_{22} \\ \frac{\partial \mathbf{R}_2}{\partial \mathbf{T}} &= \mathbf{B}_2; \quad \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_1} = -\mathbf{Q}_1^T; \quad \frac{\partial \mathbf{R}_3}{\partial \mathbf{v}_2} = -\mathbf{Q}_2^T; \quad \frac{\partial \mathbf{R}_3}{\partial \mathbf{P}} = \mathbf{0} \\ \frac{\partial \mathbf{R}_3}{\partial \mathbf{T}} &= \mathbf{0}; \quad \frac{\partial \mathbf{R}_4}{\partial \mathbf{v}_1} = \mathbf{D}_1(1)\mathbf{T}; \quad \frac{\partial \mathbf{R}_4}{\partial \mathbf{v}_2} = \mathbf{D}_2(1)\mathbf{T}; \quad \frac{\partial \mathbf{R}_4}{\partial \mathbf{P}} = \mathbf{0} \\ \frac{\partial \mathbf{R}_4}{\partial \mathbf{T}} &= \mathbf{D}_1(\mathbf{v}_1) + \mathbf{D}_2(\mathbf{v}_2) + \mathbf{L}_{11} + \mathbf{L}_{22}\end{aligned}\quad (5.7.4)$$

The three-dimensional formulation follows directly from the above definitions.

The advantages and disadvantages of Newton's method as outlined in Section 4.6.2 for the isothermal case hold equally for the convective heat transfer system. Also, the other variations of the fixed point schemes, such as Picard iteration, modified and quasi-Newton methods, and the various continuation techniques can be directly extended to the nonisothermal situation with comparable effectiveness.

5.7.3 Segregated Equation Methods

Segregated equation methods are not specifically solution algorithms but rather an approach to decomposing Eq. (5.3.14) (or similar systems) into a series of smaller problems that can be solved efficiently using direct matrix methods or are more amenable to PCG type methods. For convective heat transfer problems (or other transport problems) a particularly simple segregated or cyclic procedure is given by the following for the $(n+1)$ iteration step:

Step 1 (Energy Equation)

$$\mathbf{D}(\mathbf{v}^n, \mathbf{T}^n)\mathbf{T}^{n+1} + \mathbf{L}(\mathbf{T}^n)\mathbf{T}^{n+1} = \mathbf{G}(\mathbf{v}^n, \mathbf{T}^n) \quad (5.7.5)$$

Step 2 (Continuity and Momentum Equations)

$$-\mathbf{Q}^T\mathbf{v}^{n+1} = \mathbf{0} \quad (5.7.6)$$

$$\mathbf{C}(\mathbf{v}^n)\mathbf{v}^{n+1} + \mathbf{K}(\mathbf{T}^{n+1})\mathbf{v}^{n+1} - \mathbf{Q}\mathbf{P}^{n+1} = \mathbf{F}(\mathbf{T}^{n+1}) \quad (5.7.7)$$

The energy equation is typically solved first in each cycle, especially when considering free convection-dominated flows since the buoyancy term drives the fluid motion; the buoyancy term is included in \mathbf{F} in Eq. (5.7.7).

The procedure given in Eqs. (5.7.5)–(5.7.7), termed a cyclic Picard method, has some advantages over a combined equation method in terms of computer storage and execution time per iteration. Unfortunately, as with most variants of successive substitution, the procedure has a relatively slow rate of convergence; the convergence rate is further reduced due to the lagging of the coefficients induced by the splitting of the equations. This method has been used successfully for convection problems of moderate nonlinearity by several investigators [13–16].

Though the cyclic procedure described above offers some reduction in computational requirements, the combined momentum and continuity equations, especially in three dimensions, can still represent a very large matrix problem. Following the idea of segregating equations to its logical extreme [17], the system in (5.3.14) can be decomposed into the following series of equations:

$$(\mathbf{C}_1(\mathbf{v}) + \mathbf{K}_1)\mathbf{v}_1 - \mathbf{Q}_1\mathbf{P} = \mathbf{F}_1 \quad (5.7.8a)$$

$$(\mathbf{C}_2(\mathbf{v}) + \mathbf{K}_2)\mathbf{v}_2 - \mathbf{Q}_2\mathbf{P} = \mathbf{F}_2 \quad (5.7.8b)$$

$$\mathbf{Q}_1^T\mathbf{v}_1 + \mathbf{Q}_2^T\mathbf{v}_2 = 0 \quad (5.7.8c)$$

$$(\mathbf{D}(\mathbf{v}) + \mathbf{L})\mathbf{T} = \mathbf{G} \quad (5.7.8d)$$

for the case of two-dimensional, nonisothermal flow; the three-dimensional case follows the same procedure as does the inclusion of additional transport equations. The (\mathbf{C}, \mathbf{D}) , and (\mathbf{K}, \mathbf{L}) matrices, represent the advection and diffusion operators, respectively. A segregated solution approach to Eq. (5.7.8) would solve each individual momentum equation, the continuity equation, and the energy equation for the appropriate variable in a cyclic procedure similar to the one shown in Eqs. (5.7.5)–(5.7.7). As each variable is updated from a new solution, it is used where appropriate in the coefficients of the remaining equations, resulting in another type of cyclic Picard scheme. By reducing (5.3.14) to the series of equations in (5.7.8), a significant reduction in size and cost of the matrix problem is realized. The penalty to be paid is the slower convergence rate of the cyclic procedure. However, for very large problems there is a cross-over point for these competing effects and the segregated approach is clearly superior to standard combined equation methods.

A significant complication in the use of Eq. (5.7.8) arises due to the fact that there is no equation in the set from which the pressure may be directly obtained. The solution to this dilemma involves the construction of a pressure Poisson equation. Solving Eqs. (5.7.8a,b) symbolically for the velocities \mathbf{v}_1 and \mathbf{v}_2 and substituting this result into the incompressibility condition (5.7.8c) produces a consistent Poisson equation for the pressure

$$[\mathbf{Q}_1^T(\mathbf{C}_1(\mathbf{v}) + \mathbf{K}_1)^{-1}\mathbf{Q}_1 + \mathbf{Q}_2^T(\mathbf{C}_2(\mathbf{v}) + \mathbf{K}_2)^{-1}\mathbf{Q}_2]\mathbf{P} = \mathbf{F} \quad (5.7.9)$$

where the explicit form of \mathbf{F} is not important in this discussion. Equation (5.7.9), when used in place of the continuity equation, provides a complete set of equations

for all of the unknowns. A number of possible variations exist for the sequential solution of Eqs. (5.7.8) and (5.7.9), each of which produces a slightly different algorithm. In general, the Poisson equation given by (5.7.9) provides an initial estimate of the pressure field which is used in the pressure gradients of the momentum equations; velocity estimates are then obtained from each momentum equation in turn. Since the velocity field is not divergence free at this point, a second application of the Poisson equation is required; this solution supplies a pressure correction that allows the velocity fields to be adjusted to satisfy incompressibility. This general algorithm is analogous to the family of SIMPLE finite difference methods developed by Patankar [18]. Details regarding several variants of the method can be found in the work of Haroutunian et al. [17], who were the original developers of this technique for finite element applications. Note that another significant benefit in the usage of the segregated approach with the pressure Poisson equation is the applicability of iterative matrix solution methods for the individual equations in the sequence. The removal of the incompressibility constraint allows useful preconditioners to be found for each equation in the system. The combination of preconditioned, conjugate gradient (PCG) matrix methods with a segregated equation approach provides the most effective solution method presently available for large scale applications.

5.8 Convection with Change of Phase

There are two major difficulties in simulating melting and solidification change of phase for convection heat transfer problems: (1) accounting for the latent heat effect and (2) the application of fluid boundary conditions on the phase boundary. The latent heat problem has been detailed in the heat conduction section and its solution via the enthalpy method or any other method carries over directly to the convection problem. The boundary condition difficulty, however, is unique to the convection problem though it has some similarities with the free surface problems of Chapter 4.

As part of the formulation of the phase change problem in Section 1.11, it was noted that on the phase boundary the fluid velocity was assumed to be zero [see Eq. (1.11.1)]. This is a particularly difficult condition to apply in standard finite element formulations for two reasons. The location of the phase boundary is not known a priori, and even if it were known, the boundary does not generally coincide with a set of element edges (nodes) to which an essential boundary condition can be applied. One method to circumvent these difficulties is to explicitly track the motion of the phase boundary and allow the finite element mesh to adjust to this motion. Such mesh movement or remeshing schemes allow the phase boundary to always remain aligned with element edges and thus simplify the application of the no-slip boundary condition. However, these procedures are not without their shortcomings. They are expensive, complex to implement for general applications, and are generally limited in application to simple phase boundary motions and shapes. The moving mesh methods described in Section 4.11.3 are candidates for application in this type of phase boundary simulation.

A second, more approximate, method for imposing the no-slip boundary condition relies on the use of a temperature-dependent fluid viscosity and the allowed variation of material properties within each finite element. Depending on the type

of variable coefficient implementation that is used in the construction of element matrices, a variable viscosity is evaluated at the numerical quadrature points or nodal points within an element (see Section 3.9 for details). During each cycle of the solution process the temperature at the viscosity evaluation point is compared with the solidus (liquidus) temperature of the material. If the temperature is above the liquidus, the appropriate fluid viscosity is assigned at the evaluation point; for temperatures below the solidus temperature, the evaluation point viscosity is assigned a “large” value to “solidify” the fluid. Typically, the “solid viscosity” is set at four to five orders of magnitude larger than the normal fluid viscosity; the viscosity function is usually designed to take on a realistic, or at least convenient, variation between the liquidus and solidus temperatures (i.e., through the “mushy” zone). Thermal properties, such as the conductivity, can also be allowed to vary in some physically realistic manner across the phase change zone.

The viscosity evaluation method proposed here allows the phase boundary computation to be maintained on a subelement basis since a given element may contain both fluid and solid phases. However, this procedure is also similar to a smearing technique since a distinct phase boundary is never identified. This is in keeping with the spirit and implementation of the latent heat procedures described previously. This approach is also somewhat like the front tracking, free surface methods of Section 4.11.3. In the present case the front tracking variable is a physical variable, the temperature, rather than an artificial concentration variable [see Eq. (4.11.15)] or a volume fraction [see Eq. (4.11.17)]. The utility of the proposed approach is demonstrated in a subsequent section and in [19].

For materials that change phase over a temperature range, the simple viscosity variation method provides a first-order evaluation of the fluid mechanics and heat transfer processes through the phase change region. In some cases, a more accurate evaluation of the fluid and thermal fields can be achieved. For metal alloys, in which an interdendritic region forms between the liquidus and solidus temperatures, the so-called “mushy” zone is often characterized as a porous media. Careful study of the Darcy-Brinkman porous flow model in (1.6.2) shows that the Navier-Stokes equations can be made a subset of this model if an advective term is added to the porous flow equations. This has been theoretically justified by a number of investigators [20–22] and leads to a momentum equation of the form

$$\frac{\rho_0}{\phi} \frac{\partial v_j}{\partial t} + \frac{\rho_0}{\phi^2} v_j \frac{\partial v_i}{\partial x_j} + \left(\frac{\mu}{\kappa} \right) v_i = \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu_e \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 f_i - \rho_0 g_i \beta (T - T_0) \quad (5.8.1)$$

where the Forchheimer term has been replaced by the advective term and the Brinkman viscosity has been equivalenced to the fluid viscosity. Recall that ϕ is the porosity or liquid volume fraction. This equation can be used throughout the fluid domain and within the mushy zone by proper adjustment of the material properties and coefficients as a function of temperature. In the liquid region ($T > T_{liq}$), the porosity is unity and the permeability κ is infinite; Eq. (5.8.1) is the standard Navier-Stokes equation. When the temperature drops below the solidus temperature ($T < T_{sol}$), the porosity and permeability are zero and the fluid viscosity is again very large, resulting in a “solidified” material. Between the liquidus and

solidus temperatures, the porosity goes from unity to zero as a function of temperature according to the particular model being employed [22]. A model for the permeability as a function of porosity transitions the permeability between a very large value and zero. The consequence of all these transitions is that the flow and heat transfer problem are smoothly moved from a viscous flow description to a porous media flow description to a solidified material description as a function of temperature. The model is substantially more detailed than the simple viscosity variation and is grounded in theory. Further details of the this approach and a demonstration of its capabilities are available in [20,22–24].

5.9 Convection with Enclosure Radiation

The inclusion of enclosure radiation effects in convective heat transfer problems involves the direct application of the methods from Section 3.8. By assumption, the fluid in the enclosure is nonparticipating and the radiative exchange therefore only influences the flux and temperature on the fluid boundaries. For purposes of computation the radiation problem is decoupled from the convection/conduction part of the problem and simply provides flux boundary conditions at each iteration or time step of the simulation.

5.10 Post-Computation of Heat Flux

The results of interest from a convective heat transfer analysis generally include fluid velocities, flow patterns, temperature distributions, and heat fluxes. Most of these items are directly available from the finite element results in terms of nodal point quantities. The heat flux (or flux of an auxiliary variable), however, being a derivative of the temperature, requires an auxiliary computation. Details of a method for performing this operation were given in Section 3.10. The simplest procedure for computing the components of the heat flux vector is based on the element level shape function for the temperature. That is, since

$$T(x_i, t) = \boldsymbol{\Theta}^T(x_i)\mathbf{T}(t) \quad (5.10.1)$$

then the flux components are

$$q_i(x_i, t) = -k \frac{\partial \boldsymbol{\Theta}^T}{\partial x_i} \mathbf{T}(t) \quad (5.10.2)$$

The relation in (5.10.2) is valid for any element and allows the flux to be evaluated at any time and at any point within the element. Note that in evaluating the derivative of $\boldsymbol{\Theta}$, the element geometry must be considered for mapped or parametric elements [see Eq. (3.3.3)]. Maximum accuracy in the shape function derivative is obtained by evaluating the derivative at the element numerical integration points (typically, the 2×2 Gauss points in a quadrilateral element) [25]. Since fluxes are normally required on element or domain boundaries, the integration point values are extrapolated to the nodes [26] and the discontinuous element level fluxes averaged between adjacent elements. Other evaluation points within an element have been employed, such as the points midway between nodes on the element boundary, or the Gauss points on

the element boundary. All of these methods provide results of acceptable accuracy provided the computational mesh is sufficiently refined. Note that a common feature of these procedures is the need to compute a unit normal on an element boundary if a total or integrated flux is to be derived. Methods for determining unique and consistent normals to element boundaries are discussed by Engleman, et al. [27].

A second, less familiar, approach for computing flux quantities relies on the consistent application of the weighted residual method. This idea was proposed by Larock and Herrmann [28] and Marshall, et al. [29], and refined by Gresho, et al. [30]. In essence, a weighted residual equation is derived that relates the boundary flux to local heat sources and computed temperature gradients. The consistent flux equation requires a matrix solution, which for many cases can be simplified by the use of a “lumping” procedure. Though somewhat more accurate than the simpler basis function derivative method, this procedure requires more computation, especially for the advection-diffusion equation. For details of this procedure, see [30].

It is important to recall that the above methods were described for the case of a conductive heat flux as would be computed for a solid boundary or a no flow-through surface. The heat flux computation for an open flow boundary would have to include terms describing the advective transport of energy. The appropriate flux description for this case would be

$$q_i(x_i, t) = \rho C_v v_i T - k \frac{\partial T}{\partial x_i} \quad (5.10.3)$$

and using the usual interpolations for velocity and temperature produces

$$q_i(x_i, t) = \rho C_v \boldsymbol{\Psi}^T \mathbf{v}_i(t) - k \frac{\partial \boldsymbol{\Theta}^T}{\partial x_i} \mathbf{T}(t) \quad (5.10.4)$$

the computational form for the total flux. Equation (5.10.4) would be evaluated in the same manner as the conductive flux in (5.10.2) with the same extrapolation and averaging techniques.

In Section 3.10.2, the heat flow function was defined for a conduction problem in two-dimensional geometries. The definition can be extended to convective transport though the restriction to two dimensions remains. Using the fluxes defined in (5.10.3), the heat function is defined by

$$q_x = \rho C_v v_x T - k \frac{\partial T}{\partial x} = \frac{\partial \mathcal{H}}{\partial y} \quad (5.10.5)$$

$$q_y = \rho C_v v_y T - k \frac{\partial T}{\partial y} = -\frac{\partial \mathcal{H}}{\partial x} \quad (5.10.6)$$

As before, the change in the heat function is an exact differential and

$$\delta \mathcal{H} = \int_A^B \mathbf{q} \cdot \mathbf{n} d\Gamma \quad (5.10.7)$$

where \mathbf{n} is the normal to the integration path. The computation of \mathcal{H} follows from the flux definition in terms of the interpolation functions and the definition of the

normal for the element. The details were given in Section 3.10.2 for both the planar and axisymmetric cases and will not be repeated here. It is worth noting again that the integration of (5.10.7) around the boundary of an element should be zero; the departure from zero indicates the error in the element energy balance.

5.11 Turbulent Heat Transfer

The previous sections have been limited to consideration of laminar flows though the extension to turbulent flows is of considerable engineering importance. This section continues and extends the discussion of Section 4.12 on isothermal turbulence modeling to the nonisothermal case. Most of the ideas and turbulence models for isothermal flows have a direct counterpart when the energy equation is included in the formulation. Here we will outline some of the most important concepts for the case of forced convection heat transfer; the theory for strongly coupled, buoyancy driven flows is somewhat less developed.

Extending the assumption given in Eqs. (4.12.1) and (4.12.2), the instantaneous temperature and velocities can be written as

$$\theta = \Theta + \theta', \quad v_i = V_i + v'_i \quad (5.11.1)$$

where Θ is the mean temperature, V_i is the mean velocity, and the primed quantities denote the fluctuations. When this definition and Eq. (4.12.1) are substituted into the instantaneous energy equation and ensemble averaged, the result is an equation of the form

$$\rho C \left(\frac{\partial \Theta}{\partial t} + V_j \frac{\partial \Theta}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left(k \frac{\partial \Theta}{\partial x_j} \right) - \rho C \overline{v'_j \theta'} \quad (5.11.2)$$

The averaging process has introduced the three new variables $\overline{v'_j \theta'}$ which are the components of the Reynolds heat flux vector. These quantities can be modeled in a variety of ways, all of which are analogous to the methods used for the Reynolds stresses. The current status of turbulent heat transfer modeling is reviewed by Launder [31]. Considering only one point closure models (see Section 4.12.4), the Reynolds heat flux can be related to the mean temperature field through a Boussinesq type hypothesis. That is, let

$$-\rho C \overline{v'_j \theta'} = \frac{k_T}{C} \frac{\partial \Theta}{\partial x_j} = \frac{\mu_T}{Pr_T} \frac{\partial \Theta}{\partial x_j} \quad (5.11.3)$$

which can be used in (5.11.2) to produce

$$\rho C \left(\frac{\partial \Theta}{\partial t} + V_j \frac{\partial \Theta}{\partial x_j} \right) = \frac{\partial}{\partial x_j} \left[\left(k + \frac{\mu_T}{Pr_T} \right) \frac{\partial \Theta}{\partial x_j} \right] \quad (5.11.4)$$

where k_T is the eddy conductivity and Pr_T is the turbulent Prandtl number. Once the form of the eddy conductivity or the turbulent Prandtl number is specified, then the energy equation in (5.11.4) can be solved in conjunction with the flow equations.

Methods for specifying the eddy conductivity or turbulent Prandtl number parallel the ideas for the eddy viscosity specification. Of all the possible approaches, one of the most popular is the specification of Pr_T as an empirical relation. Reynolds

[32] describes approximately thirty different formulas for Pr_T , all of which have a limited applicability. Clearly, the modeling of turbulent heat transfer processes remains an active area of research, though its progress is directly related to progress made in turbulent flow modeling. It should also be noted that Variational Multi-scale (VMS) models for nonisothermal turbulence could be generated following the ideas of Section 4.12.6.

5.12 Chemically Reacting Systems

5.12.1 Preliminary Comments

The equations describing a reactive material were presented in Section 1.9. The incorporation of chemical kinetics into solid body heat conduction problems was outlined in Section 3.11.5. The complexity of the problem, where general chemical reactions are included with a nonisothermal flow problem, places this topic outside the primary scope of this text. In the following section a brief discussion of reactive flow modeling is provided to simply indicate the scope of the modeling problem. Readers interested in this topic should consult the specialized texts by Oran and Boris [33] or Kee, et al. [34].

5.12.2 Finite Element Modeling of Chemical Reactions

The development of a finite element model for chemically reactive flows follows the usual prescription. The continuity, momentum, and energy equations for the mixture are standard and were considered in Chapter 4 and previous sections of the present chapter. The conservation equations for the species are of the advection-diffusion type and can therefore be treated by the same methods as used for the energy equation. Recall that the auxiliary equations defined in Chapter 1 could be used directly as species equations. The discretized form of the species equations provides an additional set of I matrix equations that can, in principle, be added to the matrix system for the basic nonisothermal flow problem.

Though the development of the reactive flow model appears to be straightforward, the solution of the resulting equations is extremely difficult. First, from the standpoint of problem size, the addition of multistep chemistry leads to a very rapid growth in the number of degrees of freedom considered in any given problem. However, problem size alone is not the most significant difficulty. The equation set is highly coupled and very nonlinear, making the convergence by standard solution methods unpredictable. More importantly, the length and time scales for the chemical processes tend to be significantly different (smaller) than the scales for the fluid and thermal response. This disparity in scales makes the use of fully coupled solution procedures impractical. In mathematical terms, the equation system becomes very stiff. The most effective solution algorithms allow the flow problem to run on its natural time scale, while the chemistry solution is decoupled and run as a subcycled process on a shorter time scale. Obviously, some of the rigorous coupling between variables is destroyed by this technique and variable coefficients must be lagged or predicted and corrected. Despite the drawbacks, this type of approach is the most widely used for problems with “difficult” chemistry.

Chemical reactions also tend to produce large gradients in the dependent variables over small distances. Since the location, extent, and movement of such reaction

zones is not known a priori, excessively refined meshes are often employed to ensure spatial accuracy. Again, this leads to a large problem size and excessive computer resources. A more efficient solution to the length scale problem is the use of adaptive meshes. An example of this technique for a reactive problem can be found in reference [35].

5.13 Numerical Examples

5.13.1 Preliminary Comments

The numerical examples selected for this section are intended to illustrate the performance of the finite element models presented for convective heat transfer problems. Again, the problems are not examined in full engineering detail, but are sufficiently complete to demonstrate the flexibility and capability of the recommended algorithms. All of the examples were solved using the codes NACHOS II [36], FIDAP [37] or KACHINA [38] with the mixed finite element model, unless otherwise stated. Numerical examples of natural convection problems were solved by the penalty finite element model (see Pelletier, Reddy, and Schetz [39]).

5.13.2 Concentric Tube Flow

The first example we consider is the forced convection/conduction problem of flow in a concentric tube. The problem schematic is shown in Figure 5.13.1 along with the finite element mesh of eight-noded elements and boundary conditions. The fluid in the inner tube is a medium-weight oil and the counter-flowing cooling fluid is water. The central tube is made of copper and the outside concentric tube is made of steel. All material properties are assumed to be constant.

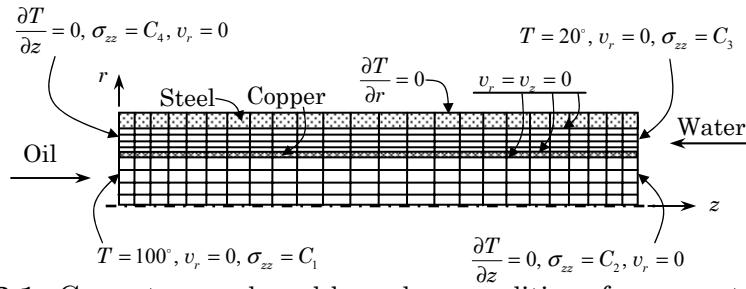


Figure 5.13.1: Geometry, mesh and boundary conditions for concentric tube flow.

As is typical for internal flow problems, flux type boundary conditions are used at the outflow boundary for both fluid regions to allow a smooth solution field without an excessive tube length. Also, in this case a specified traction boundary condition was specified at the inflow boundary; the difference in the normal tractions across the tube length produce the driving force for the fluid. Note that the nonlinear terms are identically zero in the fluid and thermal equations in this case, and therefore the problem can be solved without iteration.

The solution to the hydrodynamic part of the problem consists of a fully developed Poiseuille flow for the oil and flow in a concentric annulus for the water. Both computed profiles are essentially exact for this model. Three cases were computed in which the oil Peclet number ($Pe = \mu C_p/k$) was varied by changing the pressure drop imposed on the tube. A typical isotherm plot is shown in Figure 5.13.2. The

computed temperature profiles along the inside of the copper tube are shown in Figure 5.13.3. Further details on this analysis and other conjugate tube flow problems are available in [40].

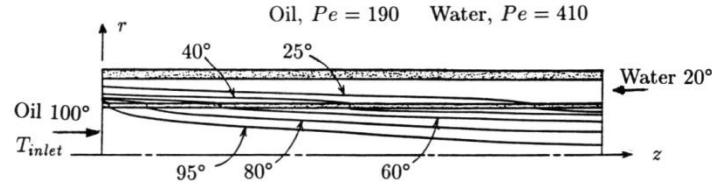


Figure 5.13.2: Isotherms for the concentric tube flow problem.

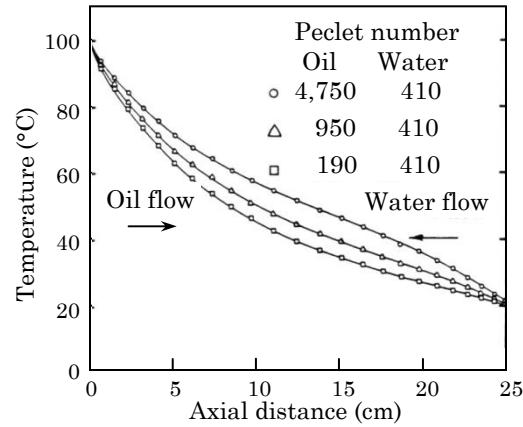


Figure 5.13.3: Axial temperature profiles for the concentric tube flow problem.

5.13.3 Tube Flow with Change of Phase

Consider the problem of a heated fluid flowing through a cold tube, the temperature of which is below the fluid solidification temperature. The problem geometry is shown in Figure 5.13.4. The fluid within the tube is initially quiescent with a uniform temperature equal to the inlet temperature. Fluid motion is initiated by imposing a specified pressure drop on the tube. The methods described in Sections 3.3 and 4.7 were used to model the latent heat release and the immobilization of the fluid at the phase boundary [19].

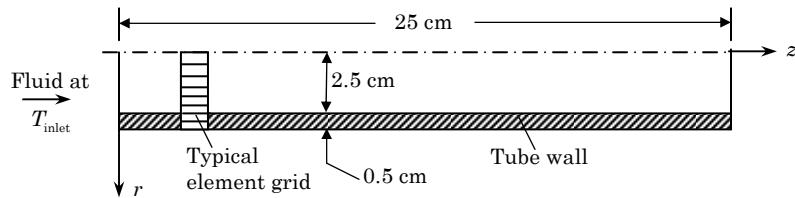


Figure 5.13.4: Schematic for tube flow with change of phase.

The initial temperature response of the fluid is due solely to conduction at the tube wall. The large initial heat loss to the tube cools the fluid below the solidus temperature, producing a solid crust on the tube wall. As the fluid velocity increases, convection along the length of the tube becomes important.

Figure 5.13.5 shows the result of a computation in which the tube eventually becomes obstructed due to the crust formation and growth. Each frame shows the phase change isotherm which corresponds to the phase boundary.

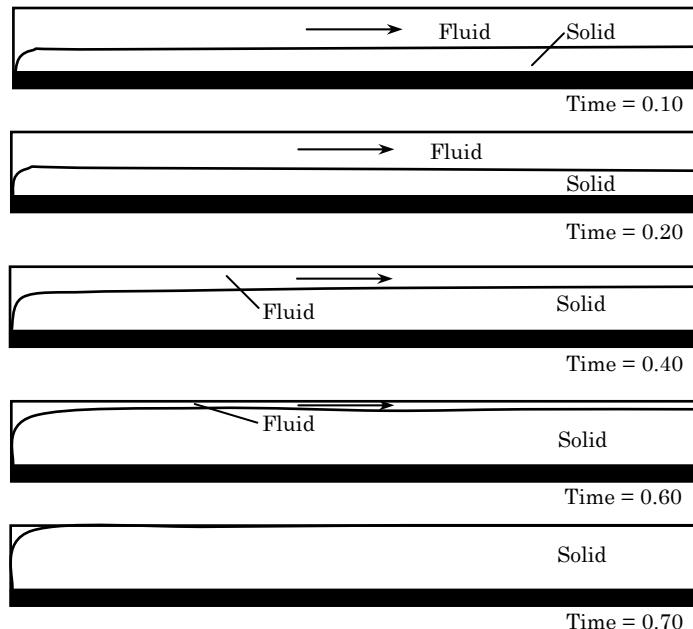


Figure 5.13.5: Motion of phase boundary for impulsively started tube flow.

Figure 5.13.6 shows velocity and temperature profiles at a station four tube diameters from the inlet. The velocity profiles show the effects of initial fluid acceleration, reduction in tube diameter due to crust growth, and finally the reduction in flow rate due to increasing viscosity.

5.13.4 Heated Cavity – Boussinesq Model

A standard benchmark problem for natural convection algorithms consists of a square, closed cavity with insulated top and bottom faces and differentially heated vertical boundaries. For the present case, the contained fluid is a gas with constant material properties. A nondimensional temperature difference of $\epsilon = (T_H - T_C)/(T_H + T_C)$ is taken. For this case, it is set to $\epsilon = 0.005$. For this small value of the temperature difference, the flow is well within the Boussinesq limit. The boundary conditions and nondimensional parameters are shown in Figure 5.13.7a. Since the velocity field is specified on all boundaries of the domain, the pressure field is determined only to within a constant value; the pressure level within the cavity is set by specifying a single pressure value at one node on the cavity boundary. A typical mesh consisting of eight-node elements is also shown in Figure 5.13.7b.

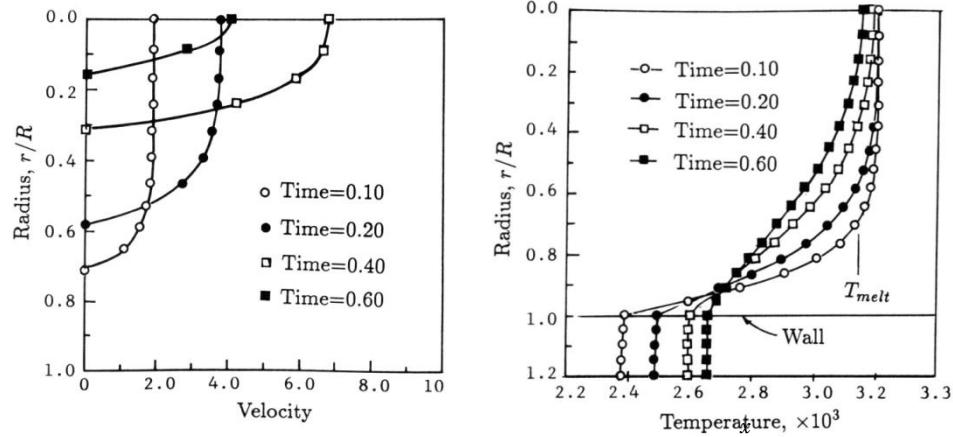


Figure 5.13.6: Velocity and temperature profiles for tube flow, $z/R = 8$.

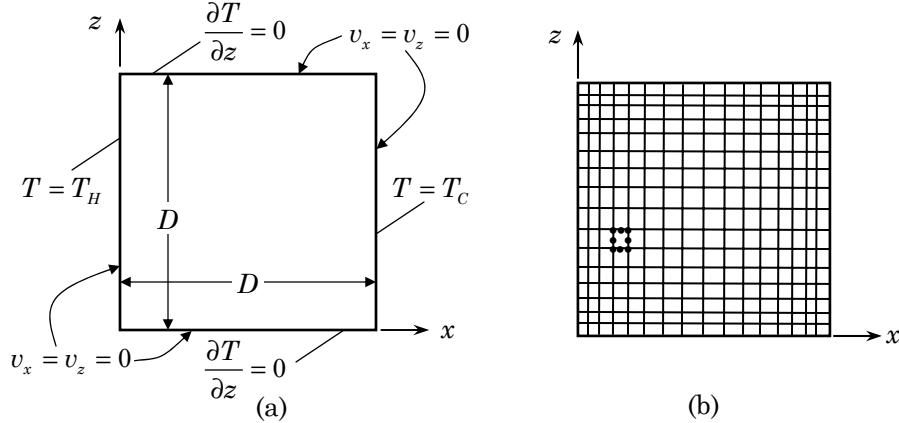


Figure 5.13.7: Schematic for natural convection in a cavity.

The cavity problem was initially solved using the cyclic Picard scheme employed in the earliest versions of the NACHOS code. More recent solutions have been obtained using Newton's method for the combined equation set and zeroth order continuation through the Rayleigh number [$Ra = \beta g(T_H - T_C)D^3/\nu\kappa$] range of interest. Results for this problem in the form of streamline and isotherm plots are shown in Figures 5.13.8 and 5.13.9 for Rayleigh numbers of $Ra = 10^4$ and 10^6 , respectively ($Pr = \nu/\kappa = 0.71$). For the lower Rayleigh number, the flow is relatively weak and the thermal field is only slightly perturbed from a conduction solution. At the higher Rayleigh number, the flow field develops a considerable structure while the thermal field becomes vertically stratified in the core of the cavity with high heat flux regions along the vertical boundaries. As required by the form of the boundary conditions and boundary value problem, the computed solutions are exactly centro-symmetric as seen in the contour plots. A detailed comparison of the present solutions with a large number of other numerical simulations is contained in [41].

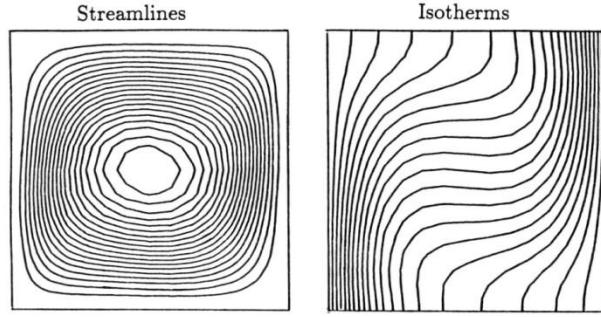


Figure 5.13.8: Streamlines and isotherms for natural convection in a square cavity, Boussinesq model ($Ra = 10^4, \epsilon = 0.005$).

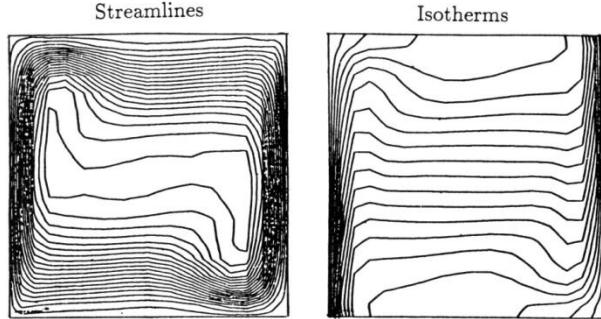


Figure 5.13.9: Streamlines and isotherms for natural convection in a square cavity, Boussinesq model ($Ra = 10^6, \epsilon = 0.005$).

5.13.5 Heated Cavity – Acoustically Filtered Model

The heated cavity flow may also be used for evaluating methods for non-Boussinesq formulations. In the present case the nondimensional temperature difference is increased to $\epsilon = 0.6$ which is outside the Boussinesq range. The flow problem in the previous section is again considered with the same Rayleigh numbers of $Ra = 10^4$ and 10^6 . For this simulation, nine-node elements with a discontinuous pressure approximation were used on a series of regular mesh refinements. Shown in Figures 5.13.10 and 5.13.11 are the acoustically filtered results in terms of the stream function and temperature fields. When compared to the previous Boussinesq results, it is immediately apparent that no symmetries are present. Quantitative comparisons of the heat transfer rates (Nusselt number) have been made with various other simulations in [4] and found to be in excellent agreement. Full Newton iteration was used to compute the acoustically filtered results and the number of iterations for convergence was higher than the comparable Boussinesq case. Although not shown here, a fully compressible fluid model has also been computed and compared to the acoustically filtered result. There is no significant difference between the results which indicates the validity of the acoustically filtered formulation for low-speed, compressible flows.

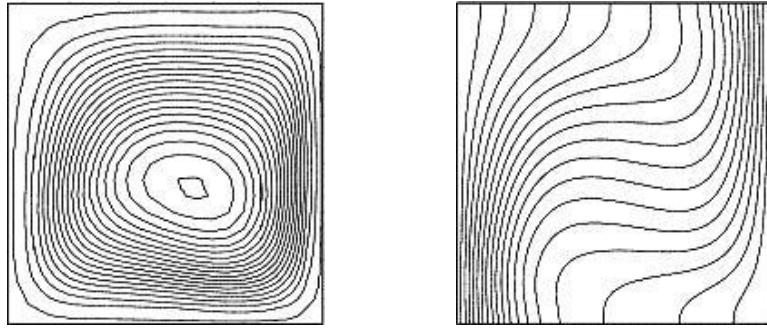


Figure 5.13.10: Streamlines and isotherms for natural convection in a square cavity, Acoustically filtered model ($Ra = 10^4, \epsilon = 0.6$).

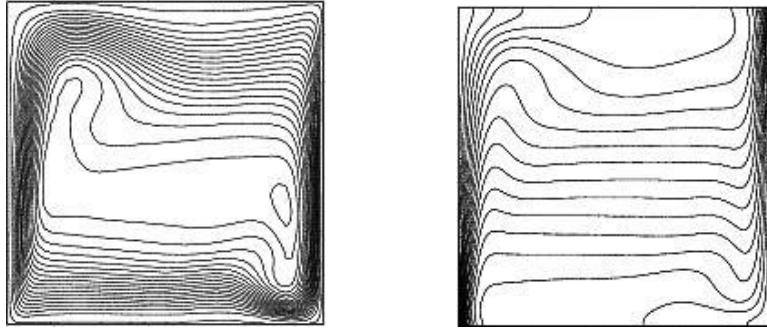


Figure 5.13.11: Streamlines and isotherms for natural convection in a square cavity, Acoustically filtered model ($Ra = 10^6, \epsilon = 0.6$).

5.13.6 Solar Receiver

The schematic shown in Figure 5.13.12 represents a cross section of an annular solar receiver tube surrounded by an eccentrically located glass envelope. The inner tube carries a heat transfer fluid that is heated by a flux that varies with position around the tube; the incident flux is due to solar energy being concentrated on the tube by a parabolic trough collector. The glass envelope provides a shield to reduce the forced convection (wind) heat loss from the collector tube. Of interest in this problem is the prediction of heat loss from the inner tube due to natural convection in the annular space. Parameters such as inner tube temperature distribution, working fluid in the annulus, annular gap, and eccentricity of the envelope were varied to assess their importance on overall heat loss.

Figures 5.13.13 through 5.13.16 contain streamline and isotherm plots for an air-filled annulus for various temperature and geometric configurations. The flow pattern and heat flux distribution is quite sensitive to variations in these parameters even though the Rayleigh number is the same for all cases. The computed heat transfer results for these cases are compared with experimental data in Figure 5.13.17.

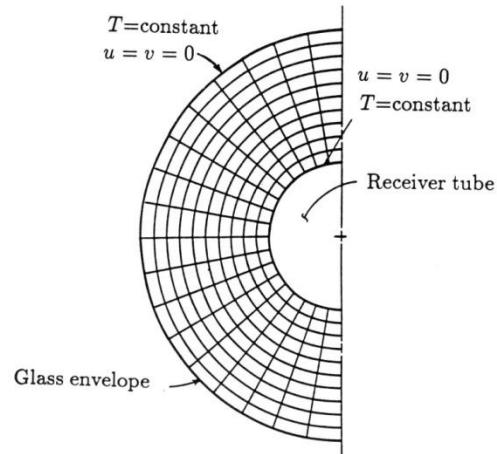


Figure 5.13.12: Mesh and boundary conditions for the annular solar receiver.

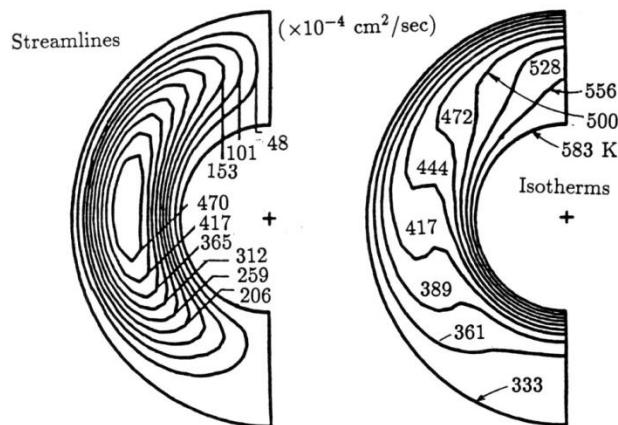


Figure 5.13.13: Streamline and isotherm plots for the solar receiver; *uniform wall temperature*, $Ra = 1.2 \times 10^4$.

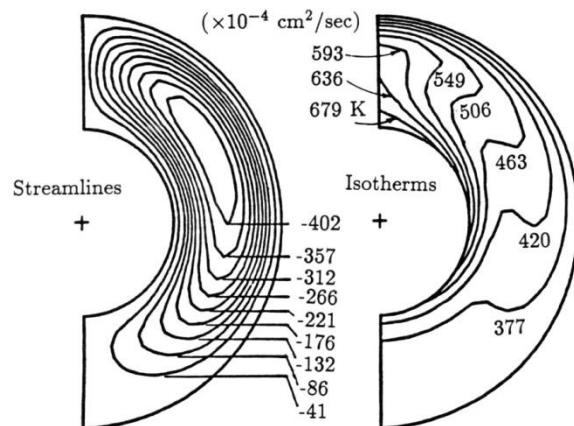


Figure 5.13.14: Streamline and isotherm plots for the solar receiver; *asymmetric wall temperature, hot on top* ($Ra = 1.2 \times 10^4$).

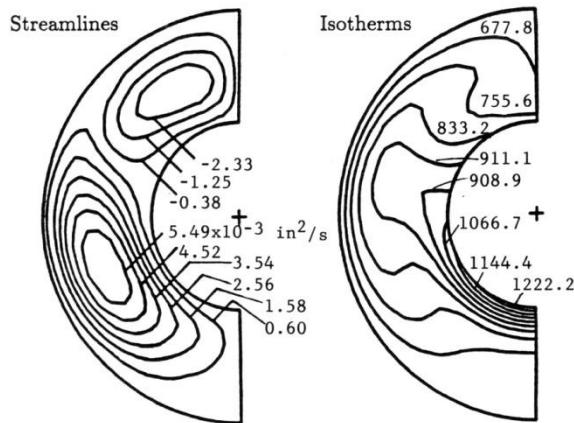


Figure 5.13.15: Streamline and isotherm plots for the solar receiver; uniform wall temperature, *hot on bottom* ($Ra = 1.2 \times 10^4$).

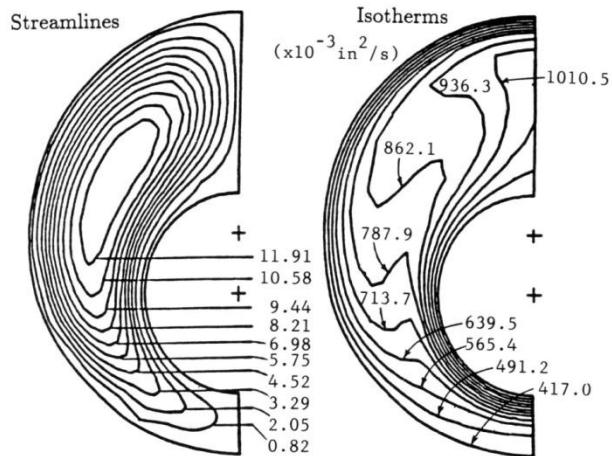


Figure 5.13.16: Streamline and isotherm plots for the solar receiver; uniform wall temperature, *eccentric geometry* ($Ra = 1.2 \times 10^4$).

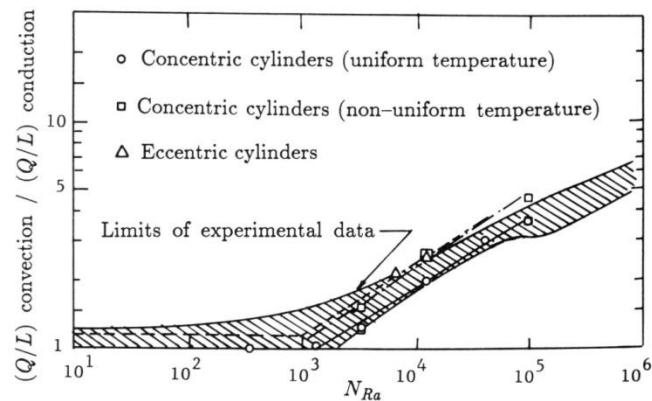


Figure 5.13.17: Heat loss by natural convection in annular spaces of a solar receiver. Computed points are shown as symbols.

5.13.7 Tube Bundle

The natural convection heat transfer from three tubes contained within a larger diameter conduit is modeled with the finite element grids shown in Figure 5.13.18. The geometry is symmetric, allowing the simulation to be performed on one-half of the conduit domain. The interior tubes have constant, but different wall temperatures; the exterior of the conduit is held at a temperature that is lower than the interior tubes. The two meshes shown in Figure 5.13.18 represent two very different mesh generation techniques. The more structured mesh was developed in three sections using a standard mapping technique. The completely unstructured quadrilateral mesh was produced using a method called “paving” [42] which works with the entire region at one time. The only input to the paving algorithm, other than a boundary description, is an element size along each boundary.

Isotherm and streamline plots for two cases are shown in Figures 5.13.19 and 5.13.20. Figure 5.13.19 illustrates the flow produced when a small temperature difference ($\Delta T_{max} = 10$) is maintained between the hottest interior tube and the exterior boundary. The second case shows the result when the temperature difference from the first case is doubled (i.e., $\Delta T_{max} = 20$). Solutions for both cases were obtained using a solution strategy with several Picard iterations followed by a Newton procedure. The computed flow fields show a characteristic plume rising from the lower cylinder with an overall circulation within the larger conduit.

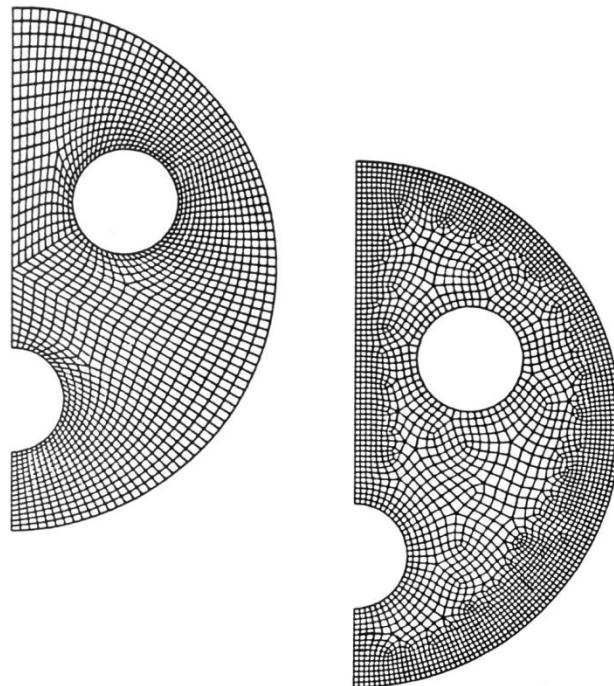


Figure 5.13.18: Finite element meshes for tube bundle geometry.

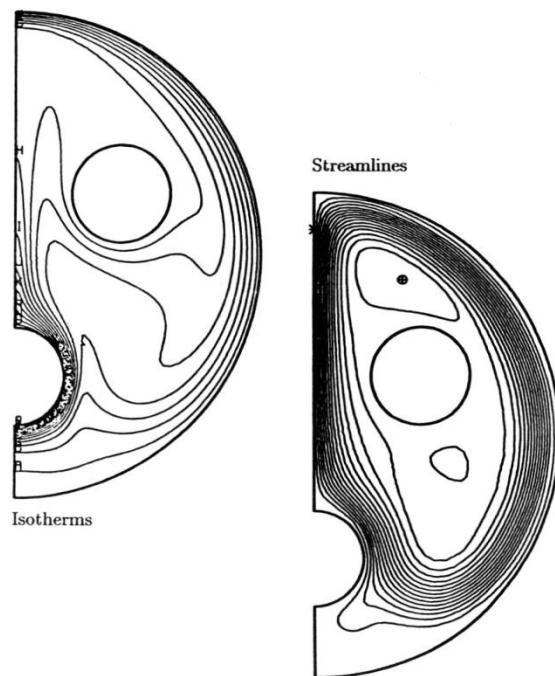


Figure 5.13.19: Contour plots for natural convection in a tube bundle, $\Delta T_{max} = 10$.

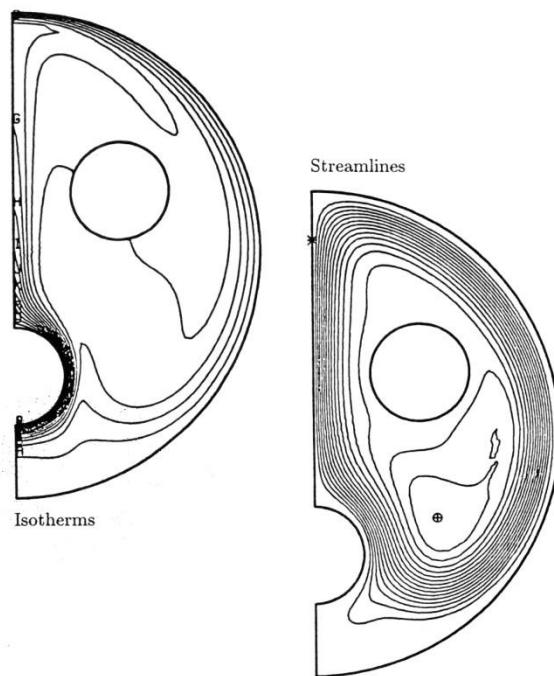


Figure 5.13.20: Contour plots for natural convection in a tube bundle, $\Delta T_{max} = 20$.

5.13.8 Volumetrically Heated Fluid

This example consists of a volumetrically heated fluid contained in an isothermal circular cylindrical container. Many of these types of flows are associated with heat transfer from radioactive gases or liquids. A quadrant of the domain, boundary conditions, and finite element mesh are shown in Figure 5.13.21. At time zero, the initially quiescent, isothermal fluid is heated volumetrically at a uniform rate. The transient solution is obtained by an implicit integration scheme using the trapezoid rule with quasi-linearization.

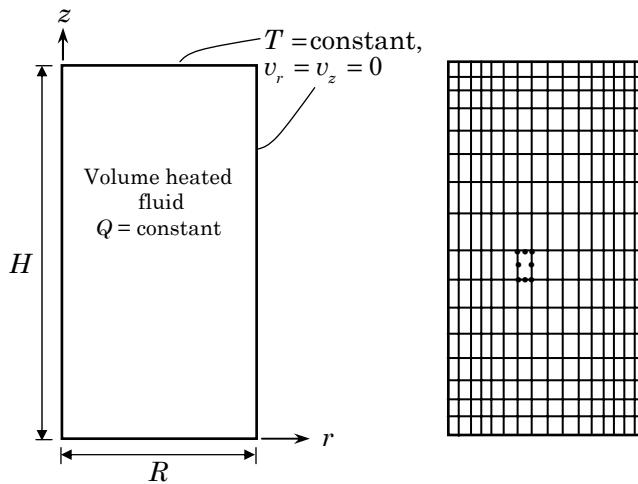


Figure 5.13.21: Computational domain, boundary conditions, and mesh used for volumetrically heated fluid in a circular cylinder.

The series of streamline and isotherm plots in Figure 5.13.22 shows the development of the flow at a moderate modified Grashof number, $Gr = g\beta l^5 Q/\nu^2 k = 2 \times 10^5$. A time history of the maximum stream function shown in Figure 5.13.23 shows the overshoot that occurs during the initial transient. The stream function vs. time plot of Figure 5.13.24 shows the interaction between the two cells at a slightly higher Grashof number, $Gr = 4 \times 10^5$. A further series of plots in Figures 5.13.25 and 5.13.26 illustrate the additional dynamics present in the flow at Grashof number, $Gr = 4 \times 10^5$. The development of a secondary cell that undergoes a damped oscillation is quite evident. The development of secondary cells is common in volumetrically heated fluids. Further details of this analysis are available in [43].

5.13.9 Porous/Fluid Layer

A simple problem that illustrates the use of the saturated porous media formulation in conjunction with an incompressible, viscous flow is shown in Figure 5.13.27. A rectangular geometry is divided vertically with a porous material occupying the right half of the cavity. The top and bottom boundaries of the cavity are insulated while the vertical boundaries are held at uniform but different temperatures. The steady natural convection problem in the cavity was modeled using a Brinkman model for flow in the porous medium. Newton's method was used to solve the flow problem.

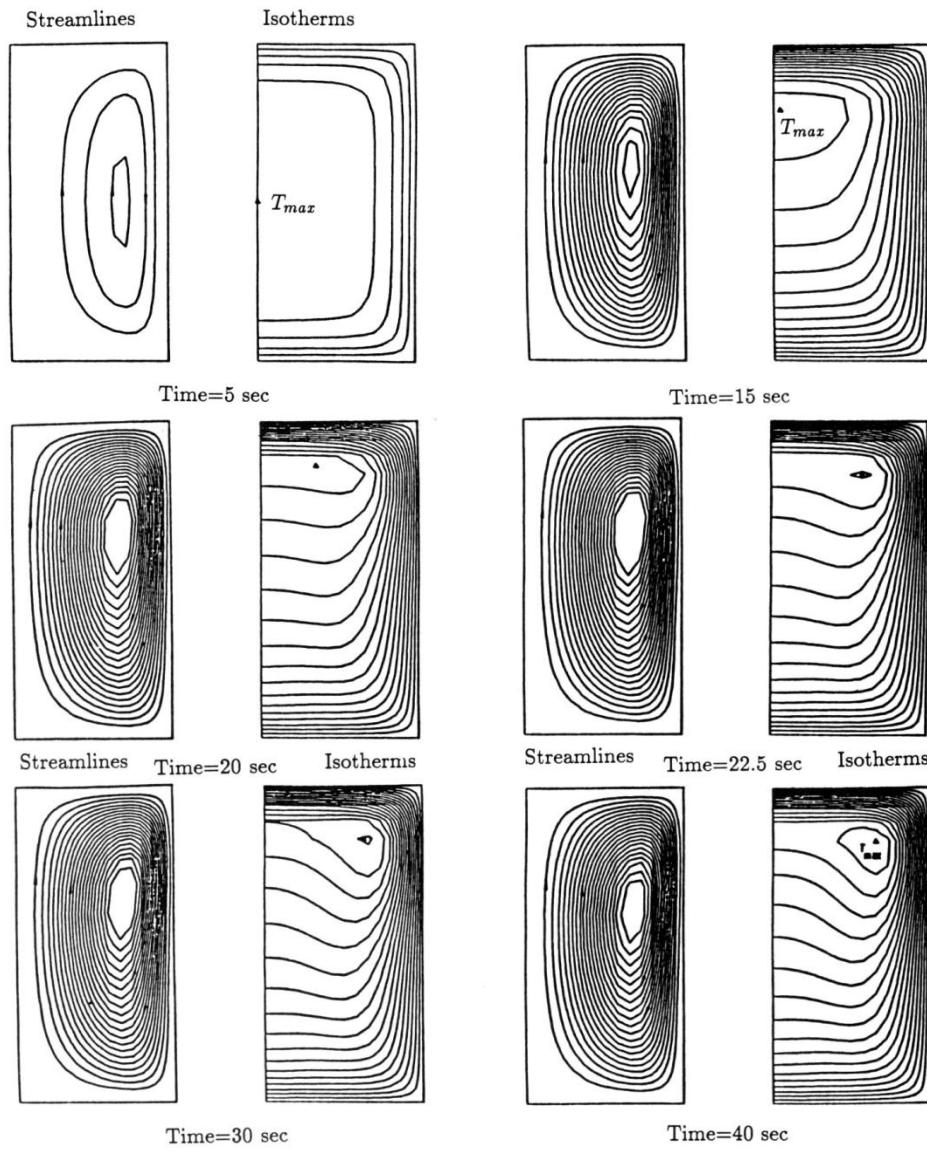


Figure 5.13.22: Streamlines and isotherms for volumetrically heated fluid ($Gr = 2 \times 10^5$).

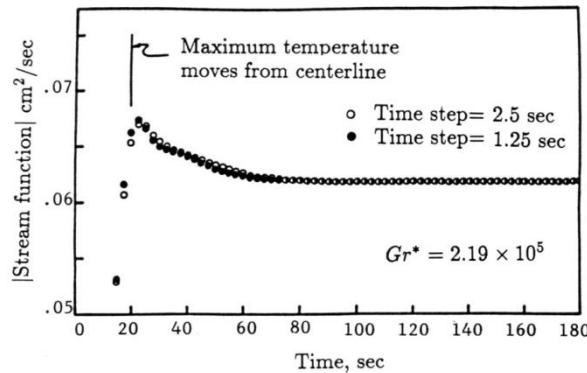


Figure 5.13.23: Stream function vs. time for volumetrically heated fluid ($Gr = 2 \times 10^5$).

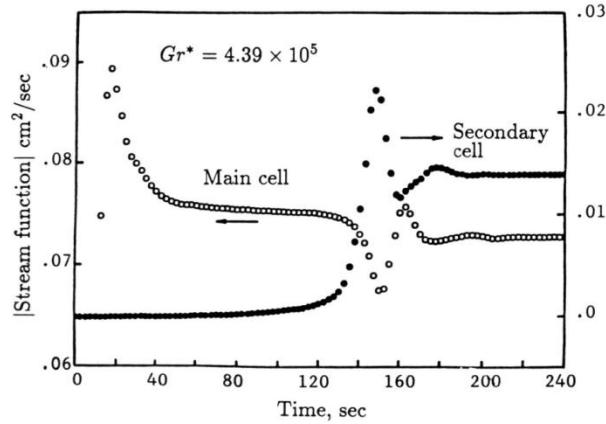


Figure 5.13.24: Stream function vs. time for volumetrically heated fluid ($Gr = 4 \times 10^5$).

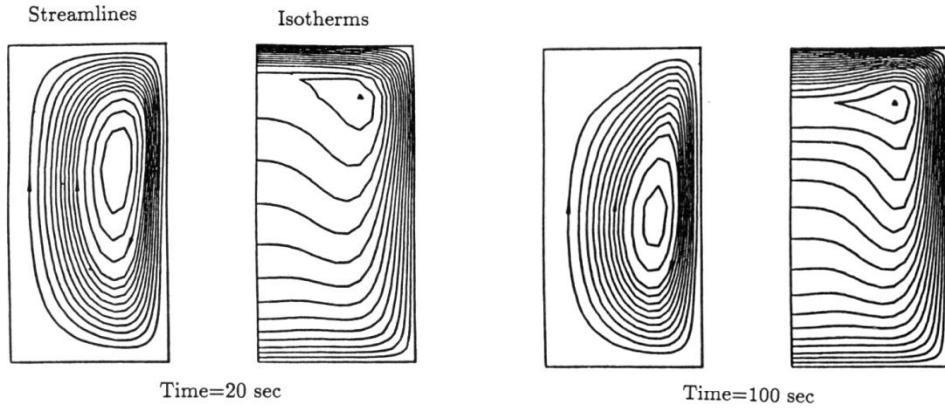


Figure 5.13.25: Streamlines and isotherms for volumetrically heated fluid ($Gr = 4 \times 10^5$).

For small values of the Darcy number (or permeability), $Da = \sqrt{k}/L$, the dominant fluid motion is confined to the fluid layer with little circulation occurring in the porous layer. At higher Darcy numbers, the fluid motion spans the cavity, although the velocities in the porous layer are still much smaller than in the bulk fluid. Representative stream function and isotherm plots are shown in Figures 5.8.28 and 5.8.29 for extremes in the Darcy number. Further discussion on the modeling of conjugate problems and the use of various porous/layer/bulk fluid interface conditions can be found in [22] and [44].

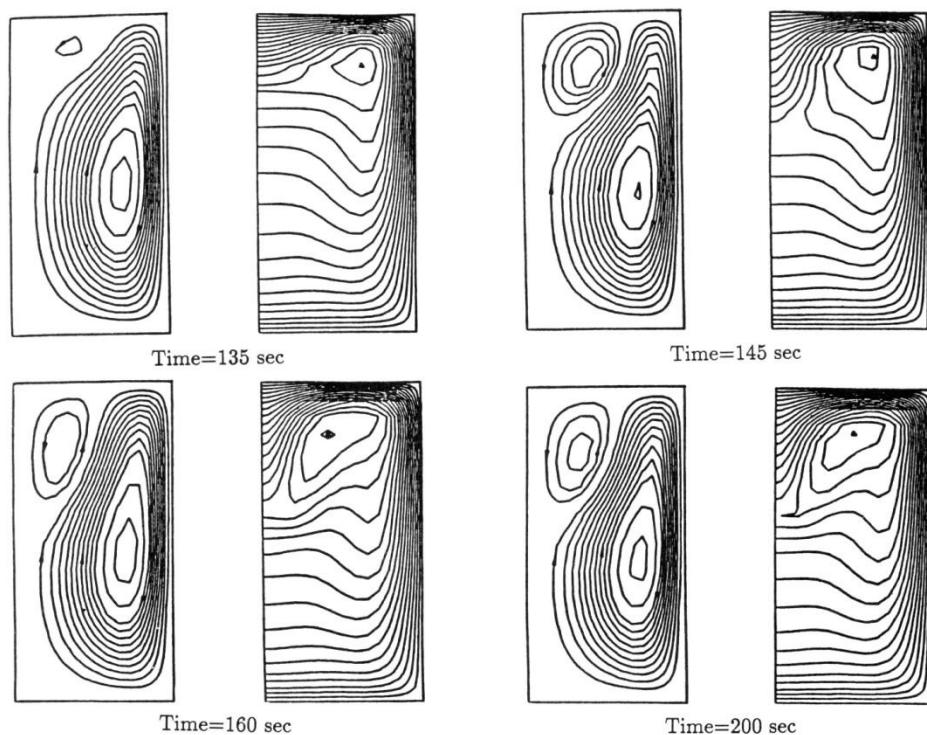


Figure 5.13.26: Streamlines and isotherms for volumetrically heated fluid ($Gr = 4 \times 10^5$).

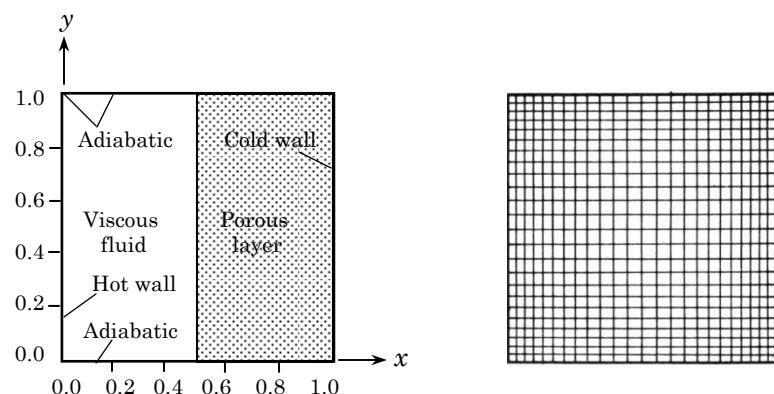


Figure 5.13.27: Schematic and mesh for a conjugate, porous/fluid layer problem.

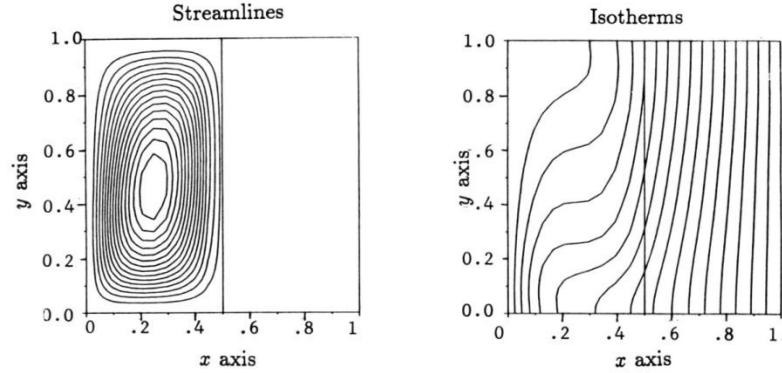


Figure 5.13.28: Streamlines and isotherms for a conjugate, porous/fluid layer ($Ra = 10^5$, $Da = 10^{-5}$).

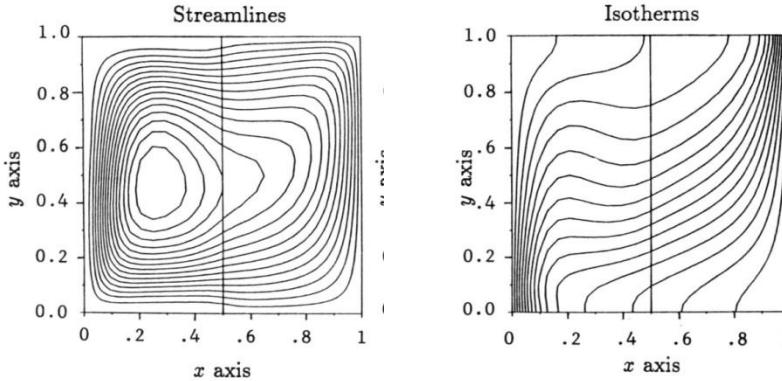


Figure 5.13.29: Streamlines and isotherms for a conjugate, porous/fluid layer ($Ra = 10^5$, $Da = 10^{-3}$).

5.13.10 Curing of an Epoxy

This example demonstrates the use of both the phase change procedures and the auxiliary equation formulation to predict the gelation behavior of an epoxy. The geometry for the simulation is a simple rectangular crucible that initially contains a well-stirred mixture of epoxy resin and curing agent (see Figure 5.13.30). The crucible is held in a constant-temperature oven during the course of the curing process. Since the curing reaction is exothermic, the volumetric heat addition to the fluid is given by

$$Q = \rho \Delta H \frac{D\alpha}{Dt} = \rho (C_p^p - C_p^r) T \frac{D\alpha}{Dt} \quad (5.13.1)$$

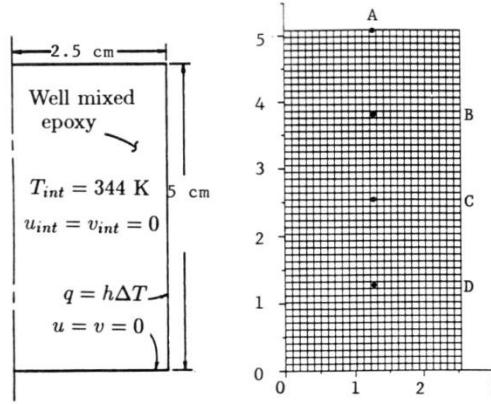


Figure 5.13.30: Schematic and mesh for epoxy curing simulation.

where ΔH is the change in enthalpy of the reacting material due to its change in state, and the superscripts p and r denote the products and reactants, respectively. The variable α is an internal state variable that describes the extent of the gelation reaction; α can be interpreted as the ratio of the mass of reacted material (per unit volume) to the total mass of material (per unit volume). Therefore, α varies between 0 (unreacted) to 1 (fully reacted) with $\alpha = 0.6$ indicating the point of gelation (solidification). The time evolution of the extent of reaction variable, α , is provided by an advection-diffusion equation

$$\frac{\partial \alpha}{\partial t} + v_j \frac{\partial \alpha}{\partial x_j} = \frac{\partial}{\partial x_j} \left(D \frac{\partial \alpha}{\partial x_j} \right) + R \quad (5.13.2)$$

where R is the reaction rate. For this particular material the reaction rate is often assumed to be of second-order with Arrhenius rate constants

$$R = [A_1 \exp(-E_1/\mathcal{R}T) + \alpha A_2 \exp(-E_2/\mathcal{R}T)] (1 - \alpha) \quad (5.13.3)$$

where A_1, A_2 are pre-exponential factors, E_1, E_2 are activation energies, and \mathcal{R} is the gas constant.

Equations (5.13.1) through (5.13.3) must be coupled to the standard nonisothermal, Navier-Stokes equations to describe the entire curing process. In addition, material properties, which vary with the extent of the reaction and the temperature, must be specified. The present problem was solved using the adaptive time step version of the trapezoid rule. A finite element model with quadratic interpolation of the velocity, temperature, and extent of the reaction, and a discontinuous linear approximation of pressure is used [45]. A time history plot of the adaptive time step employed for the analysis is shown in Figure 5.13.31. It is quite evident that the size of the time step reflects the changing complexity of the flow problem. Figures 5.13.32 through 5.13.35 show contour plots of the field variables at four different times during the curing process. As with most volumetrically heated fluids, multiple convection cells are predicted during the low heat release part of the process. When the exothermic reaction begins to accelerate, gelation occurs first at the top of the crucible and proceeds downward as a planar front. A comparison of the velocity of the gel front with the limited experimental data available showed good agreement, considering the uncertainty in material properties.

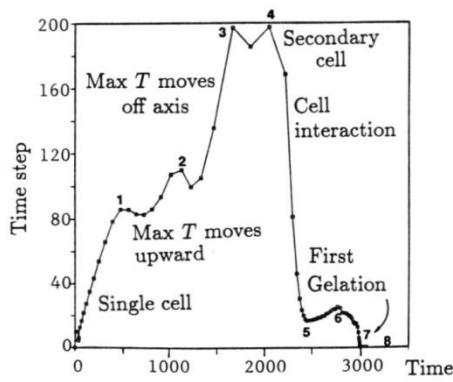


Figure 5.13.31: Time step history for epoxy curing simulation.

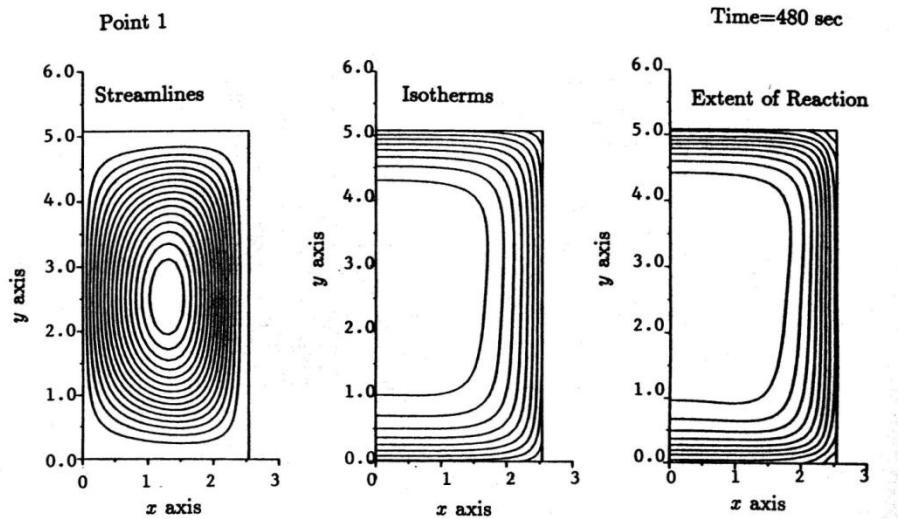


Figure 5.13.32: Contour plots for epoxy curing simulation, time = 480 sec.

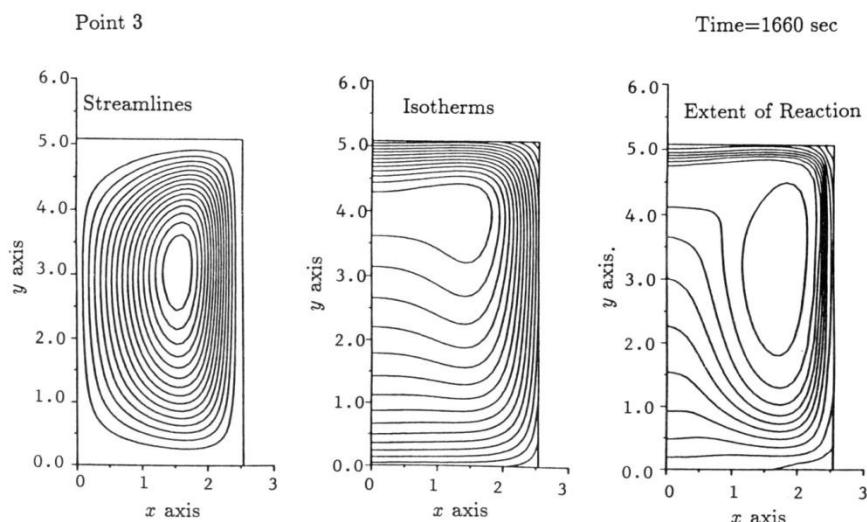


Figure 5.13.33: Contour plots for epoxy curing simulation, time = 1660 sec.

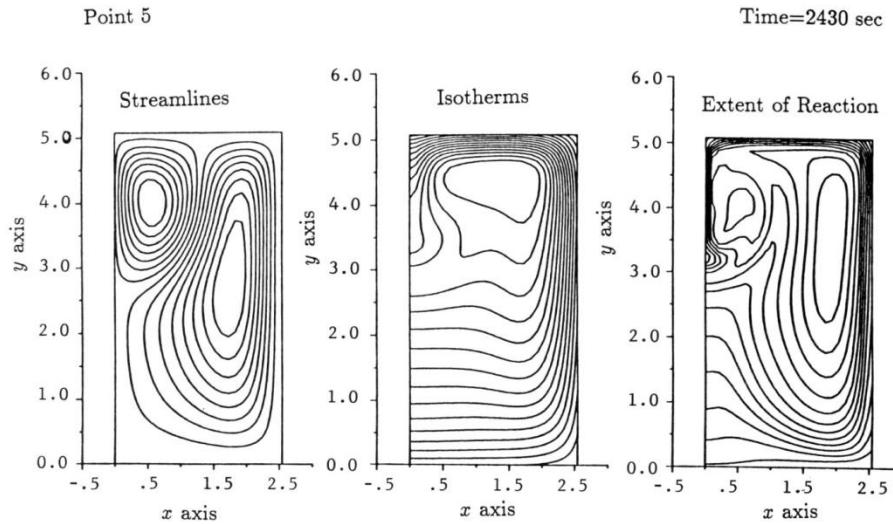


Figure 5.13.34: Contour plots for epoxy curing simulation, time = 2,340 sec.

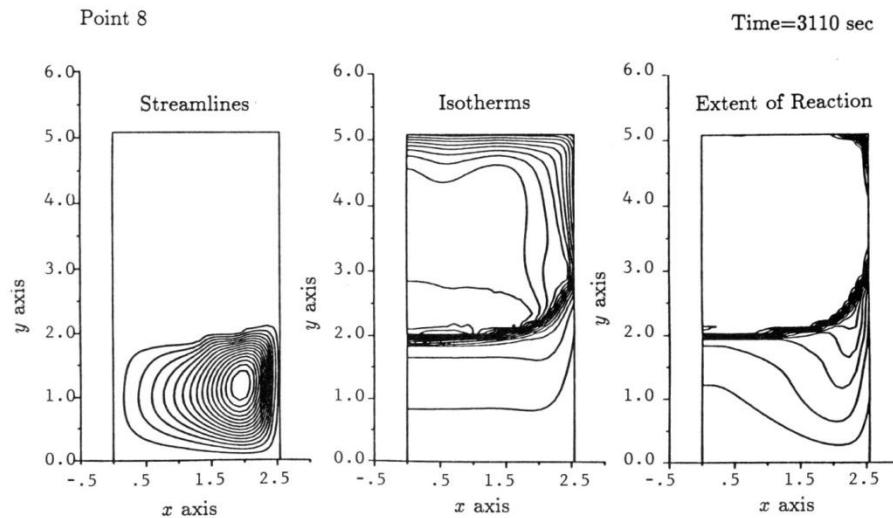


Figure 5.13.35: Contour plots for epoxy curing simulation, time = 3,110 sec.

5.13.11 Heated Channel

The final problem provides an example of the differences between the Boussinesq and acoustically filtered formulations for a time dependent flow. The problem consists of a two-dimensional channel with a heated section in the floor of the channel, as shown in Figure 5.13.36. A fully developed isothermal flow is specified as an initial condition and the transient behavior is produced by the time dependent heating of the floor section. The temperature of the heated section is increased with a linear ramp up to the final steady temperature that is higher than the temperature at the channel inlet. The problem is cast in a nondimensional form with a Reynolds number of $Re = 10$ and a Richardson number of $Ri = Gr/Re^2 = 500$, which indicates the relative importance of buoyancy and inertia forces. Using a perfect gas model, the problem was solved with both the Boussinesq and acoustically filtered equations.

Higher-order elements in a refined mesh were used for the model and an adaptive time stepping, predictor-corrector integration method was employed.

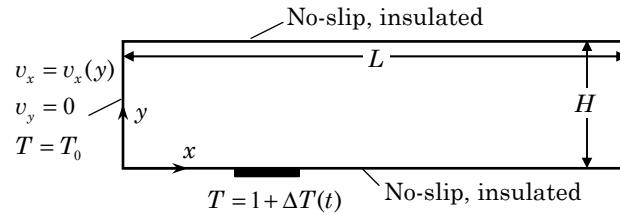


Figure 5.13.36: Schematic of heated channel flow.

Shown in Figure 5.13.37 are a series of isotherm plots for the two formulations. The differences in the fields are clear. A more significant difference in the behavior of the two models can be seen in Figure 5.13.38 where the normalized mass flux at the channel exit is plotted as a function of nondimensional time. The Boussinesq model shows a constant unit efflux while the acoustically filtered model shows an increase in the mass flux while the fluid is heated, followed by a decrease to a unit value at steady state. The differences in the two acoustically filtered curves are due to the use of constant or variable thermophysical properties. Full details of this simulation are available in [4].

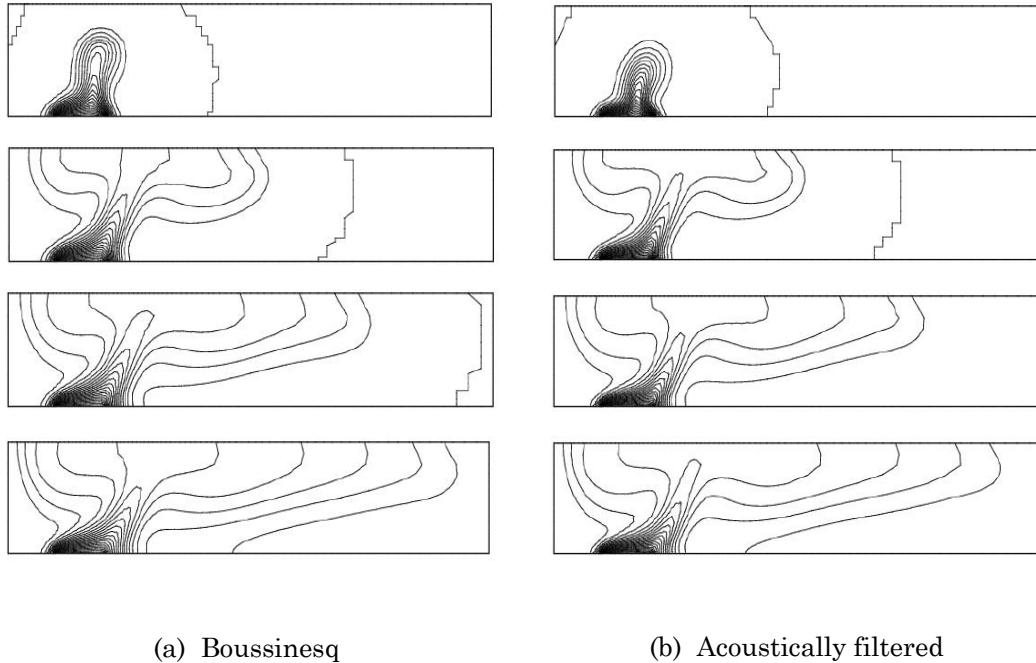


Figure 5.13.37 Isotherms for Boussinesq (left) and acoustically filtered (right) models in heated channel flow at time $t = 0.2, 0.6, 1.0$, and 1.4 .

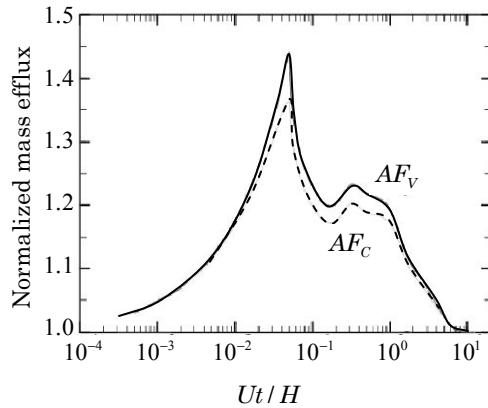


Figure 5.13.38: Normalized mass flux history for channel flow. Boussinesq formulation produces a unit mass flux for all time.

5.13.12 Closure

The numerical examples presented in this section were all based on weak-form Galerkin (mixed or penalty) finite elements models of the Navier-Stokes equations coupled with the energy equation. Coupled fluid flow and heat transfer (i.e., convective heat transfer) problems were also studied using the least-squares finite element models by Bell and Surana [46], Tang and Tsang [47], Surana, et al. [48], Pontaza and Reddy [49], and Prabhakar and Reddy [50].

References for Additional Reading

1. S. Paolucci, “On the Filtering of Sound from the Navier-Stokes Equations,” Sandia National Laboratories Technical Report, SAND82-8253, Albuquerque, NM (1982).
2. A. Majda and J. Sethian, “The Derivation and Numerical Solution of the Equations for Zero Mach Number Combustion,” *Combustion Science and Technology*, **42**, 185–205 (1985).
3. J. Principe and R. Codina, “Mathematical Models for Thermally Coupled Low Speed Flows,” *Advances in Theoretical and Applied Mechanics*, **2**, 93–112 (2009).
4. M. J. Martinez and D. K. Gartling, “A Finite Element Method for Low-Speed Compressible Flows,” *Computer Methods in Applied Mechanics and Engineering*, **193**, 1959–1979 (2004).
5. R. L. Lee, P. M. Gresho, S. T. Chan and M. J. P. Cullen, “Conservation Laws for Primitive Variable Formulations of the Incompressible Flow Equations Using the Galerkin Finite Element Method,” in *Finite Elements in Fluids, Volume 4*, R. H. Gallagher, A. H. Norrie, J. T. Oden and O. C. Zienkiewicz (eds.), John Wiley & Sons, Chichester, U. K. (1982).
6. K. A. Cliffe, “On the Conservative Finite Element Formulations of the Inviscid Boussinesq Equations,” *International Journal for Numerical Methods in Fluids*, **1**, 117–127 (1981).
7. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).
8. G. Hauke and T. J. R. Hughes, “A Unified Approach to Compressible and Incompressible Flows,” *Computer Methods in Applied Mechanics and Engineering*, **113**, 389–395 (1994).
9. G. Hauke and T. J. R. Hughes, “A Comparative Study of Different Sets of Variables for Solving Compressible and Incompressible Flows,” *Computer Methods in Applied Mechanics and Engineering*, **153**, 1–44 (1998).

10. M. J. Martinez, "Analysis of Anelastic Flow and Numerical Treatment via Finite Elements," Sandia National Laboratories Technical Report, SAND94-0320, Albuquerque, NM (1994).
11. C. Li and R. Glowinski, "Modeling and Numerical Simulation of Low Mach Number Compressible Flows," *International Journal for Numerical Methods in Fluids*, **23**, 77–103 (1996).
12. V. Heuveline, "On the Higher-Order Mixed FEM for Low Mach Number Flows: Applications to a Natural Convection Benchmark Problem," *International Journal for Numerical Methods in Fluids*, **41**, 1339–1356 (2003).
13. D. K. Gartling, "Convective Heat Transfer Analysis by the Finite Element Method," *Computer Methods in Applied Mechanics and Engineering*, **12**, 365–382 (1977).
14. B. Tabarrok and R. C. Lin, "Finite Element Analysis of Free Convection Flows," *International Journal of Heat and Mass Transfer*, **20**, 953–960 (1977).
15. J. N. Reddy and A. Satake, "A Comparison of Various Finite Element Models of Natural Convection in Enclosures," *Journal of Heat Transfer*, **102**, 659–666 (1980).
16. W. N. R. Stevens, "Finite Element, Stream Function-Vorticity Solution of Steady Laminar Natural Convection," *Int. Journal for Numerical Methods in Fluids*, **2**, 349–366 (1982).
17. V. Haroutunian, M. S. Engelmann, and I. Hasbani, "Segregated Finite Element Algorithms for the Numerical Solution of Large-Scale Incompressible Flow Problems," *International Journal for Numerical Methods in Fluids*, **17**, 323–348 (1993).
18. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, McGraw-Hill, New York (1980).
19. D. K. Gartling, "Finite Element Analysis of Convective Heat Transfer Problems with Phase Change," *Computer Methods in Fluids*, K. Morgan, C. Taylor, and C. A. Brebbia (eds.), Pentech Press, London, U. K. (1974).
20. C. Beckermann and R. Viskanta, "Double-Diffusive Convection During Dendritic Solidification of a Binary Mixture," *Physico Chemical Hydro-dynamics*, **10**, 195–213 (1988).
21. P. J. Prescott, F. P. Incropera and W. D. Bennion, "Modeling of Dendritic Solidification Systems: Reassessment of the Continuum Momentum Equation," *International Journal of Heat and Mass Transfer*, **34**, 2351–2359 (1991).
22. D. K. Gartling, C. E. Hickox and R. C. Givler, "Simulations of Coupled Viscous and Porous Flow Problems," *International Journal of Computational Fluid Dynamics*, **7**, 23–48 (1996).
23. V. R. Voller, A. D. Brent and C. Prakash, "The Modeling of Heat, Mass and Solute Transport in Solidification Systems," *International Journal of Heat and Mass Transfer*, **32**, 1719–1731 (1989).
24. D. K. Gartling and P. A. Sackinger, "Finite Element Simulation of Vacuum Arc Remelting," *International Journal for Numerical Methods in Fluids*, **24**, 1271–1289 (1997).
25. D. J. Naylor, "Stresses in Nearly Incompressible Materials by Finite Elements with Application to the Calculation of Excess Pore Pressures," *International Journal for Numerical Methods in Engineering*, **8**, 443–460 (1974).
26. E. Hinton, F. C. Scott, and R. E. Ricketts, "Local Least Squares Stress Smoothing for Parabolic Isoparametric Elements," *International Journal for Numerical Methods in Engineering*, **9**, 235–256 (1975).
27. M. Engelmann, R. L. Sani, and P. M. Gresho, "The Implementation of Normal and/or Tangential Boundary Conditions in Finite Element Codes for Incompressible Fluid Flow," *International Journal for Numerical Methods in Fluids*, **2**, 225–238 (1982).
28. B. E. Larock and L. R. Herrmann, "Improved Flux Prediction Using Low Order Finite Elements," in *Proc. First International Conference on Finite Elements in Water Resources*, Gray and Pinder (eds.), Pentech Press, London, U. K. (1977).
29. R. S. Marshall, J. C. Heinrich and O. C. Zienkiewicz, "Natural Convection in a Square Enclosure by a Finite Element, Penalty Function Method, Using Primitive Fluid Variables," *Numerical Heat Transfer*, **1**, 315–330 (1978).

30. P. M. Gresho, R. L. Lee, and R. L. Sani, "The Consistent Method for Computing Derived Boundary Quantities When the Galerkin FEM is Used to Solve Thermal and/or Fluids Problems," in *Proc. Second International Conference on Numerical Methods in Thermal Problems*, Venice, Italy (1981).
31. B. Launder, "On the Computation of Convective Heat Transfer in Complex Turbulent Flows," *Journal of Heat Transfer*, **110**, 1112–1128 (1988).
32. A. J. Reynolds, "The Prediction of Turbulent Prandtl and Schmidt Numbers," *International Journal of Heat and Mass Transfer*, **18**, 1055–1069 (1975).
33. E. S. Oran and J. P. Boris, *Numerical Simulation of Reactive Flows*, Elsevier, New York (1987).
34. R. J. Kee, M. E. Coltrin and P. Glarborg, *Chemically Reactive Flow, Theory and Practice*, John Wiley & Sons, New York (2003).
35. M. R. Baer, R. E. Benner, R. J. Gross, and J. W. Nunziato, "Modeling and Computation of Deflagration-to-Detonation Transition (DDT) in Reactive Granular Materials," *Lectures in Applied Mathematics*, **24**, American Mathematical Society, Providence, Rhode Island (1986).
36. D. K. Gartling, "NACHOS II – A Finite Element Computer Program for Incompressible Flow Problems," Sandia National Laboratories Report, SAND86-1816 and SAND86-1817, Albuquerque, New Mexico (1978).
37. FIDAP Manual, Version 5.0, Fluid Dynamics International, Evanston, Illinois (1990).
38. D. K. Gartling, "KACHINA – A Finite Element Computer Program for Incompressible Flow Problems," Sandia National Laboratories Report, SAND2009, Albuquerque, New Mexico (2009).
39. D. H. Pelletier, J. N. Reddy, and J. A. Schetz, "Some Recent Developments and Trends in Finite Element Computational Natural Convection," in *Annual Review of Numerical Fluid Mechanics and Heat Transfer*, Vol. 2, C. L. Tien and T. C. Chawla (eds.), Hemisphere, New York, 39–85 (1989).
40. D. K. Gartling, "Convective Heat Transfer Analysis by the Finite Element Method," *Computer Methods in Applied Mechanics and Engineering*, **12**, 365–382 (1977).
41. G. de Vahl Davis and I. P. Jones, "Natural Convection in a Square Cavity: A Comparison Exercise," *International Journal for Numerical Methods in Fluids*, **3**, 227–248 (1983).
42. T. D. Blacker and M. B. Stephenson, "Paving – A New Approach to Automated Quadrilateral Mesh Generation," Sandia National Laboratories Report, SAND90-0249, Albuquerque, New Mexico (1990).
43. D. K. Gartling, "A Finite Element Analysis of Volumetrically Heated Fluids in an Axisymmetric Enclosure," in *Finite Elements in Fluids*, Vol. 4, R. H. Gallagher, D. Norrie, J. T. Oden, and O. C. Zienkiewicz (eds.), John Wiley & Sons, New York (1982).
44. D.K. Gartling, "A Finite Element Formulation for Incompressible Conjugate Viscous/Porous Flow Problems" in *Proc. International Conference on Computational Methods in Flow Analysis*, Vol. 1, H. Niki and M. Kawahara (eds.), Okayama, Japan, 619–626 (1988).
45. D. K. Gartling, "The Numerical Simulation of Viscous Flow with Change of Phase and Chemical Reaction," in *Proc. International Conference on Computational Engineering Science*, Vol. 2, S. Atluri and G. Yagawa (eds.), Atlanta, Georgia (1988).
46. B. C. Bell and K. S. Surana, " p -Version least squares finite element formulation for two-dimensional non-isothermal incompressible non-Newtonian flow," *International Journal for Numerical Methods in Fluids*, **18**, 127–162 (1994).
47. L. Q. Tang and T. H. Tsang, "Temporal, Spatial and Thermal Features of 3-D Rayleigh-Benard Convection by a Least-Squares Finite Element Method," *Computer Methods in Applied Mechanics and Engineering*, **140** 201 (1997).

48. K. S. Surana, P. Gupta, P. W. TenPas, and J. N. Reddy, “hpk Least Squares Finite Element Processes for 1-D Helmholtz Equation,” *International Journal of Computational Engineering Science and Mechanics*, **7**, 263–291 (2006).
49. J. P. Pontaza and J. N. Reddy, “Spectral/hp Least-Squares Finite Element Formulation for the Navier–Stokes Equations,” *Journal of Computational Physics*, **190**(2), 523–549 (2003).
50. V. Prabhakar and J. N. Reddy, “Spectral/hp Penalty Least-Squares Finite Element Formulation for the Steady Incompressible Navier–Stokes Equations,” *Journal of Computational Physics*, **215**(1), 274–297 (2006).

6

Non-Newtonian Fluids

6.1 Introduction

In Chapters 4 and 5 we studied the finite element models of *Newtonian fluids* (i.e., fluids whose constitutive behavior is linear). Fluids that are not described by the Newtonian constitutive relations are commonly encountered in a wide variety of industrial processes. For example, such materials include motor oils, high molecular weight liquids such as polymers, slurries, pastes, and other complex mixtures. The processing and transport of such fluids are central problems in the chemical, food, plastics, petroleum, and polymer industries.

Non-Newtonian behavior manifests itself in a number of different ways. Most such fluids exhibit a shear rate dependent viscosity, with “shear thinning” (i.e., decreasing viscosity with increasing shear rate) being the most prevalent behavior. Other phenomena associated with the elasticity and memory of the fluid, such as recoil, are also observed in many situations. Differences in the normal stress components occur in many flows and lead to such well-known effects as rod climbing or the Weissenberg effect, and the curvature of the free surface in an open channel flow. A comprehensive list and discussion of these and other non-Newtonian effects is given in the book by Bird et al. [1].

For the present discussion non-Newtonian fluids can conveniently be separated into two distinct categories: (1) inelastic fluids or fluids without memory, and (2) viscoelastic fluids, in which memory effects are significant. The distinction is an important one from both a physical and computational point of view. Basically, inelastic fluids can be viewed as generalizations (in some sense) of the Newtonian fluid. The viscosity function for such materials depends on the rate of deformation of the fluid and thus allows “shear thinning” effects to be modeled. For numerical computations, inelastic fluids can be treated using minor extensions to the standard finite element models developed for Newtonian fluids. Viscoelastic fluids, on the other hand, represent a significant departure from the Newtonian limit in terms of both physical behavior and computational complexity. The primary difficulty here is the “memory” of the fluid; the motion of a material element depends not only on the present stress state, but also on the deformation history of the material element. This history dependence leads to very complex constitutive equations and the need for special computational procedures. It is the purpose of the present chapter to study some aspects of the finite element simulation of both inelastic and viscoelastic fluids.

Following this introduction, a brief outline of the continuum equations, boundary conditions, and pertinent constitutive equations for inelastic fluids will be presented, followed by a description of their finite element models. The emphasis will be on the computational treatment of the material nonlinearities inherent in these types of fluids. The second part of the chapter will be concerned with the so-called simple fluid with fading memory. We will again state the standard balance laws in a convenient form and then describe some of the many different constitutive equations that have been found useful for numerical experimentation and computation. Several numerical examples of both inelastic and viscoelastic flows will also be presented.

The approach taken in the following sections will be very pragmatic with regard to constitutive relations for non-Newtonian fluids. No attempt will be made to theoretically justify the models used or describe in detail their positive or negative aspects, as this topic is well outside the scope of the present text. For readers interested in these questions or other topics in rheology, a number of references can be consulted, including the textbooks by Bird et al. [1], Lodge [2], Walters [3] and Tanner [4]. The book by Crochet et al. [5] covers applications of finite difference and finite element methods to non-Newtonian fluids. The text by Pearson [6] treats a variety of theoretical and practical topics regarding the analysis of industrial flow problems.

6.2 Governing Equations of Inelastic Fluids

For completeness and ready reference, the basic equations of motion are summarized once again. The section concludes with a discussion of representative inelastic, non-Newtonian constitutive relations.

6.2.1 Conservation Equations

The equations resulting from the principles of conservation of mass, momentum, and energy for flows of viscous, incompressible, inelastic fluids consist of the continuity equation, the Navier–Stokes equations, and the energy equation. For most flows of interest, the assumptions of incompressibility and laminar flow are easily justified. Utilizing standard index notation, where repeated subscripts imply summation, the governing equations in a rectangular Cartesian coordinate system are summarized here [7].

Conservation of Mass

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.2.1)$$

Conservation of Momentum

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = \frac{\partial \sigma_{ij}}{\partial x_j} + \rho g_i \quad (6.2.2)$$

Conservation of Energy

$$\rho C \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} \right) = - \frac{\partial q_i}{\partial x_i} + Q + \Phi \quad (6.2.3)$$

The constitutive equations for the total stress, σ_{ij} , and heat flux, q_i , are given by

$$\sigma_{ij} = -P\delta_{ij} + 2\mu D_{ij}, \quad \mu = \mu(D_{ij}) \quad (6.2.4)$$

and

$$q_i = -k \frac{\partial T}{\partial x_i} \quad (6.2.5)$$

where Eq. (6.2.4) is written for an isotropic, inelastic, non-Newtonian or generalized Newtonian fluid [1]. Equation (6.2.5) is the well-known Fourier's heat conduction law. The flow kinematics are given by

$$D_{ij} = \frac{1}{2}(L_{ij} + L_{ji}); \quad L_{ij} = \frac{\partial v_i}{\partial x_j} \quad (6.2.6)$$

where D_{ij} and L_{ij} are the rate of deformation and velocity gradient tensors, respectively. The coefficient μ in Eq. (6.2.4) is a scalar viscosity function which, in general, depends on the rate of deformation tensor \mathbf{D} in a manner that will be made clear in a subsequent section. The function Φ in Eq. (6.2.3) represents the viscous dissipation in the flow, and it enters the energy equation as a volume source of thermal energy. The dissipation depends on the velocity gradients and is given by

$$\Phi = 2\mu D_{ij} D_{ij} \quad (6.2.7)$$

The remaining terms in Eqs. (6.2.1)–(6.2.5) are the same as defined in earlier chapters for Newtonian flows.

Though the above set of equations is quite general and describes most practical inelastic flow problems, several simplifications to the equation system are often considered. Many processing operations involving non-Newtonian fluids occur at very low Reynolds numbers, where $Re = \rho V_{ref} D / \mu$ and V_{ref} is a reference velocity and D is a characteristic dimension in the flow problem. For such cases the nonlinear terms in the momentum equation may be neglected. Despite this simplification, the momentum equations are still nonlinear due to the variation of the viscosity function with the rate of deformation tensor. There are many instances when viscous dissipation can also be neglected, in which case the energy equation is simplified. Various other simplifications involving the energy equation are possible, depending on whether buoyancy effects or temperature-dependent material properties are considered. Such possibilities are treated in the same manner as for a Newtonian fluid and thus will not be considered further in this chapter.

6.2.2 Boundary Conditions

Equations (6.2.1)–(6.2.7) represent, in general, a coupled problem that requires boundary conditions on both the fluid motion and the energy transport. The necessary boundary conditions are of the standard type and consist of specified velocities or tractions for the momentum equations and specified temperature or heat flux for the energy equation. Symbolically the boundary conditions for the momentum equation are

$$v_i = f_i^v(s_k, t) \quad \text{on } \Gamma_v \quad (6.2.8a)$$

$$T_i = \sigma_{ij}(s_k, t) n_j(s_k) = f_i^T(s_k, t) \quad \text{on } \Gamma_T \quad (6.2.8b)$$

where s_k are the coordinates along the boundary, t is the time, n_i is the outward unit normal to the boundary, and Γ_f is the total boundary enclosing the fluid domain, Ω_f , with $\Gamma_f = \Gamma_v \cup \Gamma_T$. Note that the conditions written in Eq. (6.2.8) are in component form, i.e., there is a condition on each component of the velocity and stress vectors.

The thermal boundary conditions, for the nonisothermal case, are given by

$$T = f^T(s_k, t) \quad \text{on } \Gamma_T \quad (6.2.9a)$$

$$- \left(k \frac{\partial T}{\partial x_i} \right) n_i = q_i n_i = q_c + q_r + q_a = f_q(s_k, t) \quad \text{on } \Gamma_q \quad (6.2.9b)$$

where Γ_{ht} is the total boundary enclosing the heat transfer region and $\Gamma_{ht} = \Gamma_T \cup \Gamma_q$. Also, q_a indicates a specified flux and q_c and q_r refer to the convective and radiative components given by

$$q_c = h_c(s_k, T, t)(T - T_c) \quad (6.2.10a)$$

$$q_r = h_r(s_k, T, t)(T - T_r) \quad (6.2.10b)$$

where h_c and h_r are the convective and radiative heat transfer coefficients, and T_c and T_r are the reference (or sink) temperatures for convective and radiative heat transfer.

The above boundary conditions apply to most standard situations where the fluid is contained by fixed boundaries or enters/leaves the domain, Ω_f . One other type of boundary (or interface) condition that requires mention concerns conditions along a free surface between two fluids. Many non-Newtonian (and Newtonian) flows involve situations where the fluid (liquid) is extruded, spun, drawn, or flows in a sheet or jet, such that a free surface interface exists in the problem domain. In most cases, one of the fluids is a gas and its motion relative to the other fluid is neglected. The case of an interface between two immiscible liquids can also be included in this formulation (see Section 1.10.1). The free surface problem was considered in detail in Chapter 4 for the Newtonian fluid and does not differ significantly for the non-Newtonian case. When stress or force balances are considered along the interface, the inelastic formulation is complicated by the nonlinear constitutive behavior defined in (6.2.4). Otherwise, the interface formulation and solution methods are the same as described in Section 4.11 and they will not be repeated here.

6.2.3 Constitutive Equations

The form of the constitutive equation for a generalized Newtonian fluid was given above as

$$\sigma_{ij} = -P\delta_{ij} + 2\mu(D_{ij})D_{ij} \quad (6.2.11)$$

where σ_{ij} are the components of the total stress tensor, P is the pressure, δ_{ij} is the Kronecker delta (or components of the unit tensor), and D_{ij} are the components of the rate of deformation tensor. Of interest in the present section are particular forms for the *deviatoric stress* or *extra stress* components defined by

$$\tau_{ij} = 2\mu(D_{ij})D_{ij} \quad (6.2.12)$$

The viscosity for non-Newtonian fluids is found to depend on the rate of deformation tensor (see [1,4,7,8]):

$$\mu = \mu(D_{ij}) = \mu(I_1, I_2, I_3) \quad (6.2.13)$$

where the I_i are the *invariants* of D_{ij} , defined by

$$I_1 = \text{tr}(\mathbf{D}) = \sum_i D_{ii} \quad (6.2.14a)$$

$$I_2 = \frac{1}{2} \text{tr}(\mathbf{D}^2) = \frac{1}{2} \sum_i \sum_j D_{ij} D_{ji} \quad (6.2.14b)$$

$$I_3 = \frac{1}{3} \text{tr}(\mathbf{D}^3) = \frac{1}{3} \sum_i \sum_j \sum_k D_{ij} D_{jk} D_{ki} \quad (6.2.14c)$$

where tr denotes the trace. For an incompressible fluid, $I_1 = \nabla \cdot \mathbf{v} = 0$. Also, there is no theoretical or experimental evidence to suggest that the viscosity depends on I_3 ; thus, the dependence on the third invariant is eliminated. Equation (6.2.13) then reduces to

$$\mu = \mu(D_{ij}) = \mu(I_2) \quad (6.2.15)$$

for a generalized Newtonian fluid. The viscosity can also depend on the thermodynamic state of the fluid, which for incompressible fluids usually implies a dependence only on the temperature.

Though Eq. (6.2.15) gives the general functional form for the viscosity function, experimental observation and a limited theoretical base must be used to provide specific models for non-Newtonian viscosities. A variety of models have been proposed and correlated with experimental data (e.g., [1]). Several of the most useful and popular models are cataloged below.

6.2.3.1 Power-law model

The simplest and most familiar non-Newtonian viscosity model is the power-law model which has the form

$$\mu = K I_2^{(n-1)/2} \quad (6.2.16)$$

where n and K are parameters, which are perhaps functions of temperature, termed the *power law index* and *consistency*, respectively. One of the most common features of many non-Newtonian fluids is the “power law” decrease in the apparent viscosity with increasing shear (deformation) rate as modeled by Eq. (6.2.16). Such fluids, with an index $n < 1$ are termed *shear thinning* or *pseudoplastic*. A few materials are *shear thickening* or *dilatant* and have an index $n > 1$. The Newtonian viscosity function is obtained with $n = 1$. The admissible range of the index is bounded below by zero due to stability considerations.

When considering nonisothermal flows, the following empirical relations for n and K have proved useful

$$n = n_0 + B \left(\frac{T - T_0}{T_0} \right) \quad (6.2.17)$$

$$K = K_0 \exp[-A(T - T_0)/T_0] \quad (6.2.18)$$

where subscript zero indicates a reference condition and A and B are constants for each fluid.

6.2.3.2 Carreau model

A major deficiency in the power-law model is that it fails to predict upper and lower limiting viscosities for extreme values of the deformation rate, I_2 . This problem is alleviated in the multiple parameter Carreau model, which is of the form

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty) \left(1 + [\lambda I_2]^2\right)^{(n-1)/2} \quad (6.2.19)$$

In Eq. (6.2.19), μ_0 and μ_∞ are the initial and infinite shear rate viscosities, respectively, and λ is a time constant. The remaining parameters were defined previously.

To illustrate the differences between the power-law and Carreau models, a plot of $\log \mu$ versus $\log I_2$ is shown in Figure 6.2.1 for several examples of each model. Like the power-law model, the Carreau viscosity is seen to have a “power law” region, which is bounded on either end by regions of constant viscosity.

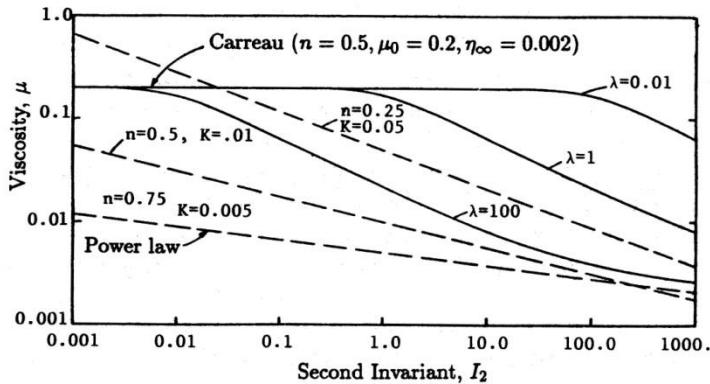


Figure 6.2.1: Viscosity functions for the power-law and Carreau models.

6.2.3.3 Bingham model

The Bingham fluid differs from most other fluids in that it can sustain an applied stress without fluid motion occurring. The fluid possesses a yield stress, τ_0 , such that when the applied stresses are below τ_0 no motion occurs; when the applied stresses exceed τ_0 the material flows, with the viscous stresses being proportional to the excess of the stress over the yield condition. Typically, the constitutive equation after yield is taken to be Newtonian (Bingham model), though other forms such as a power-law equation (Herschel–Buckley model) are possible.

In a general form, the Bingham model is expressed as [9]

$$\tau_{ij} = \left(\frac{\tau_0}{\sqrt{I_2}} + 2\mu \right) D_{ij} \quad \text{when } \frac{1}{2} tr(\tau^2) \geq \tau_0^2 \quad (6.2.20)$$

$$\tau_{ij} = 0 \quad \text{when } \frac{1}{2} tr(\tau^2) < \tau_0^2 \quad (6.2.21)$$

From Eq. (6.2.20) the apparent viscosity of the material beyond the yield point is $(\tau_0/\sqrt{T_2} + 2\mu)$. For a Herschel–Buckley fluid the μ in (6.2.20) is given by Eq. (6.2.16). The inequalities in Eqs. (6.2.20) and (6.2.21) describe a von Mises yield criterion. The implementation of the Bingham model into a computational procedure requires that (6.2.20) and (6.2.21) be modified slightly due to the condition $D_{ij} = 0$. This point will be covered in detail in a later section.

The above models serve to illustrate some of the typical viscosity functions that are available to model inelastic, non-Newtonian fluids. A more extensive list is available in [1].

6.3 Finite Element Models of Inelastic Fluids

6.3.1 Introduction

The finite element formulation of non-Newtonian flows follows very closely the formulations developed for Newtonian flow problems. Therefore, in the present section only a brief overview of the general formulation will be given, with more attention focused on those aspects that are unique to the non-Newtonian problem. For a detailed description of the finite element models of viscous incompressible flow problems, the reader is referred to Chapter 4.

As in the case of Newtonian fluids, there are several different formulations that may be used to construct the finite element models of a non-Newtonian fluid. The equations given in Section 6.2.1 are in terms of the velocity, pressure, and temperature as the dependent variables. However, equations using the stream function and vorticity or the stream function alone could also be employed. The primary argument in favor of the use of primitive variables (i.e., velocity, pressure, and temperature) for non-Newtonian flows comes from the free surface problem. The free surface boundary conditions given previously in Section 4.11 contain the pressure explicitly, and therefore can be more conveniently imposed in a primitive variable formulation. Also, as stated earlier, the use of the stream function-vorticity or stream function formulations presents difficulties in the imposition of boundary conditions, and the stream function formulation requires higher-order interpolation (i.e., C^1 -continuity).

Here, we develop finite element models based on the primitive variables. The mixed and penalty models described in Chapter 4 are presented here for the nonisothermal flow of inelastic fluids.

6.3.2 Mixed Model

Weak forms of Eqs. (6.2.1)–(6.2.3) can be developed from their Galerkin integrals, as explained in Chapters 4 and 5 (see Sections 4.2 and 5.3). The velocity components are approximated by Lagrange interpolation functions, one order higher than those used for the pressure. Suppose that the dependent variables (v_i , P , T) are approximated by expansions of the form

$$T(\mathbf{x}, t) = \sum_{n=1}^N \psi_n(\mathbf{x}) T_n(t) = \boldsymbol{\Psi}^T \mathbf{T} \quad (6.3.1a)$$

$$v_i(\mathbf{x}, t) = \sum_{m=1}^M \psi_m(\mathbf{x}) v_i^m(t) = \boldsymbol{\Psi}^T \mathbf{v}_i \quad (6.3.1b)$$

$$P(\mathbf{x}, t) = \sum_{l=1}^L \phi_l(\mathbf{x}) P_l(t) = \boldsymbol{\Phi}^T \mathbf{P} \quad (6.3.1c)$$

where $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ are vectors of interpolation (or shape) functions, \mathbf{T} , \mathbf{v}_i , and \mathbf{P} are (column) vectors of nodal values of temperature, velocity components, and pressure, respectively, and superscript $(\cdot)^T$ denotes a transpose of the enclosed vector or matrix. Note that the standard practice of interpolating the temperature and velocity variables with the same shape functions has been employed here. Substitution of Eqs. (6.3.1a)–(6.3.1c) into the weak forms of Eqs. (6.2.1)–(6.2.4) [i.e., Eq. (5.3.1)] results in the following set of nonlinear algebraic equations [see Eqs. (5.3.4)–(5.3.6) for details].

Continuity

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (6.3.2)$$

Momentum

$$\mathbf{M} \dot{\mathbf{v}} + \mathbf{C}(\mathbf{v}) \mathbf{v} + \mathbf{K}(\mathbf{v}, \mathbf{T}) \mathbf{v} - \mathbf{Q} \mathbf{P} + \mathbf{B} \mathbf{T} = \mathbf{F} \quad (6.3.3)$$

Energy

$$\mathbf{N} \dot{\mathbf{T}} + \mathbf{D} \mathbf{T} + \mathbf{L} \mathbf{T} = \mathbf{G} \quad (6.3.4)$$

where the superposed dot represents a time derivative and $\mathbf{v}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T\}$. This set of equations is virtually identical to those used in Newtonian problems except for the dependence of the viscous diffusion term \mathbf{K} on the velocity (because of viscosity's dependence on the rate of deformation tensor), and possibly temperature.

For the three-dimensional case, Eqs. (6.3.2)–(6.3.4) have the following explicit form [the continuity equation (6.3.2) and momentum equations (6.3.3) are combined into one]:

$$\begin{aligned} & \left[\begin{array}{cccc} \mathbf{M} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array} \right] \left\{ \begin{array}{c} \dot{\mathbf{v}}_1 \\ \dot{\mathbf{v}}_2 \\ \dot{\mathbf{v}}_3 \\ \dot{\mathbf{P}} \end{array} \right\} + \left[\begin{array}{cccc} \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}(\mathbf{v}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array} \right] \left\{ \begin{array}{c} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P} \end{array} \right\} \\ & + \left[\begin{array}{cccc} \hat{\mathbf{K}}_{11} & \mathbf{K}_{21} & \mathbf{K}_{31} & -\mathbf{Q}_1 \\ \mathbf{K}_{12} & \hat{\mathbf{K}}_{22} & \mathbf{K}_{32} & -\mathbf{Q}_2 \\ \mathbf{K}_{13} & \mathbf{K}_{23} & \hat{\mathbf{K}}_{33} & -\mathbf{Q}_3 \\ -\mathbf{Q}_1^T & -\mathbf{Q}_2^T & -\mathbf{Q}_3^T & \mathbf{0} \end{array} \right] \left\{ \begin{array}{c} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P} \end{array} \right\} = \left\{ \begin{array}{c} \mathbf{F}_1(\mathbf{T}) \\ \mathbf{F}_2(\mathbf{T}) \\ \mathbf{F}_3(\mathbf{T}) \\ \mathbf{0} \end{array} \right\} \end{aligned} \quad (6.3.5)$$

$$[\mathbf{N}] \{\dot{\mathbf{T}}\} + [\mathbf{D}(\mathbf{v})] \{\mathbf{T}\} + [\mathbf{L}] \{\mathbf{T}\} = \{\mathbf{G}(\mathbf{T})\} \quad (6.3.6)$$

The coefficient matrices shown in Eqs. (6.3.5) and (6.3.6) are defined by [from Eq. (6.2.12)]

$$\begin{aligned} \hat{\mathbf{K}}_{11} &= 2\mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} \\ \hat{\mathbf{K}}_{22} &= \mathbf{K}_{11} + 2\mathbf{K}_{22} + \mathbf{K}_{33} \\ \hat{\mathbf{K}}_{33} &= \mathbf{K}_{11} + \mathbf{K}_{22} + 2\mathbf{K}_{33} \end{aligned} \quad (6.3.7a)$$

$$\begin{aligned}
\mathbf{M} &= \int_{\Omega^e} \rho_0 \Psi \Psi^T d\mathbf{x}; \quad \mathbf{C}(\mathbf{v}) = \int_{\Omega^e} \rho_0 \Psi (\Psi^T \mathbf{v}_j) \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \\
\mathbf{K}_{ij} &= \int_{\Omega^e} \mu \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x}; \quad \mathbf{Q}_i = \int_{\Omega^e} \frac{\partial \Psi}{\partial x_i} \Phi^T d\mathbf{x} \\
\mathbf{F}_i(\mathbf{T}) &= - \int_{\Omega^e} \rho_0 g_i \beta \Psi \Phi^T d\mathbf{x} + \int_{\Omega^e} \rho_0 g_i \beta T_0 \Psi d\mathbf{x} + \oint_{\Gamma^e} \Psi T_i ds \\
\mathbf{D}(\mathbf{v}) &= \int_{\Omega^e} \rho_0 C \Phi (\Psi^T \mathbf{v}_j) \frac{\partial \Phi^T}{\partial x_j} d\mathbf{x} \\
\mathbf{N} &= \int_{\Omega^e} \rho_0 C \Psi \Psi^T d\mathbf{x}; \quad \mathbf{L} = \int_{\Omega^e} k \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_i} d\mathbf{x} \\
\mathbf{G} &= \int_{\Omega^e} \Psi Q d\mathbf{x} + \int_{\Omega^e} 2\mu \Psi \Phi d\mathbf{x} + \oint_{\Gamma^e} \Psi q_n ds
\end{aligned} \tag{6.3.7b}$$

where summation on repeated indices is implied. Note that the finite element model is nonlinear because of the nonlinearity in the convective terms as well as the viscosity. In addition, the conductivity k can be a function of temperature. Equations (6.3.2)–(6.3.4) can be combined into a single matrix equation

$$\begin{aligned}
\begin{bmatrix} \mathbf{M} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{N} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}} \\ \dot{\mathbf{P}} \\ \dot{\mathbf{T}} \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{v}) + \mathbf{K}(\mathbf{v}, \mathbf{T}) & -\mathbf{Q} & \mathbf{B}(\mathbf{T}) \\ -\mathbf{Q}^T & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}(\mathbf{v}) + \mathbf{L}(\mathbf{T}) \end{bmatrix} \begin{Bmatrix} \mathbf{v} \\ \mathbf{P} \\ \mathbf{T} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}(\mathbf{T}) \\ \mathbf{0} \\ \mathbf{G}(\mathbf{T}, \mathbf{v}) \end{Bmatrix}
\end{aligned} \tag{6.3.8}$$

or in a more symbolic format as

$$\bar{\mathbf{M}} \dot{\mathbf{U}} + \bar{\mathbf{K}}(\mathbf{v}, \mathbf{T}) \mathbf{U} = \bar{\mathbf{F}} \tag{6.3.9}$$

where

$$\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{P}^T, \mathbf{T}^T\} \tag{6.3.10}$$

This completes the development of the mixed finite element model for the inelastic case.

6.3.3 Penalty Model

In the penalty function method, the continuity equation is treated as a constraint (see Section 5.4) and the problem is reformulated as an unconstrained problem. The pressure, which is the Lagrange multiplier, does not appear explicitly as a dependent variable in the formulation, although it is a part of the boundary stresses [see Eq. (4.3.5)]. In two dimensions, an approximation for the pressure can be post-computed from the relation

$$P = -\gamma_e \left(\frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} \right) \tag{6.3.11}$$

where γ_e is the penalty parameter. The penalty finite element model is given by

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{v}}_1 \\ \dot{\mathbf{v}}_2 \end{Bmatrix} + \begin{bmatrix} \mathbf{C}(\mathbf{v}) & \mathbf{0} \\ \mathbf{0} & \mathbf{C}(\mathbf{v}) \end{bmatrix} \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{Bmatrix} +$$

$$\left(\begin{bmatrix} 2\mathbf{K}_{11} + \mathbf{K}_{22} & \mathbf{K}_{21} \\ \mathbf{K}_{12} & \mathbf{K}_{11} + 2\mathbf{K}_{22} \end{bmatrix} + \begin{bmatrix} \hat{\mathbf{K}}_{11} & \hat{\mathbf{K}}_{21} \\ \hat{\mathbf{K}}_{12} & \hat{\mathbf{K}}_{22} \end{bmatrix} \right) \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \end{Bmatrix} \quad (6.3.12)$$

where \mathbf{M} , $\mathbf{C}(\mathbf{v})$, \mathbf{K}_{ij} (which depend on the viscosity) and \mathbf{F}_i are the same as those defined in Eq. (6.3.7), and

$$\hat{\mathbf{K}}_{ij} = \int_{\Omega^e} \gamma_e \frac{\partial \Psi}{\partial x_i} \frac{\partial \Psi^T}{\partial x_j} d\mathbf{x} \quad (6.3.13)$$

The energy equation remains unchanged as in Eq. (6.3.6). In matrix form, Eqs. (6.3.13) and (6.3.6) can be expressed as

$$\bar{\mathbf{M}} \dot{\mathbf{U}} + \bar{\mathbf{K}}(\mathbf{v}, \mathbf{T}) \mathbf{U} = \bar{\mathbf{F}} \quad (6.3.14)$$

where

$$\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, \mathbf{T}^T\} \quad (6.3.15)$$

This completes the discussion of the penalty finite element model for the inelastic case.

6.3.4 Matrix Evaluations

The appearance of a nonlinear diffusion term, \mathbf{K} , in Eqs. (6.3.9) and (6.3.14) influences two basic aspects of the finite element analysis, namely, the matrix evaluation and solution algorithms. Matrix evaluation methods will be discussed first.

The techniques used for the construction of element level matrices that contain variable coefficients have generally fallen into one of two categories — the reconstruction methods and the hypermatrix methods. The differences in these methods and their application to non-Newtonian formulations are best explained via a specific example. Consider the construction of a particular component of the \mathbf{K} matrix in Eq. (6.3.9), for example,

$$\mathbf{K}_{11} = \mathbf{K}_{\mathbf{xx}} = \int_{\Omega^e} \frac{\partial \Psi}{\partial x} \mu(I_2, T) \frac{\partial \Psi^T}{\partial x} d\mathbf{x} \quad (6.3.16)$$

where Ψ is the vector of shape functions used for the velocity field. The viscosity function is shown in its most general form with a dependence on I_2 and the temperature; the invariant I_2 depends on velocity gradients [see Eq. (6.2.14)].

For most finite element applications the integration in Eqs. (6.3.7) and (6.3.16) is performed via a numerical quadrature. Such a computation requires that each function in the integrand be evaluated at each integration point within the element. Therefore, the evaluation of \mathbf{K} for a non-Newtonian fluid requires that the velocity gradients and perhaps temperature be evaluated at integration points, using the values from the latest available solution (\bar{u}_m, \bar{T}_n) at time t_s :

$$T(\mathbf{x}, t_s) = \sum_{n=1}^N \theta_n(\mathbf{x}) \bar{T}_n(t_s) \quad (6.3.17a)$$

$$v_i(\mathbf{x}, t_s) = \sum_{m=1}^M \psi_m(\mathbf{x}) \bar{v}_i^m(t_s) \quad (6.3.17b)$$

In many standard finite element programs the above quadrature procedure is carried out each time the element matrix is required. As a result of the iterative methods used to solve the nonlinear equations, the matrix formation operations may be required many times, especially for time-dependent problem, resulting in a large computational cost.

An alternative to the matrix reconstruction method, the hypermatrix approach may be used. This method reduces some of the computational cost but at the cost of some additional storage and I/O in the computer program. Since the viscosity is a function of position in the element (due to its functional dependence on the velocity field and temperature), it is natural to interpolate the viscosity in the same way as a dependent variable. Thus, let the viscosity be represented by

$$\mu = \Psi^T \hat{\mu} \quad (6.3.18)$$

where Ψ is a vector of shape functions and $\hat{\mu}$ is a vector of nodal point viscosity values. Substitution of Eq. (6.3.18) in Eq. (6.3.16) yields

$$\mathbf{K}_{\mathbf{xx}} = \int_{\Omega_e} \frac{\partial \Psi}{\partial x} (\Psi^T \hat{\mu}) \frac{\partial \Psi^T}{\partial x} dx \quad (6.3.19)$$

Since the $\hat{\mu}$ are the nodal values, the integral in Eq. (6.3.19) can be constructed once and stored as a three-dimensional array $\mathbf{K}(NI, NJ, NK)$, where each index corresponds to one of the shape functions in (6.3.19). This matrix is called the *hypermatrix*. A product of the hypermatrix with a known vector of nodal point viscosities produces the required element matrix without repeated numerical quadrature. This technique has been used successfully (see [10,11]) for the nonlinear advection terms in the momentum equations [i.e., the $\mathbf{C}(\mathbf{v})\mathbf{v}$ term in Eq. (6.3.7)] as well as for material property variations (see [12]). It is most effective when the variable coefficient is a nodal quantity or depends on a nodal quantity, since this permits shape function evaluations to be avoided. Unfortunately, this is not the case for viscosity which depends on velocity gradients. It is well-known that the most accurate points within a quadrilateral element at which to evaluate derivatives are the 2×2 Gauss integration points (see [13]). Thus, for maximum accuracy in the viscosity evaluation the invariant I_2 should be evaluated at the Gauss points. However, the extrapolation of Gauss point values of I_2 to the nodes by standard methods (see [14,15]), in order to use Eq. (6.3.19), has not proven to be a viable technique for most non-Newtonian models. In general, predictions of I_2 at the nodes via extrapolation are very inaccurate and lead to poor overall accuracy of the solution method. As an alternative to extrapolation, an averaging method is used in which the Gauss point values of I_2 are averaged over the element and a single value is used to evaluate the viscosity at the nodes.

Figure 6.3.1 shows velocity profiles obtained with both the extrapolation and averaging procedures described above. The results are for a power-law fluid in a cylindrical tube. The inaccuracies generated by extrapolation are on the order of 30% while the averaging procedure yields results within a few percent of the analytical result. The averaging procedure in conjunction with Eq. (6.3.19) has been used successfully in other problems with a variety of viscosity models (see [16,17]).

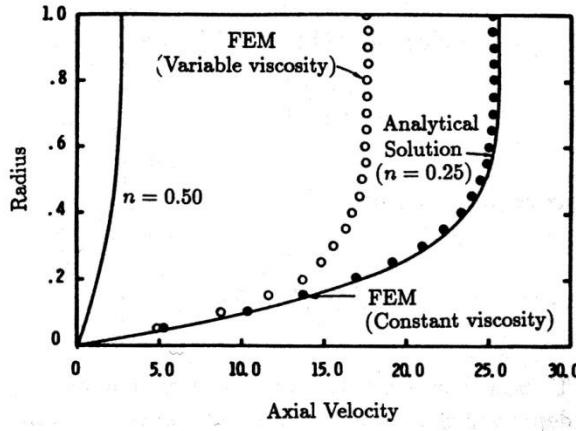


Figure 6.3.1: Velocity profile for a power fluid in a tube; various methods for viscosity computation.

To conclude this section it should be pointed out that the choice of a reconstruction or hypermatrix procedure is dependent on the type of computing resources available and the structure of the finite element software. Both methods are equivalent and effective. The relative costs of CPU and I/O time will heavily influence the selection.

6.4 Solution Methods for Inelastic Fluids

The solution procedure selected for the inelastic, non-Newtonian problem given by the finite element models in Section 6.3 must be capable of treating several different types of nonlinearities. For nonzero Reynolds number flows the nonlinear advection terms are significant and in many cases may dominate the problem. The non-Newtonian fluid introduces a second type of nonlinear behavior while the presence of unknown free surface boundaries can introduce a geometric nonlinearity. In the present section, only the first two types of phenomena will be considered; free surface solutions were covered in a previous chapter. Also, the discussion will focus on steady isothermal problems since many of the procedures and results obtained for this type of problem can be extended in a straightforward manner to more complex situations. The solution methods described in Chapters 4 and 5 are also applicable here with only minor alterations.

For the steady-state case, the mixed method Eqs. (6.3.2) and (6.3.3) take the form

$$\mathbf{C}(\mathbf{v})\mathbf{v} + \mathbf{K}(\mathbf{v}) - \mathbf{Q}\mathbf{P} = \mathbf{F} \quad (6.4.1a)$$

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (6.4.1b)$$

or, as a single matrix equation we have

$$\hat{\mathbf{K}}(\mathbf{U})\mathbf{U} = \hat{\mathbf{F}} \quad (6.4.2)$$

where the vector \mathbf{U} now contains the velocity components (v_1, v_2, v_3) and the pressure \mathbf{P} . The dependence of μ on \mathbf{v} has been absorbed into \mathbf{K} and hence into $\hat{\mathbf{K}}$.

There are a wide variety of iterative algorithms that can be applied to Eq. (6.4.2), and we will consider only two such schemes in detail here. Similar comments apply to the penalty method model from Eq. (6.3.14).

The simplest method is Picard's method (also known as successive substitution or functional iteration), which can be written in the following form

$$\hat{\mathbf{K}}(\mathbf{U}^n)\mathbf{U}^{n+1} = \hat{\mathbf{F}}^n \quad (6.4.3)$$

where the superscript n indicates the iteration level. Equation (6.4.3) describes a particularly simple procedure in which the nonlinear coefficients in the problem are evaluated using dependent variable data from the previous iteration. The method has a linear rate of convergence but works for a relatively large range of problems. For shear thinning materials it is observed that as the power law index, n [see Eq. (6.2.16) for example] decreases, the rate of convergence decreases markedly (see [5]).

A more sophisticated algorithm with a higher rate of convergence is Newton's method. For the nonlinear equation in (6.4.2) Newton's method may be written as

$$\mathbf{J}(\mathbf{U}^n)[\mathbf{U}^{n+1} - \mathbf{U}^n] = -\hat{\mathbf{K}}(\mathbf{U}^n)\mathbf{U}^n + \hat{\mathbf{F}}^n \quad (6.4.4)$$

where \mathbf{J} is the Jacobian matrix defined by

$$\mathbf{J}(\mathbf{U}^n) = \frac{\partial}{\partial \mathbf{U}} [\hat{\mathbf{K}}(\mathbf{U})\mathbf{U} - \hat{\mathbf{F}}] \Big|_{\mathbf{U}^n} \quad (6.4.5)$$

Newton's method is the standard solution procedure for Newtonian problems since it is well-suited to the quadratic nonlinearity occurring in the advection terms of the momentum equations. However, experience has shown that this procedure, as written in Eq. (6.4.4), does not perform well for many types of generalized Newtonian fluids (see [18]). In particular, Newton's method does not work well for viscosity models with shear thinning. It is therefore recommended that the non-Newtonian behavior in Eq. (6.4.4) be treated using the Picard method; the advection terms, if present, should be treated using Newton's method.

To illustrate the behavior of the Picard and Newton algorithms, a simple creeping flow problem ($Re = 0$) was solved using a power-law viscosity model. The problem consists of the ubiquitous driven cavity flow in which a fluid is contained in a square cavity three sides of which are stationary; the fourth side of the cavity moves at unit velocity in its own plane. Shown in Figure 6.4.1 are plots of the convergence measure (relative norm on the change in the solution between iterations) versus iteration number for both algorithms and several values of the power-law index. The Picard scheme converges for all values of the index below unity (shear thinning) but diverges for shear thickening fluids. The Newton method performs in the opposite way, with rapid convergence for shear thickening and divergence for smaller values of the index.

Other methods of solution of Eq. (6.4.2) are possible though the Picard and Newton methods represent the behavioral limits for most algorithms. Tanner and Milthorpe [18] have used a combination of the Picard and the Newton method successfully for several nonlinear viscosity models. Also, Engelman [19] has found

the quasi-Newton or variable metric method to be very cost effective for non-Newtonian flows.

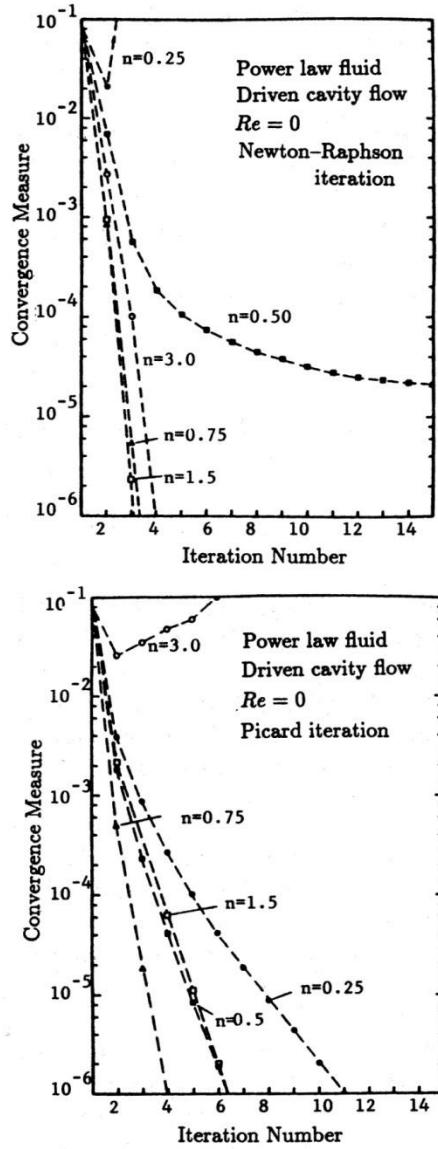


Figure 6.4.1: Iterative convergence for the Picard and Newton methods.

The solution procedures outlined above are applicable to all of the common inelastic, non-Newtonian models. However, the Bingham model [Eqs. (6.2.20) and (6.2.21)] deserves some further comments due to its unique method of implementation. For a typical velocity-based finite element method, the exact Bingham constitutive equation is quite difficult to use, mainly as a result of the requirement that no motion take place below the yield condition. To circumvent this difficulty it has proved useful to employ an approximate Bingham equation

given by

$$\tau_{ij} = \left(\frac{\tau_0(1 - \mu/\mu_r)}{\sqrt{I_2}} + 2\mu \right) D_{ij} \quad \text{when } \frac{1}{2} \operatorname{tr}(\tau^2) \geq \tau_0^2 \quad (6.4.6a)$$

$$\tau_{ij} = 2\mu_r D_{ij} \quad \text{when } \frac{1}{2} \operatorname{tr}(\tau^2) < \tau_0^2 \quad (6.4.6b)$$

where μ_r is a pre-yield viscosity and $\mu/\mu_r \ll 1$. A plot of the constitutive model given by (6.4.6) is shown in Figure 6.4.2. Unlike the true Bingham material, the fluid described by Eq. (6.4.6) can undergo deformation below the yield point though the magnitude of the motion can be made arbitrarily small (i.e., approach the Bingham model) by increasing μ_r relative to μ . Experience has shown that when $\mu/\mu_r \sim .01$ or less then the solution is virtually independent of μ_r and thus approximates the Bingham model with excellent accuracy [17,18].

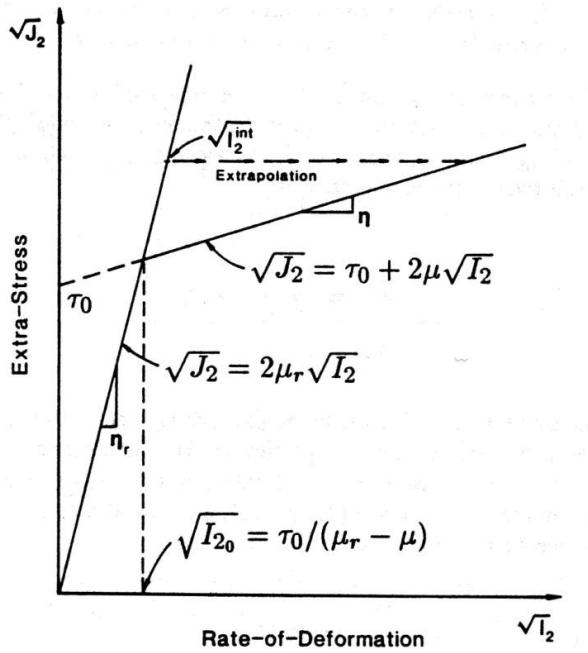


Figure 6.4.2: Bi-viscosity model for computations involving Bingham fluids.

Iterative convergence of Bingham fluids using the Picard method [see Eq. (6.4.3)] can be very slow since the apparent viscosity after yield is essentially like that of a power-law fluid with index $n = 0$. To accelerate the convergence a particular form of extrapolation can be used. Note first that for the present model the yield stress, τ_0 , corresponds to a value of the rate of deformation (invariant) given by $\sqrt{I_{20}} = \tau_0/(\mu_r - \mu)$ as seen in Figure 6.4.2. At the start of the iterative procedure the fluid is usually assumed to have a viscosity, μ_r , from which an initial velocity field and values of I_2 may be computed. For those elements which have yielded ($\sqrt{I_2} \geq \sqrt{I_{20}}$) an extrapolation at constant stress should be used to estimate a new value of the apparent viscosity. Failure to perform this extrapolation can double or

triple the number of iterations needed to reach convergence. Based on Figure 6.4.2 (dashed arrow) it is possible to provide an analytical expression for the projected value of I_2 needed for the viscosity evaluation. That is,

$$\sqrt{I_2} = \frac{2\mu_r \sqrt{I_2^{int}} - \tau_0}{2\mu} \quad (6.4.7)$$

where the superscript *int* indicates the value of I_2 computed initially using a viscosity of μ_r . The above procedure has been used in a variety of steady-state problems with good success [16,17] and is readily extended to time-dependent flows.

6.5 Governing Equations of Viscoelastic Fluids

As noted in the introduction, non-Newtonian behavior has many facets. In the previous sections we dealt with the modeling of “shear rate” dependent viscosity behavior but ignored the problems associated with fluid elasticity and memory. In the following several sections we will focus on viscoelastic fluids and their numerical simulation by the finite element method.

6.5.1 Conservation Equations

The analytical description of the motion of a continuous medium is based on conservation of mass, momentum, and energy, and the associated equations of state and constitutive relations. The present development will be limited to laminar flows of incompressible, viscous isotropic fluids. The fluid motion is assumed to be isothermal to avoid the discussion of the energy equation, as it was discussed in detail in Chapter 5. We also assume that, for simplicity, the flows are two-dimensional. Of course, the extension of the present discussion to three dimensions is conceptually straightforward, although in practice the subject is sufficiently complex and computationally taxing to have received relatively little attention. The discussion will also be limited to simple fluids with fading memory [4–7,20,21], a material description that has received the most widespread attention and development.

The equations of interest for two-dimensional flows are written in the Cartesian coordinate system (x_1, x_2) using the Eulerian description.

Conservation of Mass

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.5.1)$$

Conservation of Momentum

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = \frac{\partial \sigma_{ij}}{\partial x_j} + \rho g_i \quad (6.5.2)$$

where v_i denotes the i th component of velocity, σ_{ij} the components of the total (Cauchy) stress tensor, ρ the density, g_i the component of body force, and t denotes time. The standard index notation is used, with summation on repeated indices.

The field equations in (6.5.1) and (6.5.2) are to be solved in conjunction with a constitutive equation for the fluid and boundary and initial conditions of the flow

problem. The various constitutive equations of viscoelastic fluid behavior are a major focus of this chapter, and will be discussed in the next section. Since the boundary conditions depend on the dependent variables employed in the problem, they are discussed subsequent to the discussion of the constitutive relations.

6.5.2 Constitutive Equations

For an incompressible fluid the total stress is given by

$$\sigma_{ij} = -P\delta_{ij} + \tau_{ij} \quad (6.5.3)$$

where P is a pressure and τ_{ij} is the extra-stress. It should be noted that due to the particular forms that will be used to describe the extra-stress, the pressure defined in Eq. (6.5.3) will not, in general, be the same as a Newtonian pressure or hydrostatic pressure. That is, the extra-stress may not be traceless (i.e., $\tau_{ii} \neq 0$), in which case P is not the usual mean normal stress found in Newtonian fluid mechanics [22,23].

The central issue for the constitutive description of a viscoelastic fluid is the choice of an equation that relates the extra-stress in (6.5.3) to the flow kinematics. For the general class of materials called simple fluids, such a relationship can be established in functional form where the current extra-stress is related to the history of deformation in the fluid. Typically, such a relation is expressed as

$$\tau_{ij} = \mathcal{F}_{ij}[G_{kl}(s), 0 < s < \infty] \quad (6.5.4)$$

where \mathcal{F}_{ij} is a tensor-valued functional, G_{kl} is a finite deformation tensor (related to the Cauchy–Green tensor), and $s = t - t'$ is the time lapse from time t' to the present time, t . Unfortunately, the generality of a functional form, such as (6.5.4), does not produce usable constitutive equations for general flow problems; the solution of practical problems requires that approximate forms of (6.5.4) be developed.

Since there are numerous ways to approximate the general functional given in (6.5.4), a wide variety of constitutive equations have developed out of the simple fluid theory. To date, none of these approximate constitutive equations are universally applicable; that is, no single constitutive equation is capable of predicting all of the observed behavior in elastic fluids. In essence, the present state-of-the-art mandates that a spectrum of constitutive equations be considered for use, with the specific choice being dictated by the ability of a given model to predict the dominant, non-Newtonian effects expected in a particular application.

To further complicate the situation, approximate constitutive equations can be of several different forms. For certain limiting flow conditions, such as “small” strains, “slow” flows, or “slightly” elastic fluids, the functional in (6.5.4) can be approximated by an expansion in terms of a small parameter. Such a formulation leads to a set of “hierarchical” equations for the extra-stress in terms of the rate of deformation tensor and its various derivatives [5,6]. In these models the extra-stress is given explicitly in terms of the flow kinematics, which in principle, allows Eqs. (6.5.1)–(6.5.3) to be written in terms of only the velocity and pressure, as in a Newtonian flow. Though convenient in form, these hierarchical formulas are of very limited value in general flows due to the “smallness” assumptions used in their derivation. Some computational work has been done using these types of models

[24–26]; however, due to their limited applicability they will not be given further consideration here.

The two remaining major categories of approximate constitutive relations include the integral and differential models. The integral model represents the extra-stress in terms of an integral over past time of the fluid deformation history. For a differential model the extra-stress is determined from a differential (evolution) equation that relates the stress and stress rate to the flow kinematics. The choice between a differential and integral formulation is a crucial one with regard to numerical simulation since the computational algorithms for each type are very different. In the present book we will emphasize the differential form since the majority of the current numerical work involves this formulation. References to work with integral models will also be given.

6.5.2.1 Differential models

The differential constitutive equations to be considered here are implicit, rate-type models, generally associated with the names of Oldroyd, Maxwell, and Jeffrey. Before proceeding to the description of specific models it is important to list a number of definitions. For an Eulerian reference frame the material time derivative (or convected derivative) of a symmetric, second-order tensor can be defined in several ways, all of which are frame invariant. Let φ_{ij} denote the Cartesian components of a second-order tensor. Then the *upper-convected* (or co-deformational) derivative is defined by

$$\overset{\nabla}{\varphi}_{ij} = \frac{\partial \varphi_{ij}}{\partial t} + v_k \frac{\partial \varphi_{ij}}{\partial x_k} - L_{ik}\varphi_{kj} - L_{jk}\varphi_{ki} \quad (6.5.5a)$$

and the *lower-convected* derivative is defined as

$$\overset{\Delta}{\varphi}_{ij} = \frac{\partial \varphi_{ij}}{\partial t} + v_k \frac{\partial \varphi_{ij}}{\partial x_k} + L_{ki}\varphi_{kj} + L_{kj}\varphi_{ki} \quad (6.5.5b)$$

where v_k are the components of the velocity vector and L_{ij} are the components of the velocity gradient tensor \mathbf{L} , defined by

$$\mathbf{L} = \nabla \mathbf{v}, \quad \text{or} \quad L_{ij} = \frac{\partial v_j}{\partial x_i} \quad (6.5.6)$$

Since both Eqs. (6.5.5a) and (6.5.5b) are admissible convected derivatives, their linear combination is also admissible:

$$\dot{\varphi}_{ij} = \left(1 - \frac{a}{2}\right) \overset{\nabla}{\varphi}_{ij} + \left(\frac{a}{2}\right) \overset{\Delta}{\varphi}_{ij} \quad (6.5.7)$$

Equation (6.5.7) is a general convected derivative which reduces to (6.5.5a) for $a = 0$ and (6.5.5b) for $a = 2$. When $a = 1$ [average of (6.5.5a) and (6.5.5b)] the convected derivative in (6.5.7) is termed a *corotational* or *Jaumann derivative*.

Using τ as the extra-stress tensor with the Cartesian components denoted by τ_{ij} , the upper-convected derivative of the stress components is given by

$$\overset{\nabla}{\tau}_{ij} = \frac{\partial \tau_{ij}}{\partial t} + v_k \frac{\partial \tau_{ij}}{\partial x_k} - L_{ik}\tau_{kj} - L_{jk}\tau_{ki} \quad (6.5.8a)$$

and the lower-convected derivative is

$$\overset{\Delta}{\tau}_{ij} = \frac{\partial \tau_{ij}}{\partial t} + v_k \frac{\partial \tau_{ij}}{\partial x_k} + L_{ki} \tau_{kj} + L_{kj} \tau_{ki} \quad (6.5.8b)$$

The corotational derivative is

$$\overset{\circ}{\tau}_{ij} = \left(1 - \frac{a}{2}\right) \overset{\nabla}{\tau}_{ij} + \left(\frac{a}{2}\right) \overset{\Delta}{\tau}_{ij} \quad (6.5.9)$$

All of these derivatives have been used in various differential constitutive equations. The selection of one type of derivative over another is usually based on the physical plausibility of the resulting constitutive equation and the matching of experimental data to the model for simple (viscometric) flows.

The simplest differential constitutive models are the upper- and lower-convected Maxwell fluids, which are defined by the following equations.

Upper-Convected Maxwell Fluid

$$\tau_{ij} + \lambda \overset{\nabla}{\tau}_{ij} = 2\mu^p D_{ij} \quad (6.5.10a)$$

Lower-Convected Maxwell Fluid

$$\tau_{ij} + \lambda \overset{\Delta}{\tau}_{ij} = 2\mu^p D_{ij} \quad (6.5.10b)$$

where λ is a relaxation time for the fluid, μ^p is a viscosity, and D_{ij} are the components of the rate of deformation (or strain rate) tensor defined by

$$\mathbf{D} = \frac{1}{2} [\nabla \mathbf{v} + (\nabla \mathbf{v})^T], \text{ or } D_{ij} = \frac{1}{2} (L_{ij} + L_{ji}) \quad (6.5.11)$$

The upper-convected Maxwell model (6.5.10a) has been used extensively in testing numerical algorithms; the lower-convected and corotational forms of the Maxwell fluid predict physically unrealistic behavior and are not generally used.

By employing the general convected derivative (6.5.9) in a Maxwell-like model the following constitutive equation is produced.

Johnson–Segalman Model

$$\tau_{ij} + \lambda \overset{\circ}{\tau}_{ij} = 2\mu^p D_{ij} \quad (6.5.12)$$

which is the Johnson–Segalman fluid [27]. By slightly modifying Eq. (6.5.12) to include a variable coefficient for τ_{ij} , the Phan Thien–Tanner model [28] is derived.

Phan Thien–Tanner Model

$$Y(\tau) \tau_{ij} + \lambda \overset{\circ}{\tau}_{ij} = 2\mu^p D_{ij} \quad (6.5.13a)$$

where

$$Y(\tau) = 1 + \frac{\epsilon \lambda}{\mu^p} \operatorname{tr}(\tau) \quad (6.5.13b)$$

and ϵ is a constant. This equation is somewhat better than (6.5.12) in representing actual material behavior.

The Johnson–Segalman and Phan Thien–Tanner models suffer a common defect in that, for a monotonically increasing shear rate, there is a region where the shear stress decreases — a physically unrealistic behavior. To correct this anomaly the constitutive equations can be altered using the following procedure. Let the extra-stress be decomposed into two partial stresses, τ_{ij}^s and τ_{ij}^p such that

$$\tau_{ij} = \tau_{ij}^s + \tau_{ij}^p \quad (6.5.14)$$

where τ_{ij}^s is a purely viscous and τ_{ij}^p is a viscoelastic stress component. The decomposition used here is sometimes thought of in terms of a combination of a Newtonian solvent (superscript s) and a polymer additive (superscript p). Using the Johnson–Segalman fluid as an example, then

$$\tau_{ij}^s = 2\mu^s D_{ij}$$

$$\tau_{ij}^p + \lambda \overset{\circ}{\tau}_{ij}^p = 2\mu^p D_{ij} \quad (6.5.15)$$

It is possible to eliminate the partial stresses in (6.5.14) and (6.5.15) to produce a new constitutive relation of the following form

$$\tau_{ij} + \lambda \overset{\circ}{\tau}_{ij} = 2\bar{\mu} \left(D_{ij} + \lambda' \overset{\circ}{D}_{ij} \right) \quad (6.5.16)$$

with $\bar{\mu} = (\mu^s + \mu^p)$ and $\lambda' = \lambda\mu^s/\bar{\mu}$, where λ' is a retardation time. The constitutive equation in (6.5.16) is recognized as a type of Oldroyd fluid; it is, in fact, a reduced form of the eight-constant Oldroyd model [1,5]. For particular choices of the convected derivative in Eq. (6.5.16), specific models can be generated. When $a = 0$ ($\overset{\circ}{\tau}_{ij}$ becomes $\overset{\nabla}{\tau}_{ij}$) then (6.5.14) becomes the Oldroyd B fluid; the case $a = 2$ ($\overset{\circ}{\tau}_{ij}$ becomes $\overset{\Delta}{\tau}_{ij}$) produces the Oldroyd A fluid. In order to ensure a monotonically increasing shear stress the inequality $\mu^s \geq \mu^p/8$ must be satisfied. The stress decomposition employed above can also be used with the Phan Thien–Tanner model to produce a correct shear stress behavior.

In all of the above constitutive equations the material parameters, λ and μ^p , were assumed to be constants. For some constitutive equations the constancy of these parameters leads to material (or viscometric) functions that do not accurately represent the behavior of real elastic fluids. For example, the shear viscosity predicted by a Maxwell fluid is a constant, when in fact viscoelastic fluids normally exhibit a shear thinning behavior. This situation can be remedied to some degree by allowing the parameters λ and μ^p to be functions of the invariants of the rate of deformation tensor as was done for the generalized Newtonian fluid (see Section 6.2.3). Using the upper-convected Maxwell fluid as an example, then

$$\tau_{ij} + \lambda(I_2)\overset{\nabla}{\tau}_{ij} = 2\mu^p(I_2)D_{ij} \quad (6.5.17)$$

where I_2 is the second invariant of the strain rate tensor \mathbf{D} [see Eq. (6.5.11)], $I_2 = 1/2(D_{ij}D_{ij})$. The constitutive equation in (6.5.17) is usually termed a White–Metzner model [29]. White–Metzner forms of other differential models, such as the Oldroyd fluids, have also been developed and used in various situations.

The above list of differential models is by no means exhaustive though it does include many of the constitutive equations that have been used in computational work. Though the equations have been stated without derivation or motivation, they all rest on a reasonable theoretical basis. Some of the equations (e.g., Oldroyd and Maxwell) were developed as generalizations of simple, linear viscoelastic models. Linear viscoelasticity can be formally developed from the previously cited “hierarchical” equations or informally from mechanical analogies and heuristic arguments [1]. Other differential models, such as the Johnson–Segalman and Phan Thien–Tanner fluids, were derived using statistical mechanics ideas and a conceptual (network) model for the microstructure of a viscoelastic fluid. Due to their approximate nature, all of the above models are limited in their ability to represent true viscoelastic behavior; the more complex models provide a reasonable qualitative description of many observed phenomena. A catalog of the strengths, weaknesses, and limitations of these and other differential models can be found in [1,4,5,23].

6.5.2.2 Integral models

An approximate integral model for a viscoelastic fluid represents the extra-stress in terms of an integral over the past history of the fluid deformation. A general form for a single integral model can be expressed as (see [30])

$$\tau_{ij} = \int_{-\infty}^t 2m(t-t')H_{ij}(t,t')dt' \quad (6.5.18)$$

where t is the current time, m is a scalar memory function (or relaxation kernel), and H_{ij} is a nonlinear deformation measure (tensor) between the past time, t' , and t .

There are many possible forms for both the memory function and the deformation measure. Normally the memory function is a decreasing function of the time lapse $s = t - t'$. Typical of such a function is the exponential given by

$$m(t-t') = m(s) = \frac{\mu_0}{\lambda^2} e^{-s/\lambda} \quad (6.5.19)$$

where the parameters μ_0 , λ , and s were defined previously. Like the choice of a convected derivative in a differential model, the selection of a deformation measure for use in Eq. (6.5.18) is somewhat arbitrary. One particular form that has received some attention is given by

$$H_{ij} = \phi_1(I, II)C_{ij}^{-1} + \phi_2(I, II)C_{ij} \quad (6.5.20)$$

In Eq. (6.5.20) C_{ij} is the Cauchy–Green deformation tensor, C_{ij}^{-1} is its inverse, called the Finger tensor, and the ϕ_i are scalar functions of the invariants of the deformation tensors, i.e., $I = \text{tr}(C_{ij}^{-1})$ and $II = \text{tr}(C_{ij})$. The indicated deformation tensor is defined by (see Malvern [31])

$$C_{ij} = \frac{\partial \chi_m}{\partial x_i} \frac{\partial \chi_m}{\partial x_j} \quad (6.5.21)$$

where χ_m is the location at time t' of a fluid particle that is now at position x_i and time t . The form of the deformation measure in Eq. (6.5.20) is still quite general, though specific choices for the functions ϕ_i and the memory function m lead to several well-known constitutive models. Among these are the Kaye–BKZ fluid [1,4,23] and the Lodge rubber-like liquid [32].

As a specific example of an integral model of the type given by the previous equations we consider the Maxwell fluid. Setting $\phi_1 = 1$ and $\phi_2 = 0$ in Eq. (6.5.20) and using the memory function shown in (6.5.19) allow a constitutive equation of the following form

$$\tau_{ij} = \frac{\mu_0}{\lambda^2} \int_{-\infty}^t \exp [-(t-t')/\lambda] \left[C_{ij}^{-1}(t') - \delta_{ij} \right] dt' \quad (6.5.22)$$

In writing Eq. (6.5.22), the Finger deformation tensor in (6.5.20) has been replaced with the more usual Finger strain tensor (e.g., [4,30,31]). The above constitutive equation is an integral equivalent to the upper-convected Maxwell model shown in differential form in Eq. (6.5.10a). Note that in this case the extra-stress is given in an explicit form, though its evaluation requires that the strain history be known for each fluid particle. Also, it is important to emphasize that though the Maxwell fluid has both a differential and integral form, this is not generally true for other constitutive equations.

Since the emphasis here is on differential models we will not dwell on other specific forms of integral constitutive equations. A good source for further discussion of such models is the book by Bird et al. [1]. It is appropriate to note in closing this section that some of the integral models are very good at reproducing realistic viscoelastic behavior [4,23]. However, this improvement in modeling accuracy is offset to a large degree by the difficulties in using such models in general computational schemes. Further comments on computational procedures for integral models will be reserved for a later section.

6.5.3 Boundary Conditions

The equations in (6.5.1) and (6.5.2) are to be solved in conjunction with an appropriate constitutive relation as discussed in the previous section. The set of boundary conditions for the problem at hand consists of either specified velocities or tractions on the boundary of the fluid domain, with one condition needed for each component of velocity. However, there is also an additional set of conditions for the viscoelastic problem due to the “memory” of the fluid. The extra condition, in reality, is an initial condition on the components of the fluid extra-stress, though it is often implemented as a boundary condition, especially for flow-through type boundaries. For time-dependent problems the initial fluid stresses must be given at all points in the problem domain. All problems that contain an inflow boundary require the specification of a strain (stress) history of the fluid crossing the boundary. Completely confined flows do not require any additional data beyond the initial stress-state. Additional comments on boundary conditions can be found in [33].

6.6 Finite Element Model of Differential Form

6.6.1 Preliminary Comments

In this section we will outline typical finite element procedures for the solution of viscoelastic flow problems. At the outset it is important to realize that a standard, well-established computational procedure has not yet evolved for this particular class of flow problems. Numerous different formulations have been proposed [21], but none has proved adequate for the solution of flows with highly elastic fluids. Further comments on the unresolved general problems in this area will be given in a later section.

The wide variety of possible constitutive equations and finite element formulations makes detailed explanation of many specific algorithms impossible in this limited text. Therefore, the detailed discussion will be focused on a single, mixed finite element formulation for a particular differential constitutive equation. Other models and formulations follow easily from this basic outline. A brief description of typical procedures for integral models will also be presented.

6.6.2 Summary of Governing Equations

The governing equations of interest are summarized below in rectangular Cartesian form.

Conservation of Mass

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.6.1)$$

Conservation of Momentum

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = \frac{\partial \sigma_{ij}}{\partial x_j} \quad (6.6.2)$$

Constitutive Equations

$$\sigma_{ij} = -P\delta_{ij} + \tau_{ij} \quad (6.6.3)$$

$$\tau_{ij} = \tau_{ij}^s + \tau_{ij}^p \quad (6.6.4)$$

$$\tau_{ij}^s = 2\mu^e D_{ij} \quad (6.6.5)$$

$$\tau_{ij}^p + \lambda \nabla_p^2 v_i = 2\mu^p D_{ij} \quad (6.6.6)$$

where in Eq. (6.6.2) the body force term has been neglected for simplicity. The constitutive equation employed is the Oldroyd B model. Other models could be included by adding a generic function of the stress to the right-hand side of the constitutive equation. However, the basic algorithm remains unchanged. Also, note that the solvent or viscous stress in (6.6.5) has been expressed in terms of an effective viscosity. In many cases, this is taken as the solvent viscosity ($\mu^e = \mu^s$) as illustrated in equation (6.5.15). In other situations, the stress split is made arbitrary and μ^e represents only a part of the total viscosity. The permissible variation in this definition has been used to advantage in developing solution algorithms.

The above set of equations can be combined in various ways, each of which leads to a slightly different finite element procedure. For example, combining

Eqs.(6.6.3)–(6.6.5) and substituting for σ_{ij} in the momentum balance (6.6.2) leads to the following set of equations:

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.6.7)$$

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = - \frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (2\mu^e D_{ij}) + \frac{\partial}{\partial x_j} (\tau_{ij}^p) \quad (6.6.8)$$

$$\tau_{ij}^p + \lambda \tau_{ij}^{\nabla p} = 2\mu^p D_{ij} \quad (6.6.9)$$

This equation set is a natural extension of work on a Newtonian fluid, and was the first one considered by workers in viscoelastic flows [34–36]. Much of the early work centered on the Upper Convected Maxwell (UCM) fluid ($\mu^e = 0$) in the creeping or Stokes flow limit. Under these circumstances, as pointed out by Crochet et al. [5], the mixed finite element formulation associated with (6.6.7)–(6.6.9) has some restrictions with regard to the choice of approximating functions for the extra-stress variables.

Another possible formulation was proposed by Chang, et al. [37] and is the one that will be described here, in some detail. The implicit constitutive equation (6.6.6) can be rearranged and substituted into (6.6.8). The result is a set of equations of the following form

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.6.10)$$

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = - \frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (2\bar{\mu} D_{ij}) - \frac{\partial}{\partial x_j} (\lambda \tau_{ij}^{\nabla p}) \quad (6.6.11)$$

$$\tau_{ij}^p + \lambda \tau_{ij}^{\nabla p} = 2\mu^p D_{ij} \quad (6.6.12)$$

where $\bar{\mu} = (\mu^e + \mu^p)$, D_{ij} is defined in (6.5.11), and the stress rate, τ_{ij}^{∇} , is given in (6.5.8a). The inclusion of the stress rate in the momentum equation appears to be of some benefit for numerical computations [5]. The finite element forms of the above two formulations are often termed MIX (mixed method) formulations and form the basis for many of the later advanced developments.

For two-dimensional geometries the equations in (6.6.10)–(6.6.12) provide six relations for the six unknowns, v_1 , v_2 , P , τ_{11}^p , τ_{22}^p , and τ_{12}^p . In an axisymmetric geometry the system would increase to seven equations with the circumferential (hoop) stress, τ_{33}^p , as the seventh unknown. The distinguishing feature of this system, as compared to a Newtonian problem, is the implicit nature of the constitutive equation that forces the extra-stresses to remain as dependent variables.

6.6.3 Finite Element Model

Application of the finite element procedure to the partial differential equations in (6.6.10)–(6.6.12) follows the standard format. The velocity components, pressure, and extra-stress components are approximated by expansions of the form

$$v_i = \sum_{m=1}^M \Psi_m v_i^m = \Psi^T \mathbf{v}_i \quad (6.6.13a)$$

$$P = \sum_{n=1}^N \Phi_n P_n = \Phi^T \mathbf{P} \quad (6.6.13b)$$

$$\tau_{ij} = \sum_{k=1}^K \Pi^k \tau_{ij}^k = \Pi^T \tau_{ij} \quad (6.6.13c)$$

where Φ , Ψ , and Π are vectors of basis functions, \mathbf{v}_i , \mathbf{P} , and τ_{ij} are vectors of nodal unknowns, and M , N , and K indicate the number of nodes at which the various unknowns are defined. The superscript p for the extra-stress has been omitted for clarity. Using the finite element approximations (6.6.13) in standard weighted-integral statements (i.e., weak forms) of the system in (6.6.10)–(6.6.12) (see Chapter 5), the following system of finite element equations can be obtained.

Momentum

$$\begin{aligned} & \left[\int_{\Omega^e} \rho_0 \Phi \Phi^T \mathbf{v}_j \frac{\partial \Phi^T}{\partial x_j} d\mathbf{x} \right] \mathbf{v}_i + \left[\int_{\Omega^e} \mu \frac{\partial \Phi}{\partial x_j} \frac{\partial \Phi^T}{\partial x_j} d\mathbf{x} \right] \mathbf{v}_i + \left[\int_{\Omega^e} \mu \frac{\partial \Phi}{\partial x_j} \frac{\partial \Phi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_j \\ & + \left[\int_{\Omega^e} -\frac{\partial \Phi}{\partial x_i} \Psi^T d\mathbf{x} \right] \mathbf{P} - \left[\int_{\Omega^e} \lambda \frac{\partial \Phi}{\partial x_j} \nabla \tau_{ij} d\mathbf{x} \right] = \left[\oint_{\Gamma^e} \Phi \tau_{ij} n_j ds \right] \end{aligned} \quad (6.6.14)$$

Continuity

$$\left[\int_{\Omega^e} -\Psi \frac{\partial \Phi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_i = \mathbf{0} \quad (6.6.15)$$

Constitutive

$$\begin{aligned} & \left[\int_{\Omega^e} \Pi \Pi^T d\mathbf{x} \right] \tau_{ij} + \left[\int_{\Omega^e} \Pi \lambda \nabla \tau_{ij} d\mathbf{x} \right] - \left[\int_{\Omega^e} \mu^p \Pi \frac{\partial \Phi^T}{\partial x_j} d\mathbf{x} \right] \mathbf{v}_i \\ & - \left[\int_{\Omega^e} \mu^p \Pi \frac{\partial \Phi^T}{\partial x_i} d\mathbf{x} \right] \mathbf{v}_j = \mathbf{0} \end{aligned} \quad (6.6.16)$$

where the steady-state form of the equations has been considered. In arriving at the above equations, the Green–Gauss or divergence theorem has been used to reduce the second-order diffusion terms to first-order terms plus a boundary integral. The appearance of the boundary integral corresponds to the “natural” boundary conditions for the problem. Also, the stress rate has not been written out due to its complexity; the two-dimensional components of the stress rate are written below in terms of the finite element functions,

$$\begin{aligned} \nabla \tau_{11} &= \Phi^T \mathbf{v}_1 \frac{\partial \Pi^T}{\partial x_1} \tau_{11} + \Phi^T \mathbf{v}_2 \frac{\partial \Pi^T}{\partial x_2} \tau_{11} - 2 \frac{\partial \Phi^T}{\partial x_1} \mathbf{v}_1 \Pi^T \tau_{11} - 2 \frac{\partial \Phi^T}{\partial x_2} \mathbf{v}_1 \Pi^T \tau_{12} \\ & \quad (6.6.17) \end{aligned}$$

$$\begin{aligned} \nabla \tau_{22} &= \Phi^T \mathbf{v}_1 \frac{\partial \Pi^T}{\partial x_1} \tau_{22} + \Phi^T \mathbf{v}_2 \frac{\partial \Pi^T}{\partial x_2} \tau_{22} - 2 \frac{\partial \Phi^T}{\partial x_1} \mathbf{v}_2 \Pi^T \tau_{12} - 2 \frac{\partial \Phi^T}{\partial x_2} \mathbf{v}_2 \Pi^T \tau_{22} \\ & \quad (6.6.18) \end{aligned}$$

$$\begin{aligned} \nabla \tau_{12} &= \Phi^T \mathbf{v}_1 \frac{\partial \Pi^T}{\partial x_1} \tau_{12} + \Phi^T \mathbf{v}_2 \frac{\partial \Pi^T}{\partial x_2} \tau_{12} - \frac{\partial \Phi^T}{\partial x_1} \mathbf{v}_2 \Pi^T \tau_{11} - \frac{\partial \Phi^T}{\partial x_2} \mathbf{v}_1 \Pi^T \tau_{22} \\ & - \left(\frac{\partial \Phi^T}{\partial x_1} \mathbf{v}_1 + \frac{\partial \Phi^T}{\partial x_2} \mathbf{v}_2 \right) \Pi^T \tau_{12} \end{aligned} \quad (6.6.19)$$

Equations (6.6.17)–(6.6.19) can be substituted directly into (6.6.14) and (6.6.16) to produce the complete form of the finite element equations.

Once the form of the interpolation functions Φ , Ψ , and Π is specified (i.e., a particular element is selected), and the geometry of the element is known (i.e., x_i) then the integrals in (6.6.14)–(6.6.16) may be evaluated to produce the required coefficient matrices. The integrals are evaluated via a numerical quadrature procedure, as discussed in Chapters 2 and 3. The discrete system is given by the following matrix equations (it is common to write the momentum and continuity equations as a single system):

$$\begin{aligned} & \left[\begin{array}{ccc} \mathbf{C}_i(\mathbf{v}_i) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_i(\mathbf{v}_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array} \right] \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{P} \end{Bmatrix} + \left[\begin{array}{ccc} 2\mathbf{K}_{11} + \mathbf{K}_{22} & \mathbf{K}_{21} & \mathbf{Q}_1 \\ \mathbf{K}_{12} & \mathbf{K}_{11} + 2\mathbf{K}_{22} & \mathbf{Q}_2 \\ \mathbf{Q}_1^T & \mathbf{Q}_2^T & \mathbf{0} \end{array} \right] \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{P} \end{Bmatrix} \\ & + \left[\begin{array}{ccc} \mathbf{R}_1^{11}(\mathbf{v}_1, \mathbf{v}_2) & \mathbf{R}_1^{22}(\mathbf{v}_1, \mathbf{v}_2) & \mathbf{R}_1^{12}(\mathbf{v}_1, \mathbf{v}_2) \\ \mathbf{R}_2^{11}(\mathbf{v}_1, \mathbf{v}_2) & \mathbf{R}_2^{22}(\mathbf{v}_1, \mathbf{v}_2) & \mathbf{R}_2^{12}(\mathbf{v}_1, \mathbf{v}_2) \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{array} \right] \begin{Bmatrix} \tau_{11} \\ \tau_{22} \\ \tau_{12} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \mathbf{0} \end{Bmatrix} \quad (6.6.20) \end{aligned}$$

and for the constitutive equation

$$\begin{aligned} & \left[\begin{array}{ccc} \mathbf{N} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{N} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{N} \end{array} \right] \begin{Bmatrix} \tau_{11} \\ \tau_{22} \\ \tau_{12} \end{Bmatrix} - \left[\begin{array}{ccc} 2\mathbf{D}_1(\mathbf{v}_1) & \mathbf{0} & 2\mathbf{D}_2(\mathbf{v}_1) \\ \mathbf{0} & 2\mathbf{D}_2(\mathbf{v}_2) & 2\mathbf{D}_1(\mathbf{v}_2) \\ \mathbf{D}_1(\mathbf{v}_2) & \mathbf{D}_2(\mathbf{v}_1) & \mathbf{D}_i(\mathbf{v}_i) \end{array} \right] \begin{Bmatrix} \tau_{11} \\ \tau_{22} \\ \tau_{12} \end{Bmatrix} \\ & + \left[\begin{array}{ccc} \mathbf{C}_i^*(\mathbf{v}_i) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_i^*(\mathbf{v}_i) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_i^*(\mathbf{v}_i) \end{array} \right] \begin{Bmatrix} \tau_{11} \\ \tau_{22} \\ \tau_{12} \end{Bmatrix} - \left[\begin{array}{ccc} 2\mathbf{L}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 2\mathbf{L}_2 & \mathbf{0} \\ \mathbf{L}_1 & \mathbf{L}_2 & \mathbf{0} \end{array} \right] \begin{Bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{P} \end{Bmatrix} = \begin{Bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{Bmatrix} \quad (6.6.21) \end{aligned}$$

where sum on repeated indices is implied. The coefficient matrices shown in Eqs. (6.6.20) and (6.6.21) are defined by

$$\begin{aligned} \mathbf{C}_i(\mathbf{v}_j) &= \int_{\Omega^e} \rho_0 \Phi \Phi^T \mathbf{v}_j \frac{\partial \Phi^T}{\partial x_i} d\mathbf{x}, \quad \mathbf{K}_{ij} = \int_{\Omega^e} \bar{\mu} \frac{\partial \Phi}{\partial x_i} \frac{\partial \Phi^T}{\partial x_j} d\mathbf{x} \\ \mathbf{Q}_i &= - \int_{\Omega^e} \frac{\partial \Phi}{\partial x_i} \Psi^T d\mathbf{x}, \quad \mathbf{F}_i = \oint_{\Gamma^e} \Phi \tau_{ij} n_j ds \\ \mathbf{N} &= \int_{\Omega^e} \Pi \Pi^T d\mathbf{x}, \quad \mathbf{C}_i^*(\mathbf{v}_j) = \int_{\Omega^e} \lambda \Pi \Phi^T \mathbf{v}_j \frac{\partial \Pi^T}{\partial x_i} d\mathbf{x} \\ \mathbf{D}_i(\mathbf{v}_j) &= \int_{\Omega^e} \lambda \Pi \frac{\partial \Phi^T}{\partial x_i} \mathbf{v}_j \Pi^T d\mathbf{x}, \quad \mathbf{L}_i = \int_{\Omega^e} \mu^p \Pi \frac{\partial \Phi^T}{\partial x_i} d\mathbf{x} \end{aligned} \quad (6.6.22)$$

and the matrices \mathbf{R}_k^{ij} are defined by

$$\begin{aligned} \mathbf{R}_1^{11} &= -\mathbf{S}_{11}(\mathbf{v}_1) - \mathbf{S}_{12}(\mathbf{v}_2) + 2\mathbf{T}_{11}(\mathbf{v}_2) + \mathbf{T}_{21}(\mathbf{v}_2), \quad \mathbf{R}_1^{22} = \mathbf{T}_{22}(\mathbf{v}_1) \\ \mathbf{R}_1^{12} &= -\mathbf{S}_{21}(\mathbf{v}_1) - \mathbf{S}_{22}(\mathbf{v}_2) + 2\mathbf{T}_{12}(\mathbf{v}_1) + \mathbf{T}_{21}(\mathbf{v}_1) + \mathbf{T}_{22}(\mathbf{v}_2) \\ \mathbf{R}_2^{11} &= \mathbf{T}_{11}(\mathbf{v}_2), \quad \mathbf{R}_2^{22} = -\mathbf{S}_{21}(\mathbf{v}_1) - \mathbf{S}_{22}(\mathbf{v}_2) + \mathbf{T}_{12}(\mathbf{v}_1) + 2\mathbf{T}_{22}(\mathbf{v}_2) \\ \mathbf{R}_2^{12} &= -\mathbf{S}_{12}(\mathbf{v}_2) - \mathbf{S}_{11}(\mathbf{v}_1) + 2\mathbf{T}_{21}(\mathbf{v}_2) + \mathbf{T}_{12}(\mathbf{v}_2) + \mathbf{T}_{11}(\mathbf{v}_1) \end{aligned} \quad (6.6.23)$$

The \mathbf{S} and \mathbf{T} matrices are defined as

$$\mathbf{S}_{ij}(\mathbf{v}_k) = \int_{\Omega^e} \lambda \frac{\partial \Phi}{\partial x_i} \Phi^T \mathbf{v}_k \frac{\partial \Pi^T}{\partial x_j} d\mathbf{x}, \quad \mathbf{T}_{ij}(\mathbf{v}_k) = \int_{\Omega^e} \lambda \frac{\partial \Phi}{\partial x_i} \frac{\partial \Phi^T}{\partial x_j} \mathbf{v}_k \Pi^T d\mathbf{x} \quad (6.6.24)$$

Finally, Eqs. (6.6.20) and (6.6.21) can be expressed symbolically as

$$\mathbf{C}(\mathbf{v})\mathbf{U} + \mathbf{K}\mathbf{U} + \mathbf{R}(\mathbf{v})\tau = \mathbf{F} \quad (6.6.25)$$

$$\mathbf{N}\tau - \mathbf{D}(\mathbf{v})\tau + \mathbf{C}^*(\mathbf{v})\tau - \mathbf{L}\mathbf{U} = \mathbf{0} \quad (6.6.26)$$

where the vector of unknowns is $\mathbf{U}^T = \{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{P}^T\}$. Equations (6.6.25) and (6.6.26) represent the finite element equations for a viscoelastic (Oldroyd B) fluid for the steady-state case; very little consideration has been given to the time-dependent case. The displayed matrix system is highly nonlinear and fully coupled. Equation (6.6.25) is recognized as the standard form for a Newtonian problem with the addition of the $\mathbf{R}(\mathbf{v})\sigma$ term which comes from the stress rate. Equation (6.6.26) is the finite element analog of the implicit constitutive equation and is seen to be an equation of the advection type for the extra-stress.

Before discussing solution procedures for the matrix system, some comments on the approximations used in (6.6.13) are required. In defining the approximations to the dependent variables all of the basis functions were taken to be different. From experience with mixed methods for Newtonian problems it is well-known that the pressure approximation should be of lower order than the velocity approximation. For most applications in viscoelastic simulations this condition has been met by using quadratic functions for the velocity and linear approximations for the pressure. The choice of an appropriate approximation for the extra-stress is less clearly defined. As mentioned earlier, the problem formulation based on Eqs. (6.6.7)–(6.6.9) has a restriction on the permissible extra-stress basis functions. A solvability condition for the resulting matrix problem requires that the extra-stresses be approximated to the same order as the velocity components [5].

In the present formulation there are no such restrictions and the choice of the interpolation functions used for stress components is quite arbitrary. Heuristic arguments were made for having the extra-stress and pressure be of the same order since they are both “stress-like” variables. However, many studies using both linear and quadratic stress approximations have been produced, and in virtually all cases the mixed method has been limited to flows with small fluid elasticity. In the mid 1980s Marchal and Crochet [38,39] demonstrated that some improvement in behavior of a mixed method could be achieved by using higher-order approximations for the extra-stress. Their general criterion was that the stress approximation should be of sufficient order to represent velocity gradients. They achieved this approximation in practical computations by subdividing each velocity element into 3×3 or 4×4 bilinear subelements on which the extra-stresses were interpolated. This procedure, along with some upwinding techniques, allowed computations to be made at large values of fluid elasticity. The benefits of higher-order stresses were also examined by Rao and Finlayson [40], where a cubic stress approximation was employed. Despite this and more recent work, the problem of stress approximation is not fully understood nor resolved. Further comments will be made in Section 6.9.

6.6.4 Solution Methods

To discuss possible solution procedures for the discretized viscoelastic problem in (6.6.25) and (6.6.26), it is convenient to write the matrix system in a more compact form. Combining the incompressibility constraint with the momentum equation allows the following system to be derived

$$\mathbf{K}_{\mathbf{vv}}(\mathbf{v})\mathbf{U} + \mathbf{K}_{\mathbf{v}\tau}(\mathbf{v})\tau = \mathbf{F} \quad (6.6.27a)$$

$$\mathbf{K}_{\tau\tau}(\mathbf{v})\tau + \mathbf{K}_{\tau\mathbf{v}}\mathbf{U} = \mathbf{0} \quad (6.6.27b)$$

where the vector \mathbf{U} now represents the velocity and pressure variables. The $\mathbf{K}_{\mathbf{vv}}$ term contains the advection, diffusion, pressure gradient, and incompressibility terms from the momentum equation. The second equation in (6.6.27) is a rearranged version of the constitutive equation in (6.6.26).

As with any strongly coupled system of equations, there are two basic approaches to the solution of the system given in (6.6.27): artificially decouple the system and employ a cyclic solution process on the decoupled equations, or directly solve the combined equation set. Though split equation algorithms are generally less effective than combined schemes, they have the potential for some significant computational benefits in the present system.

Consider the simplest cyclic Picard method available for (6.6.27), which is written here for cycle $n + 1$ as

$$\text{Step 1: } \mathbf{K}_{\mathbf{vv}}(\mathbf{v}^n)\mathbf{U}^{n+1} = \mathbf{F} - \mathbf{K}_{\mathbf{v}\tau}(\mathbf{v}^n)\tau^n \quad (6.6.28a)$$

$$\text{Step 2: } \mathbf{K}_{\tau\tau}(\mathbf{v}^{n+1})\tau^{n+1} = -\mathbf{K}_{\tau\mathbf{v}}\mathbf{U}^{n+1} \quad (6.6.28b)$$

At each step of (6.6.28a) a matrix solution is required. However, the matrices in (6.6.28a) are significantly smaller than the matrix associated with the combined system in (6.6.27). Reduction of the size of the matrix problem is especially important for the viscoelastic problem since the number of unknowns per nodal point is very large (e.g., the number of unknowns in a viscoelastic flow can be more than double those of a Newtonian problem for the same finite element mesh). Unfortunately, little work has been done to improve the relatively poor iterative (convergence) performance of methods such as (6.6.28); virtually all viscoelastic problems to date have been solved using a combined equation method.

If the solution to the coupled matrix system is to be obtained by simultaneous solution of the equations in (6.6.26), then it is appropriate to rewrite the system in the operator form

$$\hat{\mathbf{K}}(\hat{\mathbf{U}})\hat{\mathbf{U}} = \hat{\mathbf{F}} \quad (6.6.29)$$

where $\hat{\mathbf{U}}$ now contains all of the problem unknowns. The nonlinear problem shown in (6.6.19) may be solved using any of the fixed point iteration schemes. The Picard method

$$\hat{\mathbf{K}}(\hat{\mathbf{U}}^n)\hat{\mathbf{U}}^{n+1} = \hat{\mathbf{F}} \quad (6.6.30)$$

was used by Coleman [36] for several viscoelastic flow problems. The most commonly used method, however, is Newton's method [26,41–44], which for (6.6.19) can be written as

$$\mathbf{J}(\hat{\mathbf{U}}^n)[\hat{\mathbf{U}}^{n+1} - \hat{\mathbf{U}}^n] = -\hat{\mathbf{K}}(\hat{\mathbf{U}}^n)\hat{\mathbf{U}}^n + \hat{\mathbf{F}} \quad (6.6.31a)$$

where

$$\mathbf{J}(\hat{\mathbf{U}}^n) = \frac{\partial}{\partial \hat{\mathbf{U}}} [\hat{\mathbf{K}}(\hat{\mathbf{U}})\hat{\mathbf{U}} - \hat{\mathbf{F}}] \Big|_{\hat{\mathbf{U}}^n} \quad (6.6.31b)$$

The Jacobian \mathbf{J} for the system given in (6.6.25) and (6.6.26) is very complex but can, in fact, be evaluated analytically. Newton's method converges rapidly and, in general, works very well for flows involving materials with low to moderate elasticity. For highly elastic fluids none of the above algorithms allow convergence, as will be discussed subsequently.

There has been no discussion of solution methods for transient flows, mainly due to the lack of previous work in this area. It is certainly possible to extend the steady-state formulation of Section 6.5 into the time-dependent regime. Following the ideas developed for Newtonian problems [45], the resulting ordinary differential equation system could be integrated in time using an implicit method, such as the trapezoid rule. However, there are two basic difficulties in such an approach. First, since the matrix system for a viscoelastic formulation is so large, the repeated solution of an implicit integration algorithm may not be economically justifiable. Splitting the system during the integration procedure [as in (18)] could alleviate some of the computational burden but would increase concerns regarding stability. The second concern is with the constitutive equation and the fact that the time scale for the material response can be significantly different than the characteristic time for the flow process [46]. Such a situation suggests that a single integration time scale may be inappropriate or not economical; the momentum and constitutive equations may require quite different integration procedures. A method developed by Gartling [47], which is similar to the work of Fortin and Fortin [48], addresses some of the questions regarding transient flows.

6.7 Additional Models of Differential Form

The finite element models developed in Section 6.6 formed the basis for viscoelastic simulations until the later part of the 1980s. Various perturbations on the basic mixed method were attempted to improve computational performance, improve convergence, and most importantly, provide solutions at increasing levels of fluid elasticity. This last problem, the "high Weissenberg number problem" (see Section 6.9), was particularly troublesome as the MIX type algorithms all failed at moderate levels of fluid elasticity. The cause of this failure is complex; some of the issues associated with the phenomena are discussed in a later section.

A number of other formulations have been developed to try to alleviate the high Weissenberg difficulties. In each case, the main objective has been to maintain the momentum and continuity equations as a fully elliptic equation system with well-understood mathematical and numerical properties (e.g., saddle point problem, LBB condition, dominant advection terms, etc.). Also, the constitutive equation treatment has been altered to better account for the hyperbolic nature of the equation set. The following subsections describe the main algorithms developed from these basic ideas. The finite element form of these algorithms will not be detailed as the computational implementation follows directly from the previous section.

6.7.1 Explicitly Elliptic Momentum Equation Method

The first significant reformulation of the viscoelastic flow problem is due to King et al. [49] and is termed the explicitly elliptic momentum equation (EEME) method. It was proven by Joseph et al. [50] that under certain conditions the equation set for the UCM model formulated via a MIX type method could undergo a change of type (change in the characteristics of the equations from imaginary to real) and lose the dominant elliptic character of the system. The essence of the analysis was a criterion on the dependent variables that must be met in order to ensure a consistent and stable numerical solution and prevent any change in the basic characteristics of the equation set. The EEME method was the result of a search for a method that would explicitly satisfy the change of type criterion.

The formulation begins with the basic continuity, momentum, and UCM constitutive models

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.7.1)$$

$$-\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (\tau_{ij}^p) = 0 \quad (6.7.2)$$

$$\tau_{ij}^p + \lambda \nabla_{ij}^p = 2\mu^p D_{ij} \quad (6.7.3)$$

where the momentum equation is simplified by considering only steady Stokes flow and no body forces. Taking the divergence of the stress in (6.7.3) and substituting into (6.7.2) produces a momentum equation of the form

$$-\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (2\mu^p D_{ij}) - \frac{\partial}{\partial x_j} (\lambda \nabla_{ij}^p) = 0 \quad (6.7.4)$$

This equation is similar in form to (6.6.11) though for a different constitutive model. Equation (6.7.4) may be rewritten using the definition of the upper convected derivative as

$$-\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (\mu^p L_{ij} + \mu^p L_{ji}) - \frac{\partial}{\partial x_j} \left(\lambda v_k \frac{\partial \tau_{ij}^p}{\partial x_k} - L_{ik} \tau_{kj}^p - L_{jk} \tau_{ki}^p \right) = 0 \quad (6.7.5)$$

Defining a new tensor variable, χ_{ij} , based on the change of type criterion, the momentum equation may be reformed as

$$\frac{\partial}{\partial x_k} (\chi_{ij} L_{jk}) + L_{ij} \frac{\partial \chi_{jk}}{\partial x_k} = \frac{\partial q}{\partial x_k} \quad (6.7.6)$$

where

$$\chi_{ij} = \delta_{ij} - \lambda \tau_{ij}^p \quad (6.7.7)$$

$$q = P + \lambda v_j \frac{\partial P}{\partial x_j} \quad (6.7.8)$$

The EEME method consists of Eqs. (6.7.1), (6.7.3), and (6.7.6) with the definitions in (6.7.7) and (6.7.8). This formulation is similar in development to the

MIX method of Section 6.6. The major difference between the methods is that the EEME was conceived for an UCM model (no solvent or viscous stress component) while the MIX methods were usually employed with constitutive models having a viscous component. A MIX formulation for an UCM has been shown by a number of investigators to be very limited in terms of maximum attainable Weissenberg number. The advantage of the EEME method is the enforced elliptic nature of the momentum equation in (6.7.6), which is maintained as the fluid elasticity is increased. The tensor variable χ_{ij} , is analogous to an anisotropic diffusivity; the variable q is a modified pressure. The lower-order derivative terms are similar to advection terms. The momentum equation is thus similar to a Navier Stokes problem. In a finite element procedure the modified pressure q is used (and interpolated) in the same way as in a Newtonian formulation.

The EEME formulation was used with a finite element implementation, primarily by the MIT group [49,51], to solve a number of viscoelastic problems. A variety of stress approximations were tested and found to produce consistent and convergent results at reasonably large values of fluid elasticity. The failure of the EEME method at higher values of elasticity was attributed to the poor resolution of elastic boundary layers. The low-order derivative terms in (6.7.6) become dominant at high elasticity, much like the advection terms in a high Reynolds number viscous flow. Note that the EEME method can be extended to constitutive models with a viscous stress contribution, such as the Oldroyd models. However, its inherent design makes it most relevant to situations where the viscous stress is much less than the viscoelastic stress.

6.7.2 Elastic Viscous Stress Splitting Method

Despite the improvements seen with the EEME algorithm, its focus on UCM type models reduced its universality. However, the philosophy of the EEME formulation, involving a strongly elliptic momentum/continuity system and a properly represented hyperbolic constitutive equation, was accepted as a way to improve viscoelastic simulation methods. A number of investigators had also demonstrated that the addition of a viscous contribution to the stress model was computationally helpful. The viscous contribution regularizes the mathematical problem and eliminates the deleterious change of equation type problem that plagues the UCM model.

The elastic viscous split stress (EVSS) method was developed [51] to generalize the EEME method and focus on constitutive models containing a purely viscous component. The formulation followed from previous work on stress splitting, but changed significantly the method for handling higher-order derivatives in the resulting equations. The development of the method again begins with the steady Stokes form of the continuity and momentum equations. In this illustration the Oldroyd model will be used as the constitutive equation with a viscous component.

The basic equations are

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.7.9)$$

$$-\frac{\partial P}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} = 0 \quad (6.7.10)$$

$$\tau_{ij}^p + \lambda \nabla_{ij}^p = 2\mu^p D_{ij} \quad (6.7.11)$$

$$\tau_{ij} = 2\mu^e D_{ij} + \tau_{ij}^p \quad (6.7.12)$$

For the EVSS scheme, a change of variable is introduced with

$$S_{ij} = \tau_{ij}^p - 2\mu^p D_{ij} \quad (6.7.13)$$

Substituting (6.7.13) and (6.7.12) into the momentum equation and (6.7.13) into the constitutive relation yields the following momentum and stress rate equations

$$-\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (2\bar{\mu}^e D_{ij}) + \frac{\partial S_{ij}}{\partial x_j} = 0 \quad (6.7.14)$$

$$S_{ij}^p + \lambda S_{ij}^p = 2\lambda\mu^p \nabla_{ij} \quad (6.7.15)$$

where the viscosity $\bar{\mu}^e$ is the sum of μ^e and μ^p .

The standard EVSS formulation thus consists of equations (6.7.9), (6.7.14), and (6.7.15) with the definitions for S_{ij} and D_{ij} . This system maintains the elliptic nature of the continuity and momentum equation set but introduces a complication for the finite element implementation. The upper convected derivative of the rate of deformation tensor D_{ij} in (6.7.15) contains second derivatives of the velocity field. A number of approaches have been demonstrated to circumvent this problem and include an integration by parts [52] leading to an inconvenient boundary term, the retention of D_{ij} as an unknown and use of a projection method [51], and a discrete version of retaining D_{ij} as an unknown variable [52]. This last method is termed a Discrete EVSS (DEVSS) and has found considerable favor within the viscoelastic community. Other perturbations and refinements of the EVSS formulation have recently been demonstrated. An adaptive procedure has been employed in [53] and [54] to split the stress leading to methods termed Adaptive Viscous Split Stress (AVSS) and Discrete AVSS (DAVSS). The effective viscosity μ^e in (6.7.12) is allowed to vary such that a balance between the viscous and viscoelastic stress levels is maintained and thereby keep the solution procedure under control.

It must be emphasized that the stress splitting methods are beneficial in maintaining the mathematical properties of the momentum/continuity problem; the proper treatment of the hyperbolic constitutive equation remains an important part of the overall algorithm. In most of the EEME and EVSS methods, some type of upwind method is employed in the formulation of the constitutive equations. The Streamline Upwind Petrov-Galerkin (SUPG) method mentioned in Chapter 4 is one popular approach to the convection operator. The discontinuous Galerkin (DG) method, first outlined by Lesaint and Raviart [55], is another method for computationally treating first-order hyperbolic equations. This method was first applied to the viscoelastic problem in [48].

The additional formulations outlined above have improved considerably the finite element computational capabilities for viscoelastic simulations. However, substantial work remains to be done, since most of these methods are not affordable in three-dimensional, transient, and/or multiple relaxation time simulations.

6.8 Finite Element Model of Integral Form

Finite element procedures for viscoelastic fluids of the integral type have generally received less attention than those for differential models. This can be attributed primarily to the difficulties in implementation of an integral constitutive equation. In this section we will outline the steps needed to use such a model and provide references to work that describes the details of several algorithms.

For consideration of a specific example the Maxwell fluid described by Eq. (6.5.22) will be used. The basic equations are rewritten here for reference

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (6.8.1)$$

$$\rho \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j} (\tau_{ij}) \quad (6.8.2)$$

$$\tau_{ij} = \frac{\mu_0}{\lambda^2} \int_{-\infty}^t \exp [-(t-t')/\lambda] [C_{ij}^{-1}(t') - \delta_{ij}] dt' \quad (6.8.3)$$

Note that the stress decomposition procedure of Eq. (6.5.13) can also be used with an integral model; its use in the present case would lead to an integral form of the Oldroyd B fluid.

The balance laws, (6.8.1) and (6.8.2), can be discretized in the standard way, using a mixed (velocity, pressure) finite element procedure. The result of such a process is the following familiar set of matrix equations

$$\mathbf{Q}^T \mathbf{v} = 0 \quad (6.8.4)$$

$$\mathbf{C}_i(\mathbf{v})\mathbf{v} + \mathbf{Q}_i\mathbf{P} = \mathbf{F}_i - \int_{\Omega^e} \frac{\partial \Phi}{\partial x_j} \tau_{ij} d\mathbf{x} \quad (6.8.5)$$

which are written here for the steady-state case. Note that in writing Eq. (6.8.5), Φ is assumed to be the velocity basis (weighting) function; an integration by parts is responsible for the form of the stress integral. The arrangement of terms in (6.8.5) anticipates the use of an iterative solution procedure. In the first part of the iteration scheme, Eqs. (6.8.4) and (6.8.5) can be solved for the velocity and pressure given a known (or predicted) stress field. The task then is to develop an algorithm for the prediction of the extra-stress given a velocity field from (6.8.5).

From Eq. (6.8.3), it is apparent that to predict the extra-stress at a particular location, say, a nodal point or quadrature point, the strain history for that point must be evaluated over past time. This computation is usually done in three stages. In the first stage, the particle path (streamline for steady flow) through the point in question is constructed. Procedures for this construction range from the Lagrangian-like computation of displacement fields for each finite element [56] to the direct computation of the streamlines [57]. The construction of the trajectory allows the previous locations of the particle (i.e., node or quadrature point) to be established. The second stage requires the evaluation of the strain measure (e.g., C_{ij}^{-1}) along the particle trajectory. The final stage uses a numerical (Laguerre) quadrature to evaluate the history integral (6.8.3) and produce the extra-stress at a particular location.

The details of the particle tracking and strain evaluation depend directly on the particular type of finite element approximation used for the velocity field. A discussion of several different approaches may be found in the work of Viriyayuthakorn and Caswell [56], Dupont et al. [57] and Bernstein et al. [58]. Once a prediction of the stress field has been made the stress forcing function in (6.8.5) can be evaluated and a new velocity obtained. This cyclic procedure is continued until convergence of the dependent variables is achieved. Details regarding the implementation of the various iteration procedures may also be found in [56,57].

There are several drawbacks to the integral methods as currently implemented. The basic structure of the equations in (6.8.1)–(6.8.3) does not permit the easy use of higher-order iterative methods, such as Newton's method. Convergence is thus limited to a linear rate, which can be a severe computational burden. The particle tracking algorithms appear to be the weak point of the overall procedure. For recirculating flows the methods must be of sufficient accuracy to ensure the closure of streamlines. This is a rather difficult task when dealing with approximate numerical schemes and an incompressible fluid. Also, the extension of the particle tracking ideas to time-dependent flows would appear to be very difficult. Despite these disadvantages the work on integral constitutive equations should continue to increase, since these models are somewhat better at predicting realistic viscoelastic behavior than their differential counterparts.

6.9 Unresolved Problems

6.9.1 General Comments

To conclude this brief treatment of finite element methods for viscoelastic flows, it is appropriate to outline some of the unresolved problems and areas of research in this field. Numerical difficulties in the simulation of Newtonian fluids are generally associated with the occurrence of large values of some relevant nondimensional parameter, e.g., the Reynolds number, Peclet number, or Rayleigh number. The situation is quite similar for the viscoelastic problem though the nondimensional parameters are not as familiar.

As noted in a previous section, there are two time scales involved in a viscoelastic flow: the time constant for the material (relaxation time, λ) and the characteristic time for the flow process, t_f . The ratio of these time parameters is called the Deborah number, $De = \lambda/t_f$. Very often the characteristic time for the flow is taken as the reciprocal of a typical (wall) shear rate, so that $De = \lambda\dot{\gamma}_w$. The nondimensionalization of simple constitutive equations, such as the Maxwell model, can lead to the definition of a second characteristic parameter. The Weissenberg number is typically defined as $W = \lambda U/L$ where U and L are a characteristic velocity and length scale for the flow. In many cases the Weissenberg and Deborah numbers can have the same definition; they also tend to be used interchangeably, which leads to some confusion in the literature.

The Deborah and Weissenberg numbers both indicate the relative importance of fluid elasticity for a given flow problem. High values of W (or De) indicate a material response that is very solid-like, while low values of the parameter represent “small” departures from normal viscous fluid behavior.

The continuing outstanding problem in the numerical simulation of viscoelastic fluids is the “high Weissenberg (or Deborah) number problem.” In essence, this phrase refers to the observed behavior that, as the Weissenberg number is increased for a given problem, a critical value is reached (W_{cr}) beyond which the numerical algorithm fails. Failure in the present context normally consists of an initial degradation in the solution (e.g., spatial oscillations in the dependent variables, especially the stresses) followed by divergence of the solution algorithm. The high Weissenberg limit is a universal phenomenon in that it has been observed in all weak form Galerkin finite element models and finite difference methods and for all types of constitutive equations. The critical value of the Weissenberg number is very sensitive to the particular flow geometry, numerical algorithm, constitutive equation, and computational mesh. Unfortunately, from an engineering analysis viewpoint the limiting value W_{cr} is usually too small to be of practical interest (i.e., they are perturbations to the Newtonian solution).

The failure of the most numerical models to provide solutions for highly elastic fluids has spawned a great deal of research activity as evidenced by the numerous publications and several international workshops on the subject [59]. Though the basic causes of the problem have yet to be fully identified, a number of issues have been explored that contribute to the basic understanding of several parts of this complex problem. A summary of some of these topics is given below. For the reader interested in further details regarding the many facets of the “high Weissenberg number problem” the text by Crochet et al. [5], the reviews by Crochet [60] and Keunings [61], and recent issues of the *Journal of Non-Newtonian Fluid Mechanics* are recommended. An excellent review of the current computational state-of-the-art has been compiled by Baaijens [62] and a review of the more mathematical issues associated with modeling viscoelastic fluids can be found in the work of Renardy [63]. Recent studies using the least-squares finite element models and k -version finite element models [64–73] have the potential to solve high Weissenberg number problems.

6.9.2 Choice of Constitutive Equation

Much of the early numerical work on viscoelastic flows was carried out using the Maxwell constitutive model. This was mainly due to the relative simplicity of the equation and the belief that it was prototypical of the more complex constitutive relations. Work by Phan Thien [74–76] demonstrated that for some flows there are inherent instabilities in the Maxwell constitutive model that occur above a critical Weissenberg number. Such a proof demonstrates one of the subtle dangers of this class of problems — the use of approximate constitutive equations that are difficult or impossible to test analytically for non-physical behavior in complex flows.

Many recent investigations have abandoned the Maxwell/Oldroyd family of fluid models in favor of more realistic (and complex) equations such as those due to Johnson–Segalman and Phan Thien–Tanner. There is, of course, no proof that the more complex models do not also contain built-in instabilities that may or may not be physically realistic. Despite the results shown in [48,56,57] most investigators do not feel that the high Weissenberg limit is due solely to the choice of the constitutive model, though it is certainly regarded as a contributing factor. It should also be noted that virtually all numerical work has considered only single relaxation time

models, while it is clear that most constitutive models require multiple relaxation times to approximate realistic fluid behavior. The large computational burden for single relaxation time models has prevented more realistic relaxation spectra from being investigated.

6.9.3 Uniqueness and Existence of Solutions

The complexity of the equations describing viscoelastic flows precludes any proof of solution uniqueness or existence. Indeed, from the behavior of the simple Newtonian fluid it is to be expected that bifurcations and multiple solutions will exist for flows of these types of fluids. Various investigators [26,41,46] working with several different geometries and constitutive relations have demonstrated the occurrence of limit points (loss of solution with increasing Weissenberg number) and bifurcation points (multiple solutions) in numerical simulations. Typically such features of the flow are computed using a continuation (or imbedding) procedure and Newton's method. Corner singularities routinely cause numerical difficulties and mathematical uncertainty in viscoelastic flows.

The question remains as to whether such behavior is an attribute of the physical model (differential constitutive equation) or is an artifact of the numerical discretization process. Keunings [77] and Debbaut and Crochet [78] have provided arguments for such phenomena being of a numerical nature in at least two particular simulations. Their findings, however, do not eliminate the concern over encountering true bifurcations or limit points under other circumstances.

Various experimental investigations of viscoelastic flows, such as [79], have demonstrated the occurrence of time-dependence and three-dimensional flow structure in relatively simple two-dimensional geometries. With the numerical emphasis on two-dimensional, steady simulations, this type of behavior is clearly not predictable. Issues such as these emphasize the difficulty in discriminating between behavior of the discretized equations, the continuum equations, and the physical system.

6.9.4 Numerical Algorithm Problems

A particularly disturbing behavior that was observed in many numerical simulations is a pathology associated with mesh refinement studies. Typically, a solution on one mesh discretization could be obtained up to a critical Weissenberg number of, say, W_{cr}^{coarse} ; a refined mesh solution would produce a solution up to W_{cr}^{fine} , where $W_{cr}^{fine} < W_{cr}^{coarse}$. Stated differently, the finite element procedure did not appear to converge as the mesh was refined — a fundamental property of a stable algorithm. Such behavior was reported for several two-dimensional simulations [5,44] and studied in detail in a one-dimensional flow [42].

In [42] it was conjectured that this behavior was due to an improper choice of the approximating functions for the extra-stress. Little theoretical guidance is available for the choice of stress approximation relative to the velocity or pressure; the LBB condition must usually still be satisfied for the momentum and continuity equations. As noted previously, the work of Marchal and Crochet [38,39] and others [40] produced some support for the need for higher-order stress approximations. Such anomalous convergence behavior could also stem from the use of an inappropriate solution algorithm or formulation for the constitutive equation.

Again, improvements in the treatment of the hyperbolic nature of the constitutive equation seem to yield improved computations.

Extensive work in this area by Dupret, et al. [80] eventually led to a better understanding of some aspects of the problem. Viscoelastic flows are particularly difficult to model due to: (a) the presence of normal stresses, which lead to very high stress gradients and thin elastic boundary layers, (b) the occurrence of singularities and stress concentrations, and (c) the strong hyperbolic character of the constitutive equation. Large stress gradients and singularities place a high demand on mesh resolution for the stress variables. The hyperbolic nature of the constitutive equation requires that an appropriate numerical method be used for the equation set; upwind techniques or Petrov–Galerkin methods are one possible approach to this problem. When any of the above demands are not met, as was the case in most standard, mixed finite element methods, the discrete equation systems for the viscoelastic problem could produce improper evolutionary behavior. Dupret, et al. [80] tracked this behavior by monitoring a characteristic tensor and correlating its loss of positive definiteness with the occurrence of numerical limit points in the solution. The combination of higher-order stress approximations and upwind techniques for the constitutive equation proved to stabilize the method, and allow solutions at very high Weissenberg numbers with appropriate mesh convergence behavior. However, upwinding and Petrov–Galarkin methods are not without their own problems. The addition of artificial diffusion with these methods must be carefully evaluated to ensure that a true solution is not masked/ altered by the numerical technique. To some extent, the use of streamline upwind methods or discontinuous Galerkin methods has been beneficial in this regard.

A proof by Fortin and Pierre [81] showed under what conditions a higher-order stress approximation was required and confirmed the method developed by Marchal and Crochet [39]. This proof, though not applicable to all formulations, also pointed to the need to properly formulate the momentum/continuity equations as a strongly elliptic system. The EVSS formulation [51] eventually developed out of these considerations and led to substantial improvements in the numerical formulation. It should be noted that a general proof for an LBB type relation for all constitutive equations and system formulations is still lacking. In essence, guidance is still often lacking in what type of stress interpolation is admissible.

6.9.5 Equation Change of Type

The work of Joseph et al. [50] demonstrated that the partial differential equations describing viscoelastic flows can sometimes undergo a formal change of type from elliptic to hyperbolic as the Weissenberg number increases. This is analogous to the transonic flow problem where local supercritical flow regions can be embedded in an otherwise subcritical flow. The proof was for a constitutive model without a viscous component, such as the Upper Convected Maxwell model. This work caused a great deal of activity since so much of the early computational work was based on this model. In combination with the work of Dupret, et al. [80] on loss of evolutionary behavior, the change of type analysis spurred development of the EEME method [49]. The proper formulation of the elliptic part of the viscoelastic problem was reinforced by this work.

The change of type was found to occur for the equations describing the fluid vorticity; the occurrence of real characteristics in such an equation admits to the possibility of discontinuities in the vorticity. The occurrence of shear waves has been observed in several transient solutions of viscoelastic flows [47,82–84]. Standard velocity based finite element methods are capable of resolving such flows, as will be demonstrated in the next section.

6.9.6 Closure

With the development of formulations such as the EVSS and its derivative methods, the high Weissenberg number problem is not as formidable as it once was. Solutions at high values of fluid elasticity have been produced and verified through mesh convergence studies. Not all problems have been eliminated however. At very high values of elasticity, numerical methods still have convergence problems. Most investigators believe this is a problem of mesh resolution and thin elastic boundary layers. Until higher resolution simulations are produced, this remains conjecture. It is clear that many simulations, when compared with experiment, fall short of a predictive capability. The constitutive relation is blamed in this case as being not representative of the viscoelastic material. The trend here is to use multimode models (multiple relaxation times) or more sophisticated micro models coupled with the macro flow equations. Finally, though some of the issues described above have been worked around computationally, a clear mathematical understanding of many of these areas is still not available.

6.10 Numerical Examples

6.10.1 Preliminary Comments

In this section a few example problems are described to demonstrate some of the procedures discussed in previous sections. The first examples consider inelastic, non-Newtonian flows and the second group illustrates a couple of viscoelastic applications. Most of the inelastic examples presented here were obtained using the mixed finite element model contained in the NACHOS II code [10], unless otherwise stated. Use of the penalty finite element models for power-law fluids can be found in [85–87]. For further examples, the literature cited in the text should be consulted.

The computations in the viscoelastic section will be limited to a few problems that illustrate some of the non-Newtonian behavior of a viscoelastic fluid. A variety of simulations have been reported in the literature. However, as example problems they suffer from a common shortcoming in that the reported solutions are often little different from those of a Newtonian fluid, mainly due to the “high Weissenberg limit” described above. One exception to the last comment is the work on die swell by Tanner [88], Crochet and Keunings [43,89,90], and others [91], which does show some departure from the Newtonian case for even slightly elastic fluids.

6.10.2 Buoyancy Driven Flow in a Cavity

The first example is concerned with the flow of a power-law fluid inside a rectangular cavity (see Figure 6.10.1). The vertical walls of the cavity are held at uniform but different temperatures, while the horizontal boundaries are insulated. The differentially heated walls produce a buoyancy driven flow in the

cavity. The computational boundary conditions for this problem consist of specified temperatures on the vertical walls, zero fluxes on the horizontal walls, and zero velocities on all four boundaries. The pressure level was set by specifying a pressure value at one point on the boundary of the cavity.

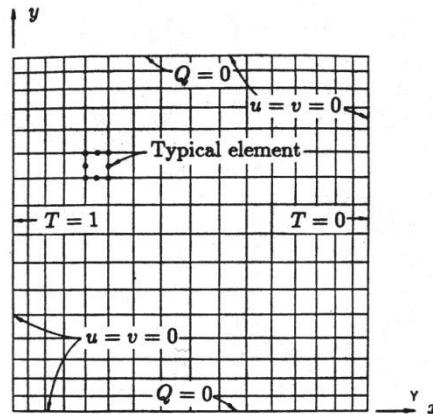


Figure 6.10.1: Schematic of natural convection in a cavity.

Figures 6.10.2 and 6.10.3 contain contour plots of the temperature and stream function obtained for three values of the power-law index. The $n = 1.0$ case corresponds to a Newtonian fluid at a Rayleigh number of $Ra = 10^4$ while the other two cases are for shear thinning fluids at the same nominal Rayleigh number. The effect of the shear thinning viscosity is evident as the fluid motion becomes more vigorous and complex for the same basic driving force. The illustrated cases were solved using a Picard iteration scheme starting from a zero initial solution field and required approximately 10 to 12 iterations to reach convergence.

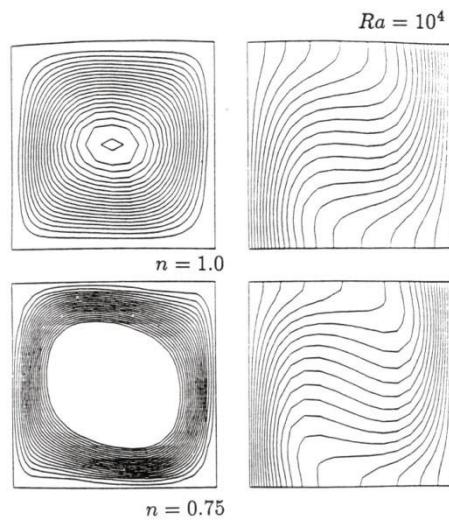


Figure 6.10.2: Streamlines and isotherms, natural convection in a cavity, power law fluid, $n = 0.75$.

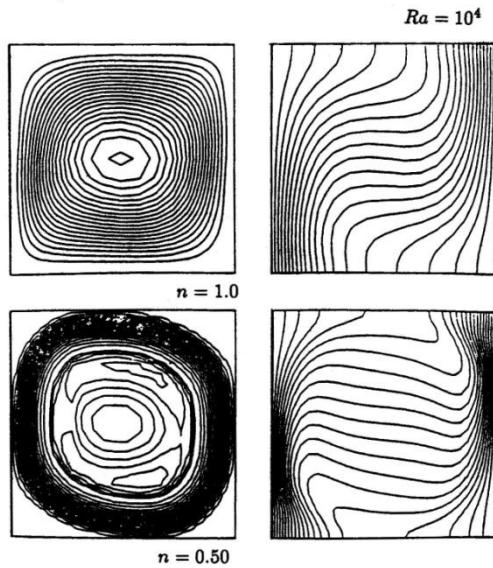


Figure 6.10.3: Streamlines and isotherms, natural convection in a cavity, power law fluid, $n = 0.50$.

6.10.3 Driven Cavity Flow

An isothermal driven cavity flow is considered as a second example. A Bingham fluid is contained in a planar cavity, three sides of which are fixed, with the fourth side moving at unit velocity in its own plane. For purposes of computation, the Bingham parameters are taken to be $\mu = 1.0$ and $\mu_r = 1000.0$ with τ_0 being varied over the range $0 \leq \tau_0 \leq 0.50$. The nominal Reynolds number for the problem is $Re = 100$. The computational mesh for this problem is the same as shown in Figure 6.10.1. Again, the Picard scheme was used to solve the problem with the extrapolation procedure of Section 6.4 also being employed.

Figure 6.10.4 contains contour plots of the stream function at four different values of the yield stress. The $\tau_0 = 0$ case is a Newtonian fluid. With increasing yield stress, the motion in the cavity is reduced and relegated to a small area near the moving lid of the cavity. This effect is seen more clearly in the velocity profiles shown in Figure 6.10.5. Note also that, as shown in Figure 6.10.5, the velocity profiles show a “break” near the vortex center. This is due to the relative magnitude of the deformation rate above and below the vortex center. Actually, the break point should occur at zero velocity but is offset slightly due to the use of a constant viscosity within each element and a relatively coarse mesh.

6.10.4 Squeeze Film Flow

The third example considers the flow of a plastic (Bingham) fluid in a squeeze film apparatus. This type of device is often used to measure the yield parameters for lubricating greases and other “stiff” materials. The geometry and a typical mesh are shown in Figure 6.10.6. Two circular plates are separated by a fluid layer of initial height h . At time zero the upper plate moves downward with a fixed velocity.

The motion of the plate is slow enough that a creeping flow assumption may be used and a series of steady-state problems examined with h as a parameter.

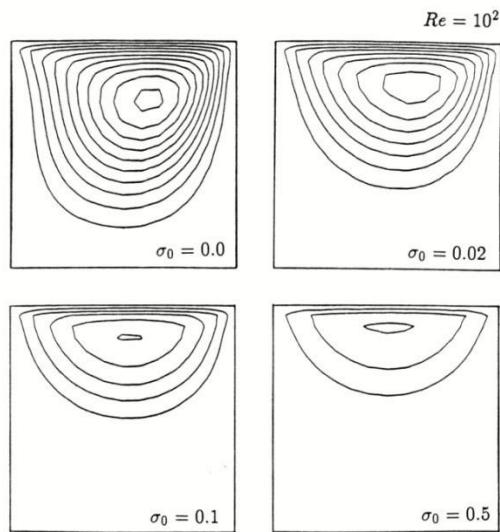


Figure 6.10.4: Streamlines for driven cavity flow; Bingham fluid; various yield stresses.

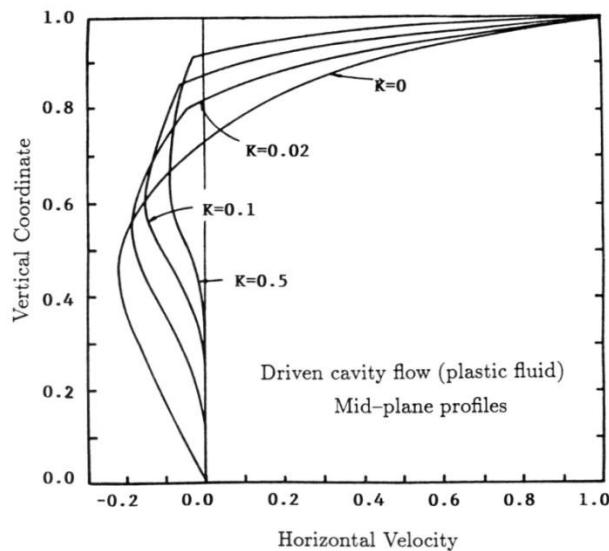


Figure 6.10.5: Midplane velocity profiles for driven cavity flow; Bingham fluid.

Boundary conditions for this simulation consisted of a specified nonzero vertical velocity and a zero horizontal velocity along the top surface of the domain. Along the vertical centerline the horizontal velocity is set to zero and the shear stress (natural boundary condition) vanishes due to symmetry. The horizontal symmetry plane has a zero vertical velocity and a zero shear stress. The outflow boundary assumes a zero vertical velocity (parallel flow assumption) and a constant normal traction. A Picard iteration scheme with extrapolation was used as a solution procedure; up to 12 iterations were required to obtain convergence.

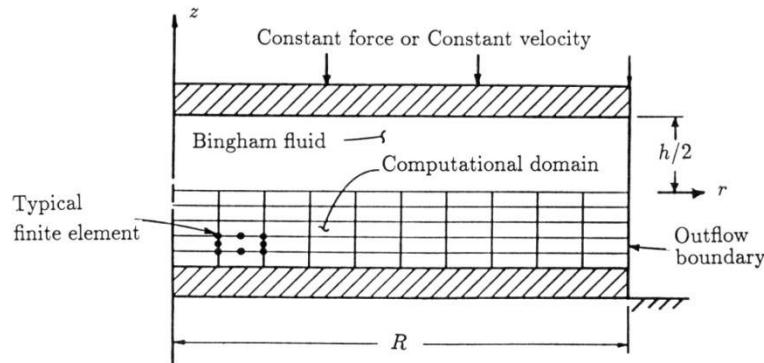


Figure 6.10.6: Schematic of squeeze film flow.

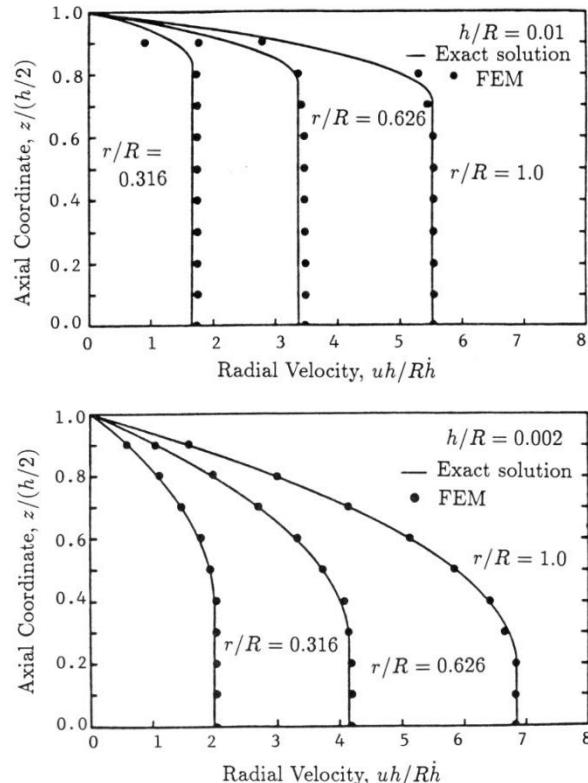


Figure 6.10.7: Radial velocity profiles for the squeeze film flow problem (Bingham fluid). (a) $h/R = 0.01$. (b) $h/R = 0.002$.

Radial velocity profiles for two plate separations are shown in Figure 6.10.7, where they are compared to an analytical solution developed in [17]. The fluid near the center plane of the device flows as an unyielded plug while near the plate a more viscous boundary layer-like flow occurs. The demarcation line between yielded and unyielded material is illustrated in Figure 6.10.8. The agreement with the analytical solution is quite good.

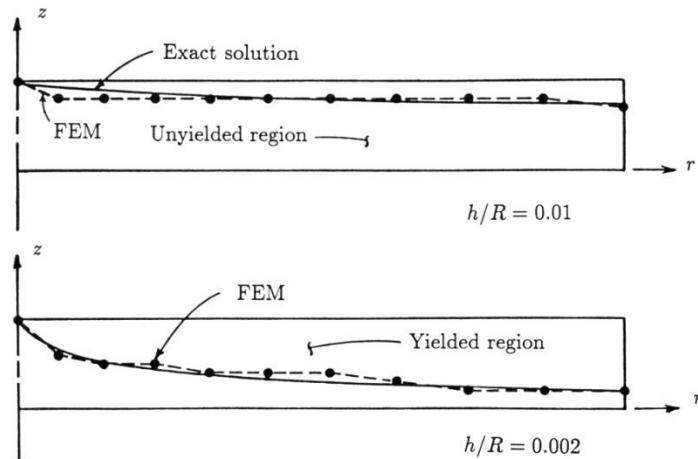


Figure 6.10.8: Plots of the yield surface for the squeeze film flow problem (Bingham fluid). (a) $h/R = 0.01$. (b) $h/R = 0.002$.

6.10.5 Time-Dependent Poiseuille Flow

The first viscoelastic example is a geometrically simple flow with a complex physical response. Consider the problem of an initially quiescent viscoelastic fluid contained between infinite parallel plates. At time zero a specified, constant pressure gradient is imposed. The objective is to compute the time-dependent response of the fluid until it reaches a steady state (Poiseuille flow). A schematic of the problem is shown in Figure 6.10.9.

The above problem was solved analytically by Rivlin [84] for both a Newtonian and an upper-convected Maxwell fluid. The numerical solutions presented here were obtained using the finite element scheme described in [47,48]; results are reported for Newtonian and Maxwell fluids.

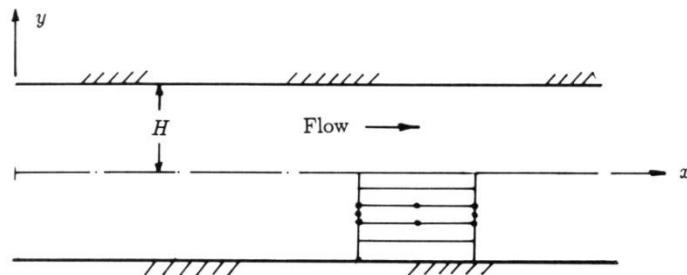


Figure 6.10.9: Schematic of Poiseuille flow.

Shown in Figures 6.10.10 and 6.10.11 are representative time histories and velocity profiles for the response of a Newtonian fluid. The evolution from the rest state to a fully developed parabolic profile is seen to be a smooth, asymptotic process. Comparison of the numerical results with the analytical work of Rivlin shows excellent agreement (less than 1% difference).

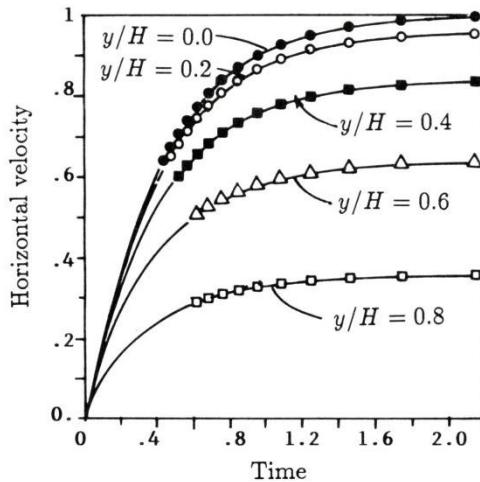


Figure 6.10.10: Velocity histories for Poiseuille flow, Newtonian fluid.

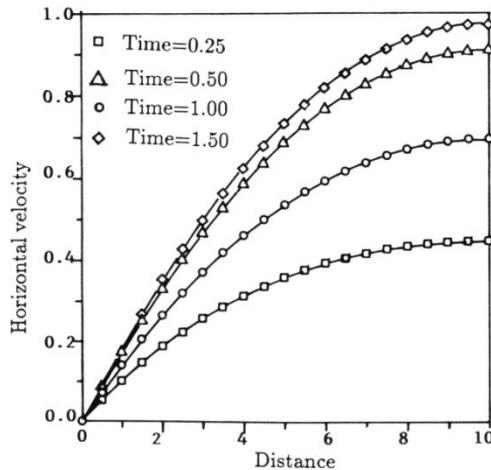


Figure 6.10.11: Velocity profiles for Poiseuille flow, Newtonian fluid.

The response of a viscoelastic fluid, such as a Maxwell material, is quite different. Figure 6.10.12 shows a typical time history for the Maxwell fluid. The damped oscillatory behavior is quite evident and represents a streamwise oscillation of the fluid. This is seen more clearly in Figure 6.10.13, where velocity profiles at various times are illustrated. Early in the start-up process the velocity field shows the propagation of a shear wave from the wall toward the centerline of the channel. The subsequent reflection of the wave causes the oscillatory behavior seen in Figure

6.10.12. At long times the velocity field approaches the expected steady-state profile. A comparison of this solution with the analytical result again shows excellent agreement.

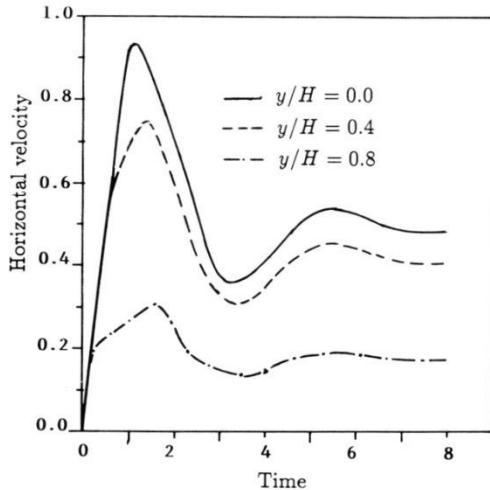


Figure 6.10.12: Velocity histories for Poiseuille flow, Maxwell fluid.

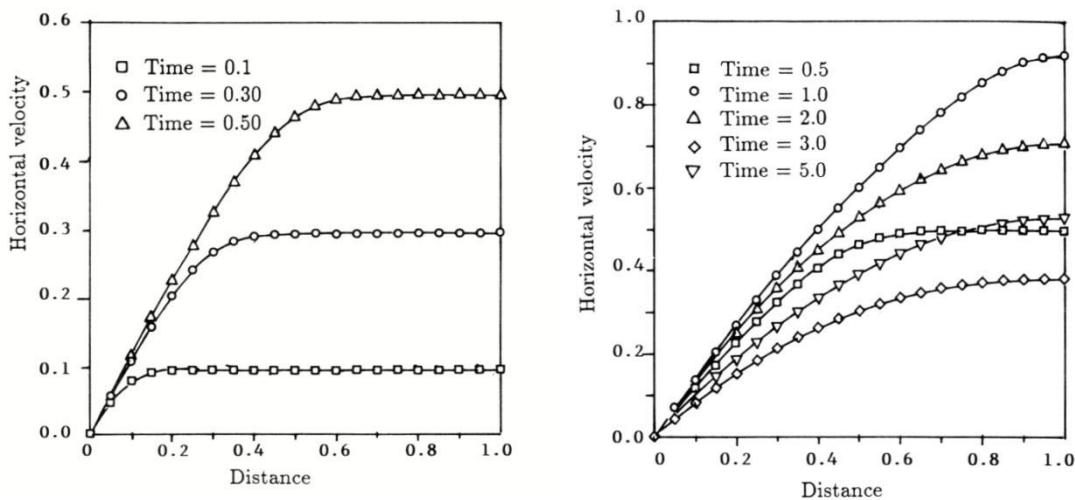


Figure 6.10.13: Velocity profiles for Poiseuille flow, Maxwell fluid.

Figure 6.10.14 shows a series of shear stress profiles that illustrates the early time behavior of the shear wave as it propagates across the channel. The normal stress component has a similar behavior. Though the present example is quite simple, it does illustrate the complex behavior that can occur in fluids with memory.

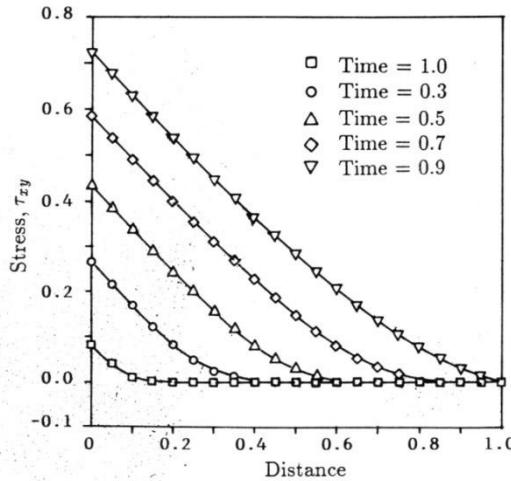


Figure 6.10.14: Shear stress profiles for Poiseuille flow, Maxwell fluid.

6.10.6 Four-to-One Contraction Problem

A standard test problem for the study of viscoelastic flow behavior is the entry flow in axisymmetric, four-to-one contraction. This geometry is popular since it is of technological importance for the polymer processing industry and also exhibits significant flow variation with changes in fluid elasticity. In particular, for several different polymers, as the fluid elasticity is increased (i.e., Weissenberg number is increased) the vortex structure in the largest entry tube undergoes a significant increase in length. The objective of many numerical simulations has been to predict this trend in vortex growth with a suitable constitutive model.

To illustrate typical results for the four-to-one contraction the method proposed by Rao and Finlayson [40] is employed. Figure 6.10.15 contains a schematic of the problem with the velocity boundary conditions shown; a typical quadrilateral mesh is shown in Figure 6.10.16. An Oldroyd B constitutive model was used with a typical mixed method; a cubic approximation for the stress components was used in conjunction with quadratic velocities and a linear pressure. A Newton iteration method with continuation was used to solve the four-to-one contraction for a series of Weissenberg numbers. It was found that to achieve significant values of the Weissenberg number an upwind procedure for the constitutive equation was required. Full details of all the combinations of formulations and algorithms that were tried on this problem are available in [92].

Computed results for the four-to-one contraction in the form of stream function contours are shown in Figure 6.10.17. The plots correspond to increasing values of the recoverable shear, S_R , which is directly proportional to the Weissenberg number. The simulations predict the correct trend in that the vortex length increases as the fluid elasticity increases. Quantitative comparisons with experimental data show that numerical simulations underpredict the vortex growth. This discrepancy is at least partly the result of using approximate constitutive models.

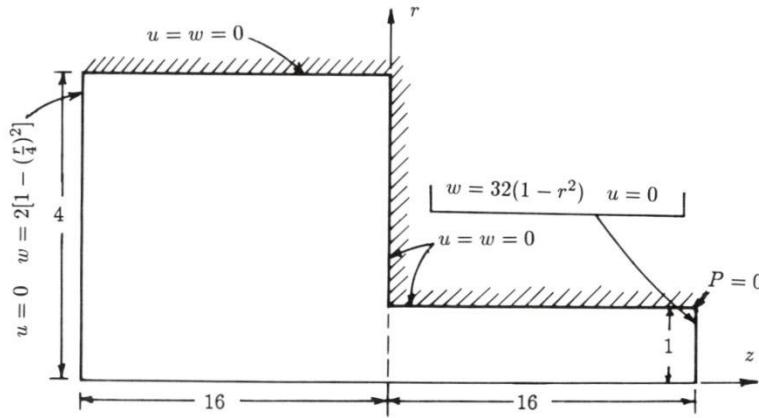


Figure 6.10.15: Schematic for axisymmetric 4:1 contraction problem.

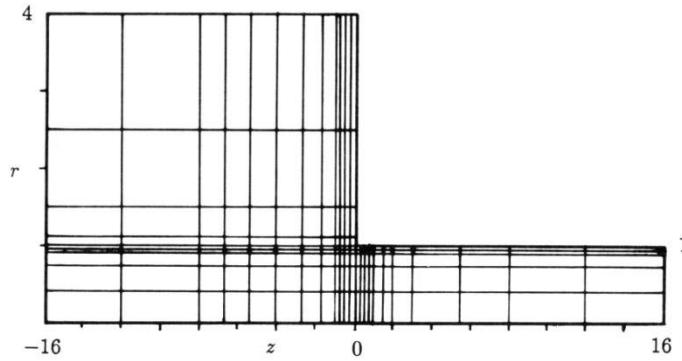


Figure 6.10.16: Quadrilateral mesh for 4:1 contraction. The stress field is approximated using cubic polynomials on each element.

Problems

6.1 The general Oldroyd model is given by (see [5])

$$\begin{aligned} \tau_{ik} + \lambda \overset{\circ}{\tau}_{ik} + \mu \tau_{jj} D_{ik} - \mu_1 (\tau_{ij} D_{jk} + \tau_{kj} D_{ji}) \\ = 2\mu_0 \left(D_{ik} + \lambda_1 \overset{\circ}{D}_{ik} - \mu_2 D_{ij} D_{jk} \right) \end{aligned}$$

where μ_0 is a constant viscosity coefficient, and $(\lambda, \lambda_1, \mu, \mu_1, \mu_2)$ are material constants. Deduce the upper-convected, lower-convected, and corotational Maxwell models by assigning values to the material constants in the Oldroyd model.

6.2 The White–Metzner model [29] is defined by

$$\tau_{ik} + \lambda(I_2) \nabla \tau_{ik} = 2\mu_0(I_2) D_{ik}$$

which describes viscoelastic effects in flow problems that are dominated by the shear viscosity. Develop the finite element model governing viscoelastic flows in two-dimensional Cartesian geometries using the White–Metzner constitutive equation (see [91]).

6.3 Repeat Problem 2 for flows in axisymmetric two-dimensional geometries.

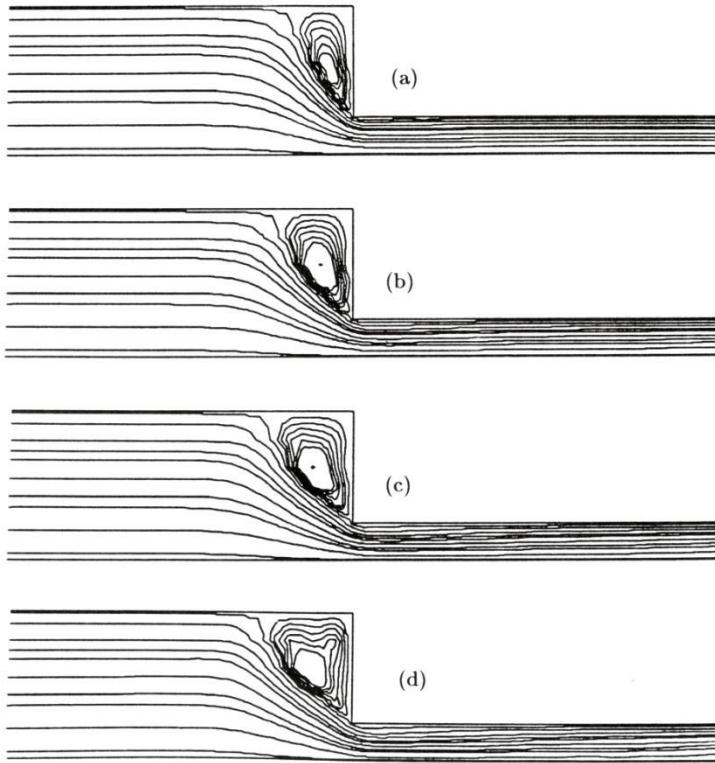


Figure 6.10.17: Streamline for flow in an axisymmetric, 4:1 contraction using an Oldroyd B fluid with recoverable shear values of (a) $S_R = 10$, (b) $S_R = 20$, (c) $S_R = 30$, and (d) $S_R = 40$ (from [91]).

References for Additional Reading

1. R. B. Bird, R. C. Armstrong, and O. Hassager, *Dynamics of Polymeric Liquids, Vol. 1: Fluid Mechanics*, Second Edition, John Wiley & Sons, New York (1971).
2. A. S. Lodge, *Elastic Liquids*, Academic Press, New York (1964).
3. K. Walters, *Rheometry*, Chapman and Hall, London, U.K. (1975).
4. R. I. Tanner, *Engineering Rheology*, Clarendon Press, Oxford, U.K. (1985).
5. M. J. Crochet, A. R. Davies, and K. Walters, *Numerical Simulation of Non-Newtonian Flow*, Elsevier, Amsterdam, The Netherlands (1984).
6. J. R. A. Pearson, *Mechanics of Polymer Processing*, Elsevier, London, U.K. (1985).
7. J. N. Reddy, *Introduction to Continuum Mechanics with Applications*, Cambridge University Press, New York (2008).
8. R. B. Bird, W. E. Stewart, and E. N. Lightfoot, *Transport Phenomena*, John Wiley & Sons, New York (1960).
9. J. G. Oldroyd, “A Rational Formulation of the Equations of Plastic Flow for a Bingham Solid,” *Proceedings of the Cambridge Philosophical Society*, **43**, 100–105 (1947).
10. D. K. Gartling, “NACHOS II-A Finite Element Computer Program for Incompressible Flow Problems,” Sandia National Laboratories Report, SAND86-1816, Albuquerque, NM (1987).
11. R. L. Lee and P. M. Gresho, “Development of a Three-Dimensional Model of the Atmospheric Boundary Layer Using the Finite Element Method,” *Lawrence Livermore National Laboratory Report, UCRL-52366*, Livermore, California (1977).

12. D. K. Gartling, "Finite Element Analysis of Convective Heat Transfer Problems with Change of Phase," in *Computer Methods in Fluids*, K. Morgan, C. Taylor, and C. A. Brebbia (eds.), Pentech Press, London, U.K., 257–284 (1980).
13. D. J. Naylor, "Stresses in Nearly Incompressible Materials by Finite Elements with Application to the Calculation of Excess Pore Pressures," *International Journal for Numerical Methods in Engineering*, **8**, 443–460 (1974).
14. E. Hinton and J. S. Campbell, "Local and Global Smoothing of Discontinuous Finite Element Functions Using a Least Squares Method," *International Journal for Numerical Methods in Engineering*, **8**, 461–480 (1974).
15. E. Hinton, F. C. Scott, and R. E. Ricketts, "Local Least Squares Stress Smoothing for Parabolic Isoparametric Elements," *International Journal for Numerical Methods in Engineering*, **9**, 235–238 (1975).
16. D. K. Gartling, "The Numerical Simulation of Plastic Fluids," in *Proceedings of Third International Conference on Numerical Methods in Laminar and Turbulent Flow*, C. Taylor, C. Johnson, and I. Smith (eds.), Pineridge Press, Swansea, U.K., 669–679 (1983).
17. D. K. Gartling and N. Phan Thien, "A Numerical Simulation of a Plastic Fluid in a Parallel-Plate Plastometer," *Journal of Non-Newtonian Fluid Mechanics*, **14**, 347–360 (1984).
18. R. I. Tanner and J. F. Milthorpe, "Numerical Simulation of the Flow of Fluids with Yield Stresses," in *Proceedings of Third International Conference on Numerical Methods in Laminar and Turbulent Flow*, C. Taylor, C. Johnson, and I. Smith (eds.), Pineridge Press, Swansea, U.K., 680–690 (1983).
19. M. S. Engelman, "Quasi-Newton Methods in Fluid Dynamics," in *The Mathematics of Finite Elements and Applications, IV*, J. R. Whiteman (ed.), Academic Press, London, U.K., 479–487 (1982).
20. C. Truesdell and W. Noll, "The Non-Linear Field Theories of Mechanics," in *Encyclopedia of Physics, Vol III*, S. Flügge (ed.), Springer-Verlag, Berlin, Germany (1965).
21. M. J. Crochet and K. Walters, "Numerical Methods in Non-Newtonian Fluid Mechanics," *Annual Review of Fluid Mechanics*, **15**, 241–260 (1983).
22. A. C. Pipkin and R. I. Tanner, "A Survey of Theory and Experiment in Viscometric Flows of Viscoelastic Liquids," *Mechanics Today, Vol. 1*, S. Nemat-Nasser (ed.), 262–321 (1972).
23. R. I. Tanner, "Recent Progress in Rheology," *Journal of Applied Mechanics*, **105**, 1181–1190 (1983).
24. M. J. Crochet and G. Pilate, "Numerical Study of the Flow of a Fluid of Second Grade in a Square Cavity," *Computers and Fluids*, **3**, 283–291 (1975).
25. K. R. Reddy and R. I. Tanner, "Finite Element Approach to Die-Swell Problems of Non-Newtonian Fluids," *Proceedings of the Sixth Australian Hydraulics and Fluid Mechanics Conference*, Adelaide, Australia, 431–434 (1977).
26. M. A. Mendelson, P. W. Yeh, R. A. Brown, and R. C. Armstrong, "Approximation Error in Finite Element Calculation of Viscoelastic Flow," *Journal of Non-Newtonian Fluid Mechanics*, **10**, 31–54 (1982).
27. M. W. Johnson and D. Segalman, "A Model for Viscoelastic Fluid Behavior Which Allows Non-Affine Deformation," *Journal of Non-Newtonian Fluid Mechanics*, **2**, 255–270 (1977).
28. N. Phan Thien and R. I. Tanner, "A New Constitutive Equation Derived From Network Theory," *Journal of Non-Newtonian Fluid Mechanics*, **2**, 353–365 (1977).
29. J. L. White and A. B. Metzner, "Development of Constitutive Equations for Polymeric Melts and Solutions," *Journal of Applied Polymer Science*, **7**, 1867–1889 (1963).
30. C. J. S. Petrie, "Measures of Deformation and Convected Derivatives," *Journal of Non-Newtonian Fluid Mechanics*, **5**, 147–176 (1979).
31. L. E. Malvern, *Introduction to the Mechanics of a Continuous Medium*, Prentice-Hall, Englewood Cliffs, New Jersey (1969).
32. A. S. Lodge, *Elastic Liquids*, Academic Press, New York (1964).

33. Y. Shimazaki and E. G. Thompson, "Elasto Visco-Plastic Flow With Special Attention to Boundary Conditions," *International Journal for Numerical Methods in Engineering*, **17**, 97–112 (1981).
34. M. Kawahara and N. Takeuchi, "Mixed Finite Element Method for Analysis of Viscoelastic Fluid Flow," *Computers and Fluids*, **5**, 33–45 (1977).
35. M. J. Crochet and M. Bezy, "Numerical Solution for the Flow of Viscoelastic Fluids," *Journal of Non-Newtonian Fluid Mechanics*, **5**, 201–218 (1979).
36. C. J. Coleman, "A Finite Element Routine for Analyzing Non-Newtonian Fluids, Part I: Basic Method and Preliminary Results," *J. of Non-Newtonian Fluid Mechanics*, **7**, 289–301 (1980).
37. P. W. Chang, T. W. Patten, and B. A. Finlayson, "Collocation and Galerkin Finite Element Methods for Viscoelastic Fluid Flow, I. Description of Method and Problems with Fixed Geometries," *Computers and Fluids*, **7**, 267–283 (1979).
38. J. M. Marchal and M. J. Crochet, "Hermitian Finite Elements for Calculating Viscoelastic Flow," *Journal of Non-Newtonian Fluid Mechanics*, **20**, 187–207 (1986).
39. J. M. Marchal and M. J. Crochet, "A New Mixed Finite Element for Calculating Viscoelastic Flow," *Journal of Non-Newtonian Fluid Mechanics*, **26**, 77–114 (1987).
40. R. R. Rao and B. A. Finlayson, "Viscoelastic Flow Simulation Using Cubic Stress Finite Elements," *Journal of Non-Newtonian Fluid Mechanics*, **43**, 61–82 (1992).
41. P. W. Yeh, M. E. Kim, R. C. Armstrong, and R. A. Brown, "Multiple Solutions in the Calculation of Axisymmetric Contraction Flow of an Upper Convected Maxwell Fluid," *Journal of Non-Newtonian Fluid Mechanics*, **16**, 173–194 (1984).
42. D. K. Gartling, "One Dimensional Finite Element Solutions for a Maxwell Fluid," *Journal of Non-Newtonian Fluid Mechanics*, **17**, 203–231 (1985).
43. M. J. Crochet and R. Keunings, "Die Swell of a Maxwell Fluid: Numerical Predictions," *Journal of Non-Newtonian Fluid Mechanics*, **7**, 199–212 (1980).
44. R. Keunings and M. J. Crochet, "Numerical Simulation of the Flow of a Viscoelastic Fluid Through an Abrupt Contraction," *J. of Non-Newtonian Fluid Mechanics*, **14**, 279–299 (1984).
45. P. Gresho, R. Lee, R. Sani, and T. Stullich, "On the Time-Dependent FEM Solution of the Incompressible Navier-Stokes Equations in Two- and Three-Dimensions," in *Recent Advances in Numerical Methods in Fluids*, Vol. 1, Pineridge Press, Swansea, U.K. (1980).
46. S. L. Josse and B. A. Finlayson, "Reflections on the Numerical Viscoelastic Flow Problem," *Journal of Non-Newtonian Fluid Mechanics*, **16**, 13–36 (1984).
47. D. K. Gartling, "A Finite Element Algorithm for Time-Dependent Viscoelastic Flows," Sandia National Laboratories Report, Albuquerque, New Mexico (1988).
48. M. Fortin and A. Fortin, "A New Approach for the FEM Simulation of Viscoelastic Flows," *Journal of Non-Newtonian Fluid Mechanics*, **32**, 295–310 (1989).
49. R. C. King, M. R. Apelian, R. C. Armstrong, and R. A. Brown, "Numerically Stable Finite Element Techniques for Viscoelastic Calculations in Smooth and Singular Geometries," *Journal of Non-Newtonian Fluid Mechanics*, **29**, 147–216 (1988).
50. D. D. Joseph, M. Renardy, and J. C. Saut, "Hyperbolicity and Change of Type in the Flow of Viscoelastic Fluids," *Archive of Rational Mechanics and Analysis*, **87**, 213–251 (1985).
51. D. Rajagopalan, R. C. Armstrong, and R. A. Brown, "Finite Element Methods for Calculation of Steady, Viscoelastic Flow Using Constitutive Equations with a Newtonian Viscosity," *Journal of Non-Newtonian Fluid Mechanics*, **36**, 159–192 (1990).
52. R. Guenette and M. Fortin, "A New Mixed Finite Element Method for Computing Viscoelastic Flows," *Journal of Non-Newtonian Fluid Mechanics*, **60**, 27–52 (1995).
53. J. Sun, N. Phan Thien, and R. I. Tanner, "An Adaptive Viscoelastic Stress Splitting Scheme and Its Applications: AVSS/SI and AVSS/SUPG," *Journal of Non-Newtonian Fluid Mechanics*, **65**, 75–91 (1996).
54. J. Sun, M. D. Smith, R. C. Armstrong, and R. A. Brown, "Finite Element Method for Viscoelastic Flows Based on the Discrete Adaptive Viscoelastic Stress Splitting and the Discontinuous Galerkin Method: DAVSS-G/DG," *Journal of Non-Newtonian Fluid Mechanics*, **86**, 281–307 (1999).

55. P. Lesaint and P. A. Raviart, "On a Finite Element Method for Solving the Neutron Transport Equation," in *Mathematical Aspects of Finite Elements in Partial Differential Equations*, C. de Boor (Ed.), Academic Press, New York (1974).
56. M. Viriyayuthakorn and B. Caswell, "Finite Element Simulation of Viscoelastic Flow," *Journal of Non-Newtonian Fluid Mechanics*, **6**, 245–267 (1980).
57. S. Dupont, J. M. Marchal, and M. J. Crochet, "Finite Element Simulation of Viscoelastic Fluids of the Integral Type," *Journal of Non-Newtonian Fluid Mechanics*, **17**, 157–183 (1985).
58. B. Bernstein, M. K. Kadivar, and D. S. Malkus, "Steady Flow of Memory Fluids with Finite Elements: Two Test Problems," *Computer Methods in Applied Mechanics and Engineering*, **27**, 279–302 (1981).
59. R. C. Armstrong, R. A Brown, and B. Caswell (eds.), "Papers From the Third International Workshop on Numerical Simulation of Viscoelastic Flows," *Journal of Non-Newtonian Fluid Mechanics*, **16** (1984).
60. M. J. Crochet, "Numerical Simulation of Viscoelastic Flow: A Review," *Rubber Reviews*, **62**, 426–455 (1989).
61. R. Keunings, "Progress and Challenges in Computational Rheology," *Rheologica Acta*, **29**, 556–570 (1990).
62. F. P. T. Baaijens, "Mixed Finite Element Methods for Viscoelastic Flow Analysis: A Review," *Journal of Non-Newtonian Fluid Mechanics*, **79**, 361–385 (1998).
63. M. Renardy, "Current Issues in Non-Newtonian Flows: A Mathematical Perspective," *Journal of Non-Newtonian Fluid Mechanics*, **90**, 243–259 (2000).
64. D. Winterscheidt and K. S. Surana, " p -Version Least Squares Finite Element Formulation for Convection Diffusion Problems," *International Journal for Numerical Methods in Engineering*, **36**, 111–133 (1993).
65. D. Winterscheidt and K. S. Surana, " p -Version Least Squares Finite Element Formulation for Burgers Equation," *International Journal for Numerical Methods in Engineering*, **36**, 3629–3646 (1993).
66. K. S. Surana, P. Gupta, P. W. Tenpas, and J. N. Reddy, " h, p, k Least Squares Finite Element Processes for 1-D Helmholtz Equation," *International Journal for Computational Methods in Engineering Science and Mechanics*, **7**(4), 263–291 (2006).
67. K. S. Surana, A. Mohammed, J. N. Reddy, and P. W. Tenpas, " k -Version of Finite Element Method in 2-D Polymer Flows: Oldroyd-B Constitutive Model," *International Journal of Numerical Methods in Fluids*, **52**(2), 119–162 (2006).
68. K. S. Surana, S. Allu, P. W. Tenpas, and J. N. Reddy, " k -Version Finite Element Method in Gas Dynamics: Higher Order Global Differentiability Numerical Solutions," *International Journal for Numerical Methods in Engineering*, **69**(6), 1109–1157 (2007).
69. K. S. Surana, M. K. EngelKemier, J. N. Reddy, and P. W. TenPas, " k -version Least Squares Finite Element Processes for 2-D Generalized Newtonian Fluid Flows," *International Journal of Computational Methods in Engineering Science and Mechanics*, **8**, 243–261 (2007).
70. K. S. Surana, P. Gupta, and J. N. Reddy, "Galerkin and Least-Squares Finite Element Processes for 2-D Helmholtz Equation in h, p, k Framework," *International Journal of Computational Methods in Engineering Science and Mechanics*, **8**(5), 341–361 (2007).
71. K. S. Surana, S. Bhola, J. N. Reddy, and P. W. TenPas, " k -Version of Finite Element Method in 2D-Polymer Flows: Upper Convected Maxwell Model," *Computers and Structures*, **86** (17-18), 1782–1808 (2008).
72. K. S. Surana, S. Allu, J. N. Reddy, and P. W. TenPas, "Least Squares Finite Element Processes in h, p, k Mathematical and Computational Framework for a Non-Linear Conservation Law," *International Journal for Numerical Methods in Fluids*, **57**(10), 1545–1568 (2008).
73. K. S. Surana, K. M. Deshpande, A. Romkes, and J. N. Reddy, "Computations of Numerical Solutions in Polymer Flows Using Giesekus Constitutive Model in the h, p, k Framework with Variationally Consistent Integral Forms," *International Journal of Computational Methods in Engineering Science and Mechanics*, to appear.
74. N. Phan Thien, "Coaxial-Disk Flow of an Oldroyd-B Fluid: Exact Solution and Stability," *Journal of Non-Newtonian Fluid Mechanics*, **13**, 325–340 (1983).

75. N. Phan Thien and W. Walsh, "Squeeze-Film Flow of an Oldroyd-B Fluid: Similarity Solution and Limiting Weissenberg Number," *Journal of Applied Mathematics and Physics (ZAMP)*, **35**, 747–759 (1984).
76. N. Phan Thien, "Cone-and-Plate Flow of the Oldroyd-B Fluid is Unstable," *Journal of Non-Newtonian Fluid Mechanics*, **17**, 37–44 (1985).
77. R. Keunings, "Mesh Refinement Analysis of the Flow of a Maxwell Fluid Through an Abrupt Contraction," *Proceedings of Fourth International Conference on Numerical Methods in Laminar and Turbulent Flow*, C. Taylor et al. (eds.), Pineridge Press, Swansea, U.K. (1985).
78. B. Debbaut and M. J. Crochet, "Further Results on the Flow of a Viscoelastic Fluid Through an Abrupt Contraction," *Journal of Non-Newtonian Fluid Mechanics*, **20**, 173–185 (1986).
79. J. V. Lawler, S. J. Muller, R. A. Brown, and R. C. Armstrong, "Laser Doppler Velocimetry, Measurements of Velocity Fields and Transitions in Viscoelastic Fluids," *Journal of Non-Newtonian Fluid Mechanics*, **20**, 51 (1986).
80. F. Dupret, J. M. Marchal, and M. J. Crochet, "On the Consequence of Discretization Errors in the Numerical Calculation of Viscoelastic Flow," *Journal of Non-Newtonian Fluid Mechanics*, **18**, 173–186 (1985).
81. M. Fortin and R. Pierre, "On the Convergence of the Mixed Method of Crochet and Marchal for Viscoelastic Flows," *Computer Methods in Applied Mechanics and Engineering*, **7**, 1035–1052 (1987).
82. N. Phan Thien, "Coaxial-Disk Flow and Flow About a Rotating Disk of a Maxwell Fluid," *Journal of Fluid Mechanics*, **128**, 427–442 (1983).
83. N. Phan Thien and R. I. Tanner, "Viscoelastic Squeeze-Film Flows – Maxwell Fluids," *Journal of Fluid Mechanics*, **129**, 265–281 (1983).
84. R. S. Rivlin, "Run-Up and Decay of Plane Poiseuille Flow," *Journal of Non-Newtonian Fluid Mechanics*, **14**, 203–217 (1984).
85. M. Iga and J. N. Reddy, "Penalty Finite Element Analysis of Free Surface Flows of Power-Law Fluids," *International Journal of Non-Linear Mechanics*, **24**(5), 383–399 (1989).
86. M. P. Reddy and J. N. Reddy, "Finite Element Analysis of Flows of Non-Newtonian Fluids in Three-Dimensional Enclosures," *International Journal of Non-Linear Mechanics*, **27**(1), 9–26 (1992).
87. M. P. Reddy and J. N. Reddy, "Numerical Simulation of Forming Processes Using a Coupled Fluid Flow and Heat Transfer Model," *International Journal for Numerical Methods in Engineering*, **35**, 807–833 (1992).
88. R. I. Tanner, "Extrudate Swell," in *Computational Analysis of Polymer Processing*, J. R. A. Pearson and S. M. Richardson (eds.), Elsevier, London, U.K., 63–91 (1983).
89. M. J. Crochet and R. Keunings, "On Numerical Die Swell," *Journal of Non-Newtonian Fluid Mechanics*, **10**, 85–94 (1982).
90. M. J. Crochet and R. Keunings, "Finite Element Analysis of Die-Swell of a Highly Elastic Fluid," *Journal of Non-Newtonian Fluid Mechanics*, **10**, 339–356 (1982).
91. J. N. Reddy and V. A. Padhye, "A Penalty Finite Element Model for Axisymmetric Flows of Non-Newtonian Fluids," *Numerical Methods in Partial Differential Equations*, **4**, 33–56 (1988).
92. R. R. Rao, "Adaptive Finite Element Analysis of Non-Newtonian Flow," Ph.D. Dissertation, Department of Chemical Engineering, University of Washington, Seattle, Washington (1990).

Multiphysics Problems

7.1 Introduction

Coupled problems in applied mechanics are generally defined as those requiring the solution of more than one physical process for adequate representation of the overall system. In reality, most engineering problems fall into this category, although it is still a common practice to perform single-physics analysis for many applications. Coupled phenomena are very prevalent in the areas of fluid mechanics and heat transfer as is evident from some of the topics from previous chapters. In Chapter 3, coupled problems involving heat conduction and radiation, and heat conduction and chemical reaction were considered. At the beginning of Chapter 5 it was noted that convective heat transfer was a coupled problem since it involved two different physical phenomena, namely, fluid mechanics and heat transfer within a fluid. Likewise, the conjugate problem of convective heat transfer in a fluid adjacent to a heat conducting solid is a type of coupled problem. In the present chapter we are going to revisit coupled problems, though the emphasis will be on problems involving more than one discipline or branch of mechanics. Specifically, the finite element solution of solid mechanics and electromagnetics problems will first be presented. Subsequent sections will describe how these types of solutions may be coupled to fluid mechanics and heat transfer simulations to provide a more complete analysis.

7.2 Coupled Boundary Value Problems

Boundary value problems describing different types of mechanics may be coupled through a variety of mechanisms and with varying degrees of interaction. Both of these characteristics are difficult to generalize and quantify, and lead to a certain vagueness when discussing coupled problems in generic terms. Before getting to the specific cases of interest here, it is worthwhile to set some terminology and outline some of the complexities that may occur.

The partial differential equations describing different phenomena are coupled when any terms in either equation are functions of the dependent variable or its derivatives from the other equation. This functional dependence may occur directly in source or volume terms, material coefficients, and/or boundary conditions. The dependent variables from one equation may also act more indirectly on the second equation by causing alterations in the geometry of the problem and changes to the temporal behavior of the problem. The degree to which one equation is coupled to

a second equation is particularly difficult to define. The terms “weak” and “strong” coupling are often used without precise definitions. In the present case, a “weakly” coupled problem is defined as one in which the transfer of dependent variable data between equations need not occur at every solution step. In essence, the influence of one physical process on the other is sufficiently mild that only periodic updating is required for an accurate representation of the overall system response. For a weakly coupled problem the data transfer may be bidirectional or unidirectional, i.e., there is no mandate that the processes be equally influential. The definition of a “strongly” coupled problem is obvious from the previous definition. If data must be transferred or shared between equations at each step of the solution to maintain accuracy of overall simulation, the problem is defined as strongly coupled. These definitions are not rigorous but do allow some general classifications for coupled problems.

The degree of coupling will also influence the style and choice of computational algorithms used to solve each equation set and the interaction between discretized equations. In general, our preference is to solve strongly coupled physical processes as fully coupled equation sets as this usually provides the most robust and strongly convergent algorithm. The natural convection problem of Chapter 5 is an example of this type of coupled problem. The fully coupled approach is not, however, always feasible. Both physical and numerical characteristics of the particular coupled problem may render full coupling computationally inefficient or impractical. Large disparities in length and time scales between physical processes are two characteristics that may influence the strong coupling algorithm. The solution procedure for weakly coupled physical processes follows from its definition and is obviously some type of cyclic algorithm with separate solution methods for each equation set intertwined with periodic exchanges of data. The frequency and timing of data updates are very problem dependent and one of the characteristics that makes general coupling methods and software difficult to develop.

7.3 Fluid Mechanics and Heat Transfer

7.3.1 Introduction

To proceed with the coupled problem discussion it is necessary to become specific with regard to physical processes and coupling mechanisms. In the present section the fluid/thermal problem will again be outlined with particular attention paid to possible dependencies on the solid mechanics and electromagnetics problems to be considered subsequently.

7.3.2 Continuum Equations

The boundary value problem for the nonisothermal flow of a viscous, incompressible fluid is described by the standard conservation relations for mass, momentum, and energy. For a Cartesian coordinate system using the Eulerian description, these relations are summarized here.

Conservation of Mass

$$\frac{\partial v_i}{\partial x_i} = 0 \quad (7.3.1)$$

Conservation of Momentum

$$\rho_0 \left(\frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} \right) - \frac{\partial}{\partial x_j} \left[-P \delta_{ij} + \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \right] + \rho_0 g_i \beta (T - T_0) = 0 \quad (7.3.2)$$

Conservation of Energy

$$\rho_0 C_v \left(\frac{\partial T}{\partial t} + v_j \frac{\partial T}{\partial x_j} \right) - \frac{\partial}{\partial x_i} \left(k_{ij} \frac{\partial T}{\partial x_j} \right) - Q - \Phi = 0 \quad (7.3.3)$$

The parameters and symbols used in (7.3.1)–(7.3.3) are standard and the same as used in Chapter 5. Note that the porous flow problem could also be considered here as part of the coupling problem with little variation in the discussion. The boundary conditions for the nonisothermal flow are

$$v_i = f_i^v(s_k, t) \quad \text{on } \Gamma_v \quad (7.3.4a)$$

$$\mathcal{T}_i \equiv \sigma_{ij}(s_k, t) n_j(s_k) = f_i^T(s_k, t) \quad \text{on } \Gamma_T \quad (7.3.4b)$$

for the fluid mechanics part of the problem, and

$$T = f^T(s_k, t) \quad \text{on } \Gamma_T \quad (7.3.5a)$$

$$-\left(k_{ij} \frac{\partial T}{\partial x_j} \right) n_i \equiv q_i n_i = q_c + q_r + q_a = f^q(s_k, t) \quad \text{on } \Gamma_q \quad (7.3.5b)$$

for the heat transfer part of the problem.

When discussing coupling, solid body thermal analysis will be considered distinct from the fluid mechanics problem since the equation system is substantially different as are the coupling interactions. In the no flow case, Eqs. (7.3.1) and (7.3.2) are neglected and Eq. (7.3.3) reduces to the simple heat conduction problem

$$\rho C \frac{\partial T}{\partial t} = \frac{\partial}{\partial x_i} \left(k_{ij} \frac{\partial T}{\partial x_j} \right) + Q \quad (7.3.6)$$

with boundary conditions as given in Eq. (7.3.5). The addition of radiation and/or chemical reaction to the problem in Eq. (7.3.6) can be anticipated but will not be described in equation form. The radiation and chemical reaction problems were described in detail in Chapter 3.

The general thermal problem described above can be coupled to a solid mechanics problem via several mechanisms. Deformation of the material leads to a new spatial orientation (new coordinates) and new density distribution. The deformation may also produce new surface orientations that influence radiation and/or surface exposures that lead to changes in thermal boundary conditions, e.g., contact. Mechanical failure can also lead to new surface definitions. Dissipation mechanisms during deformation may produce a significant volumetric heat source for the energy balance. Surface tractions in the mechanical problem will influence thermal contact at material interfaces and frictional heating models. Finally, reaction rates may be influenced by the material stress state (pressure).

The usual sources of coupling for a thermal problem to an electromagnetics problem are somewhat fewer in number. These include volumetric heat sources due to Joule heating and the possibility of thermal property dependence on electric and magnetic field strength.

7.3.3 Finite Element Models

The finite element method for the boundary value problems described above have been presented in great detail in previous chapters. For completeness and ease of reference the standard matrix form of the discretized equations will be presented here. For the heat conduction problem, the finite element equations are given by Eqs. (3.6.3) and (3.6.4) as

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} = \hat{\mathbf{F}} \quad (7.3.7)$$

with

$$\hat{\mathbf{K}} = \mathbf{K} + \mathbf{C} + \mathbf{R} \quad (7.3.8a)$$

$$\hat{\mathbf{F}} = \mathbf{Q} - \mathbf{q}_a + \mathbf{F}_{hc} + \mathbf{F}_{hr} \quad (7.3.8b)$$

In the standard uncoupled problem, the matrices \mathbf{M} , $\hat{\mathbf{K}}$, and $\hat{\mathbf{F}}$ are functions of temperature and/or time. The coupled problem will introduce other dependencies which will be discussed in a later section.

In the case of the nonisothermal, viscous flow problem, the finite element equations are given by Eqs. (5.3.10)–(5.3.12), which are

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (7.3.9)$$

$$\mathbf{M}\dot{\mathbf{v}} + \mathbf{Cv} + \mathbf{Kv} - \mathbf{QP} + \mathbf{BT} = \mathbf{F} \quad (7.3.10)$$

$$\mathbf{N}\dot{\mathbf{T}} + \mathbf{DT} + \mathbf{LT} = \mathbf{G} \quad (7.3.11)$$

and represent the continuity, momentum, and energy equations. Like the heat conduction problem the dependencies of the various terms in (7.3.9)–(7.3.11) will be described in a later section for the various types of coupling.

7.4 Solid Mechanics

7.4.1 Introduction

Computational solid mechanics is a very broad discipline with a well-developed finite element base and widespread use in engineering analysis. Solid mechanics applications that will be of interest in the context of coupled problems are limited to large deformation and inelastic response of solid materials. Dynamics problems are also of interest but are not considered here because of the limited coupling with the diffusion dominated thermal problems and viscous incompressible flow problems that are the main focus of the book. The coverage of even this limited area in solid mechanics must, by necessity, be rather superficial with little explanation of important aspects, such as constitutive behavior and finite element models and solution methods. The interested reader may consult any of several comprehensive texts dedicated to finite elements in solid mechanics [1–8]. The details of theoretical concepts and derivations from solid mechanics may be found in continuum mechanics or elasticity books (see [9–13] and references therein).

7.4.2 Kinematics of Deformation

The starting point for describing the equations of solid mechanics is usually kinematics, which deals with changes in geometry (i.e., deformation) of the body under external loads. Various measures of deformation are introduced in the form of strain tensor and deformation rate tensor. The geometric changes are accompanied by stresses in the body. Suitable measures of stress are introduced and equations of motion are derived using the principles of conservation of linear and angular momenta, as discussed in Section 7.4.3.

The simultaneous positions occupied in space by all material points of the solid body at different instants of time are called configurations. In the study of solid bodies, it is usual to start with a reference configuration, often the undeformed configuration, of a body and describe its subsequent motion and deformation. Suppose that the continuum initially occupies a configuration in which a particle X occupies the position \mathbf{X} , referred to a rectangular Cartesian system (X_1, X_2, X_3) , called the *material coordinates*. After the application of the loads, the continuum changes its geometric shape and thus assumes a new configuration in which the particle X occupies the position \mathbf{x} , as shown in Figure 7.4.1. The mapping χ from the reference configuration to the deformed configuration, called the *deformation mapping*, takes the position vector \mathbf{X} from the reference configuration and places the same point in the deformed configuration as

$$\mathbf{x} = \chi(\mathbf{X}, t) \quad (7.4.1)$$

7.4.2.1 Descriptions of motion

The mathematical description of the deformation of a continuous body follows one of the two approaches: (1) spatial description and (2) material description. The spatial description is known as the *Eulerian description* and the material description is also known as the *Lagrangian description*. In the spatial description, the motion is referred to the current configuration occupied by the continuum, and a typical field variable ϕ is described with respect to the current position \mathbf{x} in space, currently occupied by material particle X :

$$\phi = \phi(\mathbf{x}, t), \quad \mathbf{X} = \mathbf{X}(\mathbf{x}, t) \quad (7.4.2)$$

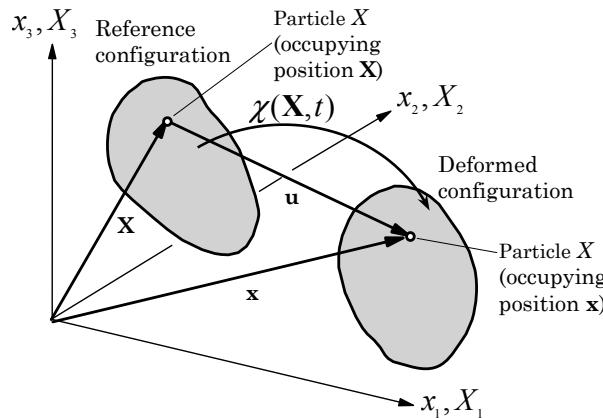


Figure 7.4.1: Reference and deformed configurations of a body.

The coordinates (\mathbf{x}) are termed the *spatial coordinates*. For a fixed value of \mathbf{x} , $\phi(\mathbf{x}, t)$ gives the value of ϕ associated with a fixed point \mathbf{x} in space, which will be the value of ϕ associated with different material points at different times, because different material points occupy the position \mathbf{x} at different times. Thus, a change in time t implies that a different value ϕ is observed at the *same* spatial location \mathbf{x} , now probably occupied by a different material particle X . Hence, attention is focused on a spatial position \mathbf{x} .

In the Lagrangian description, the current coordinates \mathbf{x} are expressed in terms of the reference coordinates \mathbf{X}

$$\mathbf{x} = \chi(\mathbf{X}, t), \quad \chi(\mathbf{X}, 0) = \mathbf{X} \quad (7.4.3)$$

and the variation of a typical variable ϕ over the body is described with respect to the material coordinates \mathbf{X} and time t :

$$\phi = \phi(\mathbf{X}, t) \quad (7.4.4)$$

For a fixed value of \mathbf{X} from the reference configuration, $\phi(\mathbf{X}, t)$ gives the value of ϕ at time t associated with the fixed material point X whose position in the reference configuration is \mathbf{X} , as shown in Fig. 7.4.2. Thus, a change in time t implies that the *same* material particle X occupying position \mathbf{X} has a different value ϕ . Thus the attention is focused on the material particles X of the continuum.

The Eulerian description is the preferred description for the study of motion of fluids because the configuration is known and remains unchanged, and changes in the fluid velocities, pressure, density and so on are determined, as discussed in much of this book. In the study of solid bodies, the Eulerian description is less useful since the deformed configuration is unknown.

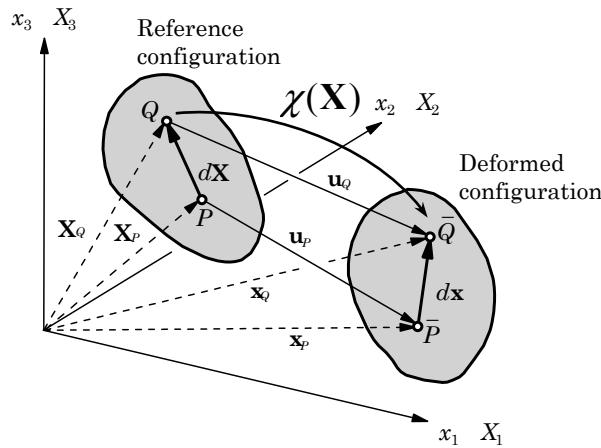


Figure 7.4.2: Reference configuration and deformed configurations in the material description.

7.4.2.2 Displacement vector

The word *deformation* refers to relative displacements and changes in the geometry experienced by the body under the influence of a force system. The displacement of the particle X is given by

$$\mathbf{u} = \mathbf{x} - \mathbf{X} \quad (7.4.5)$$

In the Lagrangian description, the displacements are expressed in terms of the material coordinates X_i

$$\mathbf{u}(\mathbf{X}, t) = \mathbf{x}(\mathbf{X}, t) - \mathbf{X}. \quad (7.4.6)$$

In the material description, the velocity and acceleration vectors are simply given by [note that $\mathbf{u} = \mathbf{u}(\mathbf{X}, t)$]

$$\mathbf{v}(\mathbf{X}, t) = \frac{\partial \mathbf{u}}{\partial t}, \quad \mathbf{a}(\mathbf{X}, t) = \frac{\partial \mathbf{v}}{\partial t} = \frac{\partial^2 \mathbf{u}}{\partial t^2} \quad (7.4.7)$$

A rigid-body motion is one in which all material particles of the body undergo the same linear and angular displacements. On the other hand, a deformable body is one in which the material particles can move relative to each other. Then the deformation of a body can be determined only by considering the change of distance between any two arbitrary but infinitesimally close points of the body.

7.4.2.3 Deformation gradient tensor

One of the key quantities in deformation analysis is the *deformation gradient* relative to the reference configuration, denoted \mathbf{F} , which gives the relationship of a material line $d\mathbf{X}$ before deformation to the line $d\mathbf{x}$ (consisting of the same material as $d\mathbf{X}$) after deformation. It is defined as [9]

$$d\mathbf{x} = \mathbf{F} \cdot d\mathbf{X} = d\mathbf{X} \cdot \mathbf{F}^T \quad (7.4.8a)$$

$$\mathbf{F} = \left(\frac{\partial \chi}{\partial \mathbf{X}} \right)^T = \left(\frac{\partial \mathbf{x}}{\partial \mathbf{X}} \right)^T \equiv (\nabla_0 \mathbf{x})^T \quad (7.4.8b)$$

and ∇_0 is the gradient operator with respect to \mathbf{X} . By definition, \mathbf{F} is a second-order tensor. The inverse relations are given by

$$d\mathbf{X} = \mathbf{F}^{-1} \cdot d\mathbf{x} = d\mathbf{x} \cdot \mathbf{F}^{-T}, \quad \mathbf{F}^{-T} = \frac{\partial \mathbf{X}}{\partial \mathbf{x}} \equiv \nabla \mathbf{x} \quad (7.4.9)$$

and ∇ is the gradient operator with respect to \mathbf{x} . In indicial notation, Eqs. (7.4.8b) and (7.4.9) can be written as

$$\mathbf{F} = F_{iJ} \hat{\mathbf{e}}_i \hat{\mathbf{E}}_J, \quad F_{iJ} = \frac{\partial x_i}{\partial X_J} \quad \text{and} \quad \mathbf{F}^{-1} = F_{Ji}^{-1} \hat{\mathbf{E}}_J \hat{\mathbf{e}}_i, \quad F_{Ji}^{-1} = \frac{\partial X_J}{\partial x_i} \quad (7.4.10)$$

The determinant of \mathbf{F} is called the *Jacobian of the motion*, and it is denoted by $J = \det \mathbf{F}$. The equation $\mathbf{F} \cdot d\mathbf{X} = 0$ for $d\mathbf{X} \neq 0$ implies that a material line in the reference configuration is reduced to zero by the deformation. Since this is physically not realistic, we conclude that $\mathbf{F} \cdot d\mathbf{X} \neq 0$ for $d\mathbf{X} \neq 0$. That is, \mathbf{F} is a non-singular tensor, $J \neq 0$. Hence, \mathbf{F} has an inverse \mathbf{F}^{-1} .

7.4.2.4 Green strain tensor

The geometric changes that a solid continuum experiences can be measured in a number of ways. Here, we discuss a general measure of deformation of a solid body. Consider two material particles P and Q in the neighborhood of each other, separated by $d\mathbf{X}$ in the reference configuration. In the current (deformed) configuration the material points P and Q occupy positions \bar{P} and \bar{Q} , and they are separated by $d\mathbf{x}$, as shown in Figure 7.4.2. The change in the squared lengths that occurs as a body deforms from the reference configuration to the current configuration can be expressed relative to the original length as

$$(ds)^2 - (dS)^2 = d\mathbf{x} \cdot d\mathbf{x} - d\mathbf{X} \cdot d\mathbf{X} = d\mathbf{X} \cdot (\mathbf{F}^T \cdot \mathbf{F}) \cdot d\mathbf{X} - d\mathbf{X} \cdot d\mathbf{X} \equiv 2 d\mathbf{X} \cdot \mathbf{E} \cdot d\mathbf{X} \quad (7.4.11)$$

where

$$\begin{aligned} \mathbf{E} &= \frac{1}{2} (\mathbf{F}^T \cdot \mathbf{F} - \mathbf{I}) = \frac{1}{2} [(\mathbf{I} + \nabla_0 \mathbf{u}) \cdot (\mathbf{I} + \nabla_0 \mathbf{u})^T - \mathbf{I}] \\ &= \frac{1}{2} [\nabla_0 \mathbf{u} + (\nabla_0 \mathbf{u})^T + (\nabla_0 \mathbf{u}) \cdot (\nabla_0 \mathbf{u})^T] \end{aligned} \quad (7.4.12)$$

By definition, \mathbf{E} is a symmetric second-order tensor, called the *Green strain tensor*. In index notation, the rectangular Cartesian components of \mathbf{E} are given by

$$E_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_k}{\partial X_i} \frac{\partial u_k}{\partial X_j} \right) \quad (7.4.13)$$

The components of \mathbf{E} in other coordinate systems can be obtained using the definition (7.4.13) and writing ∇_0 and \mathbf{u} in that coordinate system.

The Green strain tensor can be expressed as the sum of linear and nonlinear parts

$$\mathbf{E} = \mathbf{e}^1 + \mathbf{e}^2, \quad E_{ij} = e_{ij}^1 + e_{ij}^2 \quad (7.4.14)$$

where

$$\mathbf{e}^1 = \frac{1}{2} [\nabla_0 \mathbf{u} + (\nabla_0 \mathbf{u})^T], \quad \mathbf{e}^2 = \frac{1}{2} (\nabla_0 \mathbf{u}) \cdot (\nabla_0 \mathbf{u})^T \quad (7.4.15a)$$

$$e_{ij}^1 = \frac{1}{2} \left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} \right), \quad e_{ij}^2 = \frac{1}{2} \frac{\partial u_k}{\partial X_i} \frac{\partial u_k}{\partial X_j} \quad (7.4.15b)$$

Other strain measures can be defined. For example, the change in the squared lengths that occurs as the body deforms from the initial to the current configuration can be expressed relative to the current length as

$$(ds)^2 - (dS)^2 = 2 d\mathbf{x} \cdot \mathbf{e} \cdot d\mathbf{x} \quad (7.4.16)$$

where \mathbf{e} is called the *Euler strain tensor*, which can be expressed as

$$\mathbf{e} = \frac{1}{2} (\mathbf{I} - \mathbf{F}^{-T} \cdot \mathbf{F}^{-1}) = \frac{1}{2} (\mathbf{I} - \mathbf{B}^{-1}) \quad (7.4.17)$$

where $\mathbf{B} = \mathbf{F} \cdot \mathbf{F}^T$ is the *Cauchy strain tensor*, and its inverse is called the *left Cauchy–Green or Finger tensor*.

The choice of a strain measure is mainly one of convenience for the type of constitutive model used. When path-dependent material response is important, it is usual to formulate the equations in terms of strain rates and velocities rather than strains and displacements. Because we will not be exploring constitutive relations in detail, the rate-dependent definitions will not be discussed here.

7.4.3 Kinetics

7.4.3.1 Stress measures

The Cauchy stress tensor σ used in fluid mechanics is defined as the force per unit area in the current configuration. The equations of motion or equilibrium must be derived for the deformed configuration. However, since the geometry of the deformed configuration is not known, the equations must be written in terms of the known reference configuration. In doing so we introduce the second Piola–Kirchhoff stress tensor $\tilde{\sigma}$, which characterizes the current force in the deformed configuration but transformed to undeformed configuration, and measured per unit area in the undeformed configuration. The second Piola–Kirchhoff stress tensor is energetically conjugate to the rate of Green strain tensor in the sense that the rate of internal work done (power) in a continuous medium is given by

$$W = \frac{1}{2} \int_v \sigma : \mathbf{D} d\mathbf{x} = \frac{1}{2} \int_V \tilde{\sigma} : \dot{\mathbf{E}} d\mathbf{X} \quad (7.4.18)$$

where σ is the Cauchy stress tensor and \mathbf{D} is the symmetric part of the *velocity gradient tensor*. This emerges in a natural way as we transform volumes and areas from the deformed configuration \mathcal{C} to undeformed configuration \mathcal{C}^0 (see Reddy [8,9] and Bonet and Wood [10] for details).

The second Piola–Kirchhoff stress tensor $\tilde{\sigma}$ is related to the Cauchy stress σ tensor by

$$\tilde{\sigma} = \frac{\rho_0}{\rho} \mathbf{F}^{-1} \sigma \mathbf{F}^{-T} \quad (7.4.19a)$$

$$\tilde{\sigma}_{ij} = \frac{\rho_0}{\rho} \frac{\partial X_i}{\partial x_k} \sigma_{kl} \frac{\partial X_j}{\partial x_l} \quad (7.4.19b)$$

An inverse relation exists where the Cauchy stress is a function of the Piola–Kirchhoff stress

$$\sigma = \frac{\rho}{\rho_0} \mathbf{F} \tilde{\sigma} \mathbf{F}^T \quad (7.4.20a)$$

$$\sigma_{ij} = \frac{\rho}{\rho_0} \frac{\partial x_i}{\partial X_k} \tilde{\sigma}_{kl} \frac{\partial x_j}{\partial X_l} \quad (7.4.20b)$$

7.4.3.2 Equilibrium statements

The equations of equilibrium for a solid are provided by the principle of conservation of linear momentum and they are the same as described in the previous chapters

$$\nabla \cdot \sigma + \rho \mathbf{b} = \mathbf{0}, \quad \frac{\partial \sigma_{ij}}{\partial x_j} + \rho b_i = 0 \quad (7.4.21)$$

where σ is the Cauchy stress tensor, ρ is the density and \mathbf{b} is the body force vector.

The finite element models in solid mechanics are often based on principles of virtual work and the principle of minimum total potential energy [9,13], which are equivalent to conservation of linear momentum. The virtual work statement in the deformed configuration is transformed to one on the undeformed configuration using the equivalence in Eq. (7.4.18). Consequently, the virtual work statement is cast in terms of the second Piola–Kirchhoff stress tensor and the Green strain tensor. We shall return to the discussion of the virtual work statement in Section 7.4.5.

A statement of mass conservation is also required and in material coordinates is given by

$$\rho \mathbf{J} = \rho \left| \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \right| = \rho_0 \quad (7.4.22)$$

which is valid for each material point. In Eq. (7.4.22) the current density is ρ , the initial density is ρ_0 and \mathbf{J} is the determinant of the Jacobian and specifies the ratio of the current configuration to the initial state. The mass conservation equation need not be enforced explicitly in most Lagrangian methods but is used to account for density changes in the material.

Coupling of the solid mechanics problem to a thermal analysis may occur through several phenomena and requires the addition of the energy equation to the equations of equilibrium. Temperature- and/or species-dependent material properties (constitutive parameters) are a common occurrence. Also, the material state, such as the extent of reaction, decomposition or gas fraction, and material addition or removal due to a thermal process, will influence the mechanical response. Changes in mechanical boundary conditions, e.g., pressure loading, may occur due to heat transfer. Mechanical response may be influenced by electromagnetics primarily through Lorentz body forces (magnetic pressures) and field-dependent constitutive response, as illustrated by piezoelectric materials.

7.4.4 Constitutive Relations

Constitutive relations for the stress-strain response of a solid material are quite numerous and range in complexity from simple, linear models to multi-variable, path-dependent, nonlinear models. Here, we will limit the outline to linear and nonlinear elastic constitutive relations and leave the complexities of elastic-plastic, creep, and viscoelastic behavior to the more specialized sources [3,9–12].

A standard elastic constitutive relation (generalized Hooke's law) that is valid for large deformation can be written as

$$\tilde{\sigma} = \mathbf{C} \mathbf{E}, \quad \tilde{\sigma}_{ij} = C_{ijkl} E_{kl} \quad (7.4.23)$$

where $\tilde{\sigma}$ is the second Piola–Kirchhoff stress, \mathbf{E} is the Green strain tensor, \mathbf{C} is the elasticity tensor.

For small strains but large displacements and moderate rotations, the elasticity tensor \mathbf{C} is assumed to be constant. For an isotropic material it can be written in its usual form involving the Lamé constants λ and μ

$$C_{ijkl} = \lambda \delta_{ij} \delta_{kl} + \mu (\delta_{ik} \delta_{jl} + \delta_{il} \delta_{jk}) \quad (7.4.24)$$

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)} , \quad \mu = \frac{E}{2(1+\nu)} \quad (7.4.25)$$

where E is the modulus of elasticity and ν is Poisson's ratio.

$$\sigma = \mathbf{C} \mathbf{e}^1 \quad \text{or} \quad \sigma_{ij} = C_{ijkl} e_{kl}^1 = \lambda e_{kk}^1 \delta_{ij} + 2\mu e_{ij}^1 \quad (7.4.26)$$

When strains are not small, the relation in (7.4.23) can still be used though the elasticity tensor \mathbf{C} is no longer a constant. If the tensor \mathbf{C} depends on the strains, the functional dependence is usually expressed in terms of the invariants of the strain tensor. This type of material is usually termed hyperelastic and the elasticity tensor is most often derived from a strain energy functional. Another nonlinear elastic material model defines the elasticity tensor to be a function of a number of variables including the strain, stress, load, damage, and so on. This type of model is termed hypoelastic and is usually written in an incremental form. Note that the above constitutive relations can be augmented with a term that produces a thermal stress. This additional effect is quite common and leads directly to coupling with the energy equation.

7.4.5 Boundary Conditions

The boundary conditions for the momentum equation in (7.4.9) are standard and consist of specified displacements on part of the boundary

$$u_i = f_i^u(s_k, t) \quad \text{on} \quad \Gamma_u \quad (7.4.27)$$

and specified tractions on the remainder of the boundary

$$\tau_{ij} n_j = f_i^\tau(s_k, t) \quad \text{on} \quad \Gamma_\tau \quad (7.4.28)$$

The displacement conditions are often homogeneous and may be used to eliminate the rigid body motions from the deforming geometry. Specified displacements or velocities are also used to define the boundary motion while the region undergoes deformation. Applied tractions, often in the form of normal pressures, provide another method for loading the structure. The contact boundary condition, either frictionless or with friction, is in widespread use but is difficult to implement. In quasi-static problems it is often difficult to uniquely define the geometric aspects of contact and to know how to apportion the forces and deformation between contacting bodies of comparable strength.

7.4.6 Finite Element Models

The development of a finite element model for solid mechanics is normally accomplished through the principles of virtual work or the principle of minimum total potential energy [8,13]. These are equivalent to the weak forms or variational problems, and they form the basis of most finite element models in solid mechanics. The weak form formulations are also equivalent to these two principles. In this section the principle of virtual displacements is used to derive a finite element model.

The total virtual work done by actual forces in moving through virtual displacements, δW , for a solid body consists of the virtual internal (strain) energy δW_I and the virtual work done δW_e by the volume and surface forces. The virtual internal energy is taken to be only the virtual strain energy for the solid region and

it is expressed in terms of the Piola–Kirchhoff stress components $\tilde{\sigma}_{ij}$ and the virtual Green strain components δE_{ij} as

$$\delta W_I = \int_{\Omega_0} \tilde{\sigma}_{ij} \delta E_{ij} d\mathbf{X} \quad (7.4.29)$$

where Ω_0 is the volume occupied by the undeformed or reference configuration \mathcal{C}^0 . Using the constitutive relation (7.4.23) and the decomposition of the strain into linear and quadratic terms, the strain energy can be written as [8,13]

$$\begin{aligned} \delta W_I &= \int_{\Omega_0} C_{ijkl}(e_{kl}^1 + e_{kl}^2)(\delta e_{ij}^1 + \delta e_{ij}^2) d\mathbf{X} \\ &= \int_{\Omega_0} [C_{ijkl} e_{kl}^1 \delta e_{ij}^1 + C_{ijkl} (e_{kl}^1 \delta e_{ij}^2 + e_{ij}^2 \delta e_{kl}^1) + C_{ijkl} e_{kl}^2 \delta e_{ij}^2] d\mathbf{X} \end{aligned} \quad (7.4.30)$$

The strain components appearing in Eq. (7.4.30) can be expressed in terms of the displacement derivatives using the definitions in (7.4.15b). The virtual work done by the applied forces is given by

$$\delta W_E = - \left[\int_{\Omega_0} b_i \delta u_i d\mathbf{X} + \int_{\Gamma_0} f_i^\tau \delta u_i dS \right] \quad (7.4.31)$$

where Γ_0 is the boundary of the region Ω_0 and u_i are the displacement components. Note that the total virtual work done $\delta W = \delta W_I + \delta W_E$ is only a function of the displacements. The principle of virtual displacements, $\delta W = 0$, yields the equilibrium equations and force boundary conditions of the problem.

A finite element approximation is defined for the nodal values of the displacement components u_i as

$$u_i(\mathbf{x}, t) = \Phi^T \mathbf{u}_i \quad (7.4.32)$$

where Φ are the shape functions. The strain-displacement relations can then be expressed in matrix form as

$$\mathbf{e}^1 = \mathbf{D}^1 \mathbf{u} \quad , \quad \mathbf{e}^2 = \mathbf{u}^T \mathbf{D}^2 \mathbf{u} \quad (7.4.33)$$

where \mathbf{D}^i are matrix differential operators as implied by Eq. (7.4.15b). Substituting (7.4.32) and (7.4.33) into the expression for $\delta W = \delta W_I + \delta W_E$, we obtain the displacement finite element model of the problem in the form

$$[\mathbf{K}^0(\mathbf{u}) + \mathbf{K}^1(\mathbf{u}) + \mathbf{K}^2(\mathbf{u})] \mathbf{u} = \mathbf{F} \quad (7.4.34)$$

where each of the stiffness terms corresponds to the terms in (7.4.30), and the possible dependency of the elasticity tensor on the strain has been indicated. The linear, small strain and small deformation part is embodied in coefficient matrix \mathbf{K}^0 ; material nonlinearity may also be present in this term. The second matrix \mathbf{K}^1 is usually identified with geometric stiffness since it contains the quadratic strain terms. The matrix \mathbf{K}^2 is a nonlinear term that contains the quadratic strains and possible material nonlinearities.

7.4.7 Solution Methods

The finite element equations developed in the previous section for the quasi-static response of a solid generally represent a system of highly nonlinear algebraic equations. Two basic methods of solution may be considered. In the first case, the full matrix system in (7.4.34) is constructed and solved using any of the fixed point iterative methods, such as Newton's method. There are generally numerous difficulties with this approach, though when the problem is suitable the method may be very effective. The biggest drawbacks to direct iteration are nonconvergence of the iterative process and construction of the Jacobian for Newton's method. With any nonlinear problem, the starting point for the iterative procedure is crucial in achieving convergence (see Appendix C). The starting guess for most problems involving geometric and/or material nonlinearities will be far enough from the final, equilibrium solution that convergence will not be possible. The construction of the Jacobian is also difficult for many material models that are not analytic; the geometric nonlinearities can be incorporated analytically into the Jacobian through a tangent stiffness matrix.

The second approach to solving (7.4.34) involves an incremental or imbedding process, in which the load is applied in a series of increments and a sequence of equilibrium problems is solved. This method has the distinct advantage of keeping the next solution in the sequence "close" to the previously converged solution. Iterative methods, like Newton's method, are more robust in this incremental algorithm and convergence is more rapid. Though not formulated here, path- or history-dependent material models must be approached with an incremental strategy. The variety of incremental procedures is quite large and outside the scope of this discussion. Additional details on solution algorithms for solid mechanics problems may be found in [3,9-12].

7.5 Electromagnetics

7.5.1 Introduction

Problems involving the coupling of electromagnetic (EM) fields with fluid and thermal transport have a broad spectrum of applications ranging from astrophysics to manufacturing and to electromechanical devices and sensors. Here we will limit the discussion to the interaction of electromagnetic fields with solid bodies or incompressible fluids that are good electrical conductors. This eliminates many of the interesting problems involving plasmas, and concentrates the applications in the area of metals and liquid metal flows. In many of these problems an applied or induced magnetic field provides an additional body force to the fluid, which results in a convective motion. For high current applications, resistive heating may also be important as a volumetric energy source.

In the following section an outline of the field equations for electromagnetics is given along with their coupling to the nonisothermal, viscous flow problem and the solid body heat conduction problem. Subsequent sections show how the EM problem is redefined in terms of potential functions that are more suitable for finite element model development, and then describe some of the numerical issues associated with EM field simulation. A good introduction to coupled fluid-EM problems is available in [14]; general EM field theory is available in texts such as the one by Jackson

[15]. Finite element models for EM applications are well covered in the texts by Jin [16], Sadiku [17], Binns, et al. [18] and Silvester [19]. The coverage of the electromagnetics problem is substantially more detailed than the solid mechanics problem mainly because it is less familiar to practitioners of thermal sciences.

7.5.2 Maxwell's Equations

The appropriate mathematical description of electromagnetic phenomena in a conducting material region, Ω_C , is given by Maxwell's equations. In rational MKSA notation these equations are expressed as (see [9,10])

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \quad (7.5.1)$$

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \quad (7.5.2)$$

$$\nabla \cdot \mathbf{B} = 0 \quad (7.5.3)$$

$$\nabla \cdot \mathbf{D} = \rho \quad (7.5.4)$$

where the field variables are the electric field intensity, \mathbf{E} , the magnetic field intensity, \mathbf{H} , the magnetic flux density, \mathbf{B} , the electric flux (displacement) density, \mathbf{D} , the conduction current density, \mathbf{J} , and the source charge density, ρ . Typically, Eq. (7.5.1) is referred to as Faraday's law, Eq. (7.5.2) as Ampere's law (as modified by Maxwell), and Eq. (7.5.4) as Gauss's law. A continuity condition on the current density is also defined by

$$\nabla \cdot \mathbf{J} = \frac{\partial \rho}{\partial t} \quad (7.5.5)$$

Note that only three of the above five equations are independent; the combinations (7.5.1), (7.5.2) and (7.5.4) or (7.5.1), (7.5.2) and (7.5.5) form valid descriptions of the fields.

7.5.2.1 Constitutive relations

To complete the formulation, the constitutive relations for the material are required. The fluxes are functionally related to the field variables by

$$\mathbf{D} = f_D(\mathbf{E}, \mathbf{B}) \quad (7.5.6)$$

$$\mathbf{H} = f_H(\mathbf{E}, \mathbf{B}) \quad (7.5.7)$$

$$\mathbf{J} = f_J(\mathbf{E}, \mathbf{B}) \quad (7.5.8)$$

where the functions (f_D, f_H, f_J) may also depend on external variables such as temperature or mechanical stress. The form of the material response to applied \mathbf{E} or \mathbf{B} fields can vary strongly depending on the state of the material, its microstructure and the strength, and time-dependent behavior of the applied field.

Conductive and dielectric materials

For conducting materials, the standard f_J relation is Ohm's law which relates the current density \mathbf{J} to the electric field intensity \mathbf{E}

$$\mathbf{J} = \sigma \cdot \mathbf{E} \quad (7.5.9)$$

where σ is the conductivity tensor. For isotropic materials σ is a scalar. In general, the conductivity may be a function of \mathbf{E} or an external variable such as temperature. This form of Ohm's law applies to stationary conductors. If the conductive material is moving in a magnetic field, then Eq. (7.5.9) is modified to read

$$\mathbf{J} = \sigma \cdot \mathbf{E} + \sigma \cdot (\mathbf{v} \times \mathbf{B}) \quad (7.5.10)$$

where \mathbf{v} is the velocity vector describing the motion of the conductor and \mathbf{B} is the magnetic flux vector.

For dielectric materials, the standard f_D function relates the electric flux density \mathbf{D} to the electric field \mathbf{E} and polarization vector \mathbf{P}

$$\mathbf{D} = \epsilon_0 \mathbf{E} + \mathbf{P} \quad (7.5.11)$$

where ϵ_0 is the permittivity of free space. The polarization is generally related to the electric field through

$$\mathbf{P} = \epsilon_0 \chi_e \mathbf{E} + \mathbf{P}_0 \quad (7.5.12)$$

where χ_e is the electric susceptibility tensor that accounts for the different types of polarization, and \mathbf{P}_0 is the remnant polarization that may be present in some materials. The electric susceptibility is always positive and may be a simple scalar (linearly polarizable material) or a field-dependent scalar or tensor. In some situations the polarization exhibits a hysteretic behavior with respect to the electric field (ferroelectric material) and the susceptibility is defined as the local slope of the \mathbf{P} versus \mathbf{E} curve. Combining Eqs. (7.5.11) and (7.5.12), we obtain

$$\mathbf{D} = (\epsilon_0 \mathbf{I} + \epsilon_0 \chi_e) \cdot \mathbf{E} + \mathbf{P}_0 = \epsilon_0 (\mathbf{I} + \chi_e) \cdot \mathbf{E} + \mathbf{P}_0 = \epsilon \cdot \mathbf{E} + \mathbf{P}_0 \quad (7.5.13)$$

where \mathbf{I} is the unit tensor and $\epsilon = \epsilon_0 (\mathbf{I} + \chi_e)$. Note that if \mathbf{P}_0 exists, it only needs to be explicitly defined for linear materials (i.e., when ϵ is independent of \mathbf{E}); for nonlinear constitutive behavior, \mathbf{P}_0 is usually absorbed in the functional form for ϵ .

Magnetic materials

For magnetic materials, the standard f_H function relates the magnetic field intensity \mathbf{H} to the magnetic flux \mathbf{B}

$$\mathbf{H} = \frac{1}{\mu_0} \mathbf{B} - \mathbf{M} \quad (7.5.14)$$

where μ_0 is the permeability of free space and \mathbf{M} is the magnetization vector. The magnetization can be related to either the magnetic flux or magnetic intensity

$$\mathbf{M} = \frac{\chi_m}{(\mathbf{I} + \chi_m)} \frac{1}{\mu_0} \cdot \mathbf{B} + \mathbf{M}_0 \quad (7.5.15a)$$

or

$$\mathbf{M} = \chi_m \cdot \mathbf{H} + (\mathbf{I} + \chi_m) \cdot \mathbf{M}_0 \quad (7.5.15b)$$

where χ_m is the magnetic susceptibility for the material and \mathbf{M}_0 is the remnant magnetization. If the susceptibility is negative, the material is diamagnetic; while a positive susceptibility defines a paramagnetic material. Generally, these

susceptibilities are quite small and are often neglected. Ferromagnetic materials have large positive susceptibilities and produce a nonlinear (hysteretic) relationship between \mathbf{B} and \mathbf{H} . These materials may also exhibit spontaneous and remnant magnetization. Combining Eq. (7.5.14) with either Eq. (7.5.15a) or Eq. (7.5.15b) leads to

$$\mathbf{H} = \frac{1}{\mu_0(\mathbf{I} + \chi_m)} \cdot \mathbf{B} - \mathbf{M}_0 = \frac{1}{\mu} \cdot \mathbf{B} - \mathbf{M}_0 = \nu \cdot \mathbf{B} - \mathbf{M}_0 \quad (7.5.16)$$

Here $\mu = \mu_0(\mathbf{I} + \chi_m)$ is the permeability tensor for the material. The relative permeability is $\mu_r = \mu/\mu_0 = (\mathbf{I} + \chi_m)$ and the reluctivity ν , is defined as the inverse of the permeability. The remnant magnetization, \mathbf{M}_0 , need only be explicitly defined for linear magnetic materials (e.g., permanent magnets); in the nonlinear case \mathbf{M}_0 is usually absorbed into the $\mu(\mathbf{B})$ function.

7.5.2.2 Electromagnetic forces and volume heating

The coupling of electromagnetic fields with a fluid or thermal problem occurs through the dependence of material properties on EM field quantities and the production of EM-induced body forces and volumetric energy production. The Lorentz body force in a conductor due to the presence of electric currents and magnetic fields is given by

$$\mathbf{F}_B = \rho \mathbf{E} + \mathbf{J} \times \mathbf{B} \quad (7.5.17)$$

where, in the general case, the current is defined by Eq. (7.5.10). The first term on the right-hand side of Eq. (7.5.17) is the electric field contribution to the Lorentz force; the magnetic term $\mathbf{J} \times \mathbf{B}$ is usually of more interest in applied mechanics problems. The energy generation or Joule heating in a conductor is described by

$$Q_J = \mathbf{J} \cdot \mathbf{E} \quad (7.5.18)$$

which takes on a more familiar form if the simplified ($\mathbf{u} = \mathbf{0}$) form of Eq. (7.5.10) is used to produce

$$Q_J = \sigma^{-1}(\mathbf{J} \cdot \mathbf{J}) \quad (7.5.19)$$

The above forces and heat source occur in the fluid momentum and energy equations, respectively.

7.5.2.3 Quasi-static approximation

For good conductors, the conduction current \mathbf{J} is large compared to the displacement current \mathbf{D} for most frequencies of interest. Neglecting the displacement current allows Ampere's law (7.5.2) to be simplified and Coulomb's law (7.5.4) to be omitted. Also, the continuity relation is simplified since the divergence of $\nabla \times \mathbf{H}$ is zero by a vector identity. The omission of the displacement current is termed a quasi-static approximation (pre-Maxwell equations) since the propagation of electromagnetic waves is precluded. Under this assumption Maxwell's equations for a conducting region become

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \quad (7.5.20)$$

$$\nabla \times \mathbf{H} = \mathbf{J} \quad (7.5.21)$$

$$\nabla \cdot \mathbf{B} = 0 \quad (7.5.22)$$

The continuity condition is

$$\nabla \cdot \mathbf{J} = 0 \quad (7.5.23)$$

The simplified forms of the required constitutive relations are

$$\mathbf{B} = \mu \cdot \mathbf{H} \text{ or } \mathbf{H} = \nu \cdot \mathbf{B} \quad (7.5.24)$$

$$\mathbf{J} = \sigma \cdot \mathbf{E} + \sigma \cdot (\mathbf{v} \times \mathbf{B}) \text{ or } \mathbf{J} = \sigma \cdot \mathbf{E} \quad (7.5.25)$$

7.5.3 Electromagnetic Potentials

For many static and quasi-static applications it is usual to introduce a set of potential functions to represent the electric and magnetic field variables and reduce the number of partial differential equations requiring solution. Two basic systems of potentials may be considered: (a) the electric scalar potential V and a magnetic vector potential \mathbf{A} , and (b) the electric vector potential \mathbf{T} and the scalar magnetic potential ψ . The \mathbf{T} - ψ formulation is of limited value for a general analysis since there are significant difficulties in representing multiply-connected domains. Though the \mathbf{A} – V formulation generally leads to a larger number of differential equations, it is preferred for numerical computation due to its complete generality.

From Gauss' relation $\nabla \cdot \mathbf{B} = 0$, it follows that \mathbf{B} is derivable from a vector potential. By definition, then

$$\mathbf{B} = \nabla \times \mathbf{A} \quad (7.5.26)$$

where \mathbf{A} is the *magnetic vector potential*. In addition, from Faraday's law (7.5.20), one has

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} = -\frac{\partial(\nabla \times \mathbf{A})}{\partial t} \quad (7.5.27a)$$

Rearranging the above equation, we obtain

$$\nabla \times \left(\mathbf{E} + \frac{\partial \mathbf{A}}{\partial t} \right) = \mathbf{0} \quad (7.5.27b)$$

For a scalar V , the vector identity $\nabla \times \nabla V = \mathbf{0}$ holds and allows the definition

$$-\nabla V \equiv \mathbf{E} + \frac{\partial \mathbf{A}}{\partial t}$$

or

$$\mathbf{E} = -\nabla V - \frac{\partial \mathbf{A}}{\partial t} \quad (7.5.28)$$

The scalar function V is known as the *electric potential*. The definitions in Eqs. (7.5.26) and (7.5.28) may be used in the appropriate forms of Ampere's law and the current continuity equation to produce the needed field equations for \mathbf{A} and V in conductors and in free-space regions. Using Eqs. (7.5.24)–(7.5.26) and (7.5.28) in Eq. (7.5.21), one obtains

$$\nabla \times [\nu \cdot (\nabla \times \mathbf{A})] = -\sigma \cdot \frac{\partial \mathbf{A}}{\partial t} - \sigma \cdot \nabla V \quad (7.5.29)$$

Also, in the conduction regions the electric field is described by the continuity equation (7.5.23), which is rewritten in terms of \mathbf{A} and V by the use of Eqs. (7.5.25b) and (7.5.28)

$$\nabla \cdot \left(-\sigma \cdot \nabla V - \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} \right) = 0 \quad (7.5.30)$$

Equations (7.5.29)–(7.5.30) describe the general, quasi-static electromagnetic field problem for free space and conductors in terms of the magnetic vector potential \mathbf{A} and electric scalar potential V . The above derivation has been executed for stationary conductors with simple material behavior; the formulation is altered slightly if conductors are moving and/or constitutive behavior is more complex. These equations are appropriate for solution by the finite element method.

The general problem outlined above may now be specialized to particular material regions and types of current (\mathbf{J}) specifications. For conduction regions without source currents, both Eqs. (7.5.29) and (7.5.30) are generally required to describe both the electric and magnetic fields. Conduction regions that have specified currents may require some alteration of Eqs. (7.5.29)–(7.5.30), depending on the form of the current source (7.5.8). Electric currents described by distributions of the electric potential require no alteration to the equation set since this specification would appear as a boundary condition on the variable V . However, when current densities are assumed known and are specified directly, the total current density in Ohm's law must be rewritten as the combination of two parts. Define the total current density as

$$\mathbf{J} = \mathbf{J}_s + \mathbf{J}_e \quad (7.5.31)$$

where the known source current \mathbf{J}_s is associated with the electric potential term $-\sigma \cdot \nabla V$, and the induced or self-induced current is defined by the time-dependent magnetic potential term $-\sigma \cdot (\partial \mathbf{A} / \partial t)$. Therefore, in a conductor with a known current density \mathbf{J}_s , Eq. (7.5.29) is rewritten as

$$\nabla \times (\nu \cdot \nabla \times \mathbf{A}) + \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} = \mathbf{J}_s \quad (7.5.32)$$

Also, Eq. (7.5.30) is no longer required in the source region since the imposed current density is assumed to be divergence free; the induced currents in the source conductor are also divergence free due to the gauge condition that will be discussed in a subsequent section. Note that the total current through the source conductor is given by

$$I = \int_{\Gamma} \mathbf{J} \cdot \hat{\mathbf{n}} d\Gamma = \int_{\Gamma} \mathbf{J}_s \cdot \hat{\mathbf{n}} d\Gamma - \int_{\Gamma} \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} \cdot \hat{\mathbf{n}} d\Gamma \quad (7.5.33)$$

where $\hat{\mathbf{n}}$ is the vector normal to the cross-sectional area Γ of the conductor. For the general time-dependent problem, the total current, \mathbf{J} , cannot be specified a priori, since the self-induced portion of the current is obtained as a part of the solution. This implies that an iterative procedure is needed if specified current problems are defined.

The equations needed for free space or dielectric regions are simply those of (7.5.29) with σ set to zero; Eq. (7.5.30) is not required as no electric fields are

considered. Finally, note that simplification of the equations for all regions is possible for problems of reduced dimensionality. If the geometry is two-dimensional (planar or axisymmetric) and the currents and potential gradients are oriented orthogonal to the plane, then Eq. (7.5.30) is not required and Eq. (7.5.29) reduces to a single equation for the remaining (axial or circumferential) component of the magnetic potential.

The general time-dependent equations given in (7.5.29)–(7.5.30) are applicable to any type of time varying field problem. However, in the often encountered special case of a single frequency, time-harmonic excitation (e.g., alternating current), the equations may be simplified through use of a phasor representation. Let any specified current densities be represented as a time harmonic excitation and assume that the electric scalar and magnetic vector potentials have a time-harmonic form that can be represented as

$$\begin{aligned}\mathbf{J}_s &= \mathbf{J}_{s_0} e^{i\omega t} = (\mathbf{J}_s^R + i\mathbf{J}_s^I) e^{i\omega t} \\ V &= V_0 e^{i\omega t} = (V^R + iV^I) e^{i\omega t} \\ \mathbf{A} &= \mathbf{A}_0 e^{i\omega t} = (\mathbf{A}^R + i\mathbf{A}^I) e^{i\omega t}\end{aligned}\quad (7.5.34)$$

where ω is the circular frequency ($= 2\pi f$, f is the imposed AC frequency in Hz), $i = \sqrt{-1}$ and t is the time. The superscripts R and I denote the real and imaginary components of a variable. Then substituting (7.5.34) into (7.5.29) and (7.5.30) and eliminating the common exponential factor produces

$$\nabla \times (\nu \cdot \nabla \times \mathbf{A}_0) + i\omega\sigma\mathbf{A}_0 + \sigma \cdot \nabla V_0 = 0 \quad (7.5.35)$$

$$\nabla \cdot (\sigma \cdot \nabla V_0 + i\omega\sigma \cdot \mathbf{A}_0) = 0 \quad (7.5.36)$$

and for conduction regions with source currents (7.5.32) produces

$$\nabla \times (\nu \cdot \nabla \times \mathbf{A}_0) + i\omega\sigma\mathbf{A}_0 = \mathbf{J}_{s_0} \quad (7.5.37)$$

These complex equations describe the amplitudes for the potentials. Note that the boundary conditions for V and \mathbf{A} must also be expressed in terms of the harmonic approximation given in (7.5.34). Implicit in the use of the phasor representation is the assumption that material properties are independent of the temporal behavior of the electromagnetic fields. This is a particularly stringent requirement that is violated when considering high field applications such as induction heating or most types of ferromagnetic materials.

7.5.4 Boundary and Interface Conditions

Boundary and interface conditions for the quasi-static, electromagnetic field problem are most easily described by reference to the generic domain shown in Figure 7.5.1. The region Ω is composed of a number of different materials ($\Omega = \Omega_C \cup \Omega_J \cup \Omega_D$), several of which are illustrated in the figure. The boundary or interface between two conductors is denoted by Γ_{CC} while the boundary between a dielectric or free space region and a conductor is labelled Γ_{DC} . Note that since the equations for Ω_J and Ω_D are the same except for a source function, no specific designation for an interface between these regions is required. The external boundary of the entire domain Ω

is defined by Γ which may be composed of one or more well-defined segments. A two-dimensional representation of the region is used for simplicity.

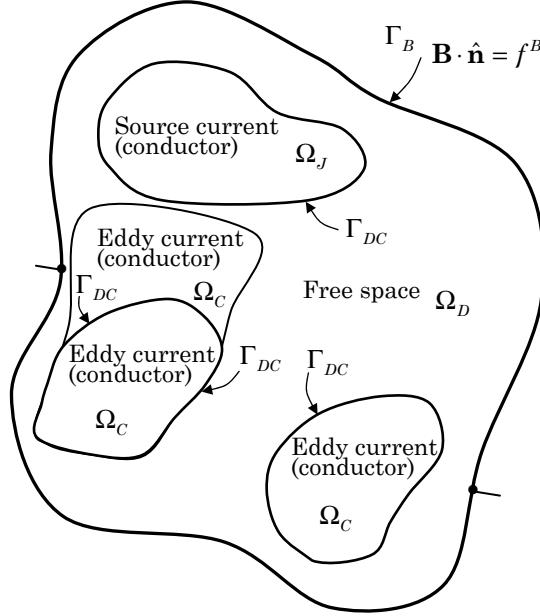


Figure 7.5.1: Schematic of regions for general electromagnetic field problem.

The electromagnetic problem requires that on the exterior free space boundary either the magnetic flux or the magnetic field be specified at all points of the boundary, Γ . In equation form these conditions are given by

$$\mathbf{B} \cdot \hat{\mathbf{n}} = (\nabla \times \mathbf{A}) \cdot \hat{\mathbf{n}} = f^B(s_k, t) \text{ on } \Gamma_B \quad (7.5.38)$$

$$\mathbf{H} \times \hat{\mathbf{n}} = (\nu \cdot \mathbf{B}) \times \hat{\mathbf{n}} = (\nu \cdot \nabla \times \mathbf{A}) \times \hat{\mathbf{n}} = \mathbf{f}^H(s_k, t) \text{ on } \Gamma_H \quad (7.5.39)$$

In Eqs. (7.5.38) and (7.5.39) the f^B and \mathbf{f}^H functions are specified values of the known boundary magnetic flux and magnetic field. Also, $\hat{\mathbf{n}}$ is the outward unit normal to the boundary Γ , s_k are coordinates defined on the boundary, t is the time, and $\Gamma = \Gamma_B \cup \Gamma_H$. The functions f^B and \mathbf{f}^H are generally simple expressions for most boundaries of practical interest. When a conductor forms part of the external boundary the above conditions are augmented with a condition on the current flux or the electric field. That is,

$$\mathbf{J} \cdot \hat{\mathbf{n}} = \sigma \cdot \mathbf{E} \cdot \hat{\mathbf{n}} = \left(-\sigma \cdot \frac{\partial \mathbf{A}}{\partial t} - \sigma \cdot \nabla V \right) \cdot \hat{\mathbf{n}} = f^J(s_k, t) \text{ on } \Gamma_J \quad (7.5.40)$$

or

$$\mathbf{E} \times \hat{\mathbf{n}} = -\nabla V - \frac{\partial \mathbf{A}}{\partial t} \times \hat{\mathbf{n}} = \mathbf{f}^E(s_k, t) \text{ on } \Gamma_E \quad (7.5.41)$$

Along a material interface, such as Γ_{CC} or Γ_{DC} , the usual assumption is that the normal component of the magnetic flux is continuous and the tangential component of the magnetic field is discontinuous by an amount equal to the surface current, \mathbf{J}_{surf} . These conditions are specified by

$$(\mathbf{B}_2 - \mathbf{B}_1) \cdot \hat{\mathbf{n}} = (\nabla \times \mathbf{A}_1 - \nabla \times \mathbf{A}_2) \cdot \hat{\mathbf{n}} = 0 \text{ on } \Gamma_{CC}, \Gamma_{DC} \quad (7.5.42)$$

$$(\mathbf{H}_2 - \mathbf{H}_1) \times \hat{\mathbf{n}} = (\nu_1 \cdot \nabla \times \mathbf{A}_1 - \nu_2 \cdot \nabla \times \mathbf{A}_2) \times \hat{\mathbf{n}} = \mathbf{J}_{surf} \text{ on } \Gamma_{CC}, \Gamma_{DC} \quad (7.5.43)$$

where the subscripts 1 and 2 indicate variables evaluated on either side of the interface. In many cases the surface current is not important and may be neglected.

The divergence and curl relations for the electric field also provide two conditions at a material interface. In this case the normal component of the current density is continuous and the tangential components of the electric field are continuous. That is,

$$(\mathbf{J}_2 - \mathbf{J}_1) \cdot \hat{\mathbf{n}} = (\sigma_2 \cdot \mathbf{E}_2 - \sigma_1 \cdot \mathbf{E}_1) \cdot \hat{\mathbf{n}} = 0 \text{ on } \Gamma_{CC} \quad (7.5.44)$$

$$(\mathbf{E}_2 - \mathbf{E}_1) \times \hat{\mathbf{n}} = \mathbf{0} \text{ on } \Gamma_{CC} \quad (7.5.45)$$

for the boundary between two conductors and

$$\mathbf{J}_2 \cdot \hat{\mathbf{n}} = \sigma_2 \cdot \mathbf{E}_2 \cdot \hat{\mathbf{n}} = 0 \text{ on } \Gamma_{DC} \quad (7.5.46)$$

$$\mathbf{E}_2 \times \hat{\mathbf{n}} = \mathbf{0} \text{ on } \Gamma_{DC} \quad (7.5.47)$$

for the boundary between a very good conductor and a dielectric where the subscript 2 refers to the conducting region.

7.5.5 Gauge Conditions

The quasi-static form of Maxwell's equations is given by Eqs. (7.5.29)–(7.5.30) in terms of the magnetic and electric potentials. The original or primitive variable form of Maxwell's Eqs. (7.5.20)–(7.5.22) can be shown to provide unique solutions for the \mathbf{B} and \mathbf{E} fields when appropriate boundary conditions are specified. However, with the introduction of the potential variables, uniqueness of the solution is not retained, i.e., Eqs. (7.5.29) and (7.5.30) define the curl of \mathbf{A} , but \mathbf{A} itself is only defined up to the gradient of an arbitrary scalar function. Typically this arbitrariness in \mathbf{A} is resolved by defining the divergence of \mathbf{A} and supplying appropriate boundary conditions for \mathbf{A} (rather than boundary conditions for the curl of \mathbf{A}). The incorporation of a $\nabla \cdot \mathbf{A}$ constraint, termed gauging, may be accomplished in any of several ways. The Coulomb gauge is one particular choice that has found extensive use in numerical simulation methods. In this case the magnetic vector potential is made unique by the constraint

$$\nabla \cdot \mathbf{A} = 0 \quad (7.5.48)$$

Other choices for the gauge condition, such as the Lorentz gauge, select a nonhomogeneous form for Eq. (7.5.48). Note that in some cases of reduced dimensionality, the explicit use of Eq. (7.5.48) is not required since vector \mathbf{A} is automatically divergence free.

The implementation of Eq. (7.5.48) in a numerical method may take any of several forms, including modification of the field equations, penalty methods, projection methods, and the construction of divergence free basis functions. All of these techniques incur a computational penalty in terms of either additional work or the modification of the equation system to a less desirable form.

For some applications a unique value of the magnetic potential is not required and the above gauge condition may be neglected. In particular, if the magnetic field is time independent, then all field quantities are related to the curl of \mathbf{A} and a unique value of \mathbf{A} is not required. However, for the time-dependent case, the electric field \mathbf{E} is related to the time derivative of \mathbf{A} in Eq. (7.5.28) and the \mathbf{A} field must be unique if \mathbf{E} is required.

7.5.6 Static Field Problems

Within the general framework established above, a number of simpler static problems may also be defined. Each of these problem classes may have importance in the context of coupling with other mechanics problems or as a stand alone analysis in electromagnetics. As subclasses of the general formulation, they may be solved with many of the same numerical techniques as the general quasi-static problem.

7.5.6.1 Electrostatics

The electrostatic problem is described by Gauss's law (7.5.4) and the definition of the electric flux (displacement current). Combining Eq. (7.5.4) with the simple conductive form of the constitutive relation in (7.5.11) leads to

$$\nabla \cdot \mathbf{D} = \nabla \cdot (\epsilon \mathbf{E}) = \rho \quad (7.5.49)$$

where ρ is the spatial distribution of (free) electric charge. Faraday's law (7.5.1) under steady conditions ($\nabla \times \mathbf{E} = 0$) implies that \mathbf{E} is derivable from a scalar potential, $\mathbf{E} = -\nabla V$, and thus (7.5.49) becomes

$$\nabla \cdot (\epsilon \cdot \nabla V) = -\rho \quad (7.5.50)$$

which is valid for electrically conductive materials. Note that constant potential regions (conductors without a specified charge distribution) may be removed from the analysis domain or approximated as a high permittivity material. For dielectrics, the free charge is zero and the more general form of the constitutive relation in (7.5.11) can be used to produce

$$\nabla \cdot (\epsilon \cdot \nabla V) = \nabla \cdot \mathbf{P}_0 \quad (7.5.51)$$

Nonlinear dielectrics would ignore the source term in (7.5.51) and use $\epsilon(\mathbf{E})$. Boundary conditions for electrostatics generally involve specification of the scalar potential, V , or the definition of the electric flux normal to the boundary (i.e., the normal derivative of the potential). When the spatial variation of the potential has been determined, the electric field \mathbf{E} and the electric flux may be found from the relevant definitions.

7.5.6.2 Steady current flow

For time-independent problems the system in (7.5.29) and (7.5.30) becomes decoupled and the current continuity condition (7.5.30) may be written as

$$\nabla \cdot (-\sigma \cdot \nabla V) = 0 \quad (7.5.52)$$

with

$$\mathbf{J} = -\sigma \cdot \nabla V \quad (7.5.53)$$

Equations (7.5.52) and (7.5.53) describe steady electric currents within a conductor. Boundary conditions on the system would normally include specification of the electric potential (voltage) over part of the boundary and/or the current density normal to the boundary, i.e., $\frac{\partial V}{\partial n}$. Once the electric potential and current distributions have been found then the Joule heating could be recovered from the definition in Eq. (7.5.19).

7.5.6.3 Magnetostatics

Ampere's law (7.5.30) or (7.5.32) for the time-independent case becomes

$$\nabla \times (\nu \cdot \nabla \times \mathbf{A}) = -\sigma \cdot \nabla V = \mathbf{J}_s \quad (7.5.54)$$

This describes the magnetic field due to specified current distributions \mathbf{J}_s . Note that the conduction currents could be specified directly or computed from the steady current flow Eq. (7.5.52) for the electric potential V . When the magnetic potential is known, then the magnetic field \mathbf{B} may be computed from its definition in (7.5.26). In addition, the Lorentz forces can be found from Eq. (7.5.17).

7.5.7 Finite Element Models for EM Fields

The potential equations in (7.5.29) and (7.5.30) represent the three components of the vector magnetic potential and the scalar electric potential for the general quasi-static field problem. Using the standard method of weighted residual techniques these equations can be converted to appropriate weak forms for subsequent use with a finite element approximation. This process will be outlined here for the case of a conducting material; simplifications for free space or source current regions are obvious.

7.5.7.1 Quasi-static potential equations

The weak or weighted integral forms corresponding to (7.5.29) and (7.5.30) are obtained by defining a vector weighting function \mathbf{W} and a scalar weighting function W , multiplying (7.5.29) and (7.5.30) by the appropriate function and integrating each equation over the conducting region. That is,

$$\int_{\Omega_C} \mathbf{W} \cdot \nabla \times (\nu \cdot \nabla \times \mathbf{A}) d\Omega + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} d\Omega + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \nabla V d\Omega = 0 \quad (7.5.55)$$

and

$$\int_{\Omega_C} W \nabla \cdot (-\sigma \cdot \nabla V) d\Omega + \int_{\Omega_C} W \nabla \cdot (-\sigma \cdot \frac{\partial \mathbf{A}}{\partial t}) d\Omega = 0 \quad (7.5.56)$$

The weak forms in (7.5.55) and (7.5.56) may be further transformed by utilizing Gauss's theorem to reduce the highest order derivative terms. Proceeding first with the magnetic potential equation, use the definition $\mathbf{H} = \nu \cdot \nabla \times \mathbf{A}$ and rewrite (7.5.55) as

$$\int_{\Omega_C} \mathbf{W} \cdot \nabla \times \mathbf{H} d\Omega + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} d\Omega + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \nabla V d\Omega = 0 \quad (7.5.57)$$

Using the vector identity

$$\nabla \cdot (\mathbf{W} \times \mathbf{H}) = \mathbf{H} \cdot \nabla \times \mathbf{W} - \mathbf{W} \cdot \nabla \times \mathbf{H} \quad (7.5.58)$$

Eq. (7.5.57) is written as

$$\begin{aligned} & \int_{\Omega_C} \mathbf{H} \cdot \nabla \times \mathbf{W} d\Omega - \int_{\Omega_C} \nabla \cdot (\mathbf{W} \times \mathbf{H}) d\Omega + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} d\Omega \\ & + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \nabla V d\Omega = 0 \end{aligned} \quad (7.5.59)$$

Introducing the divergence theorem for the second integral and the vector identity $(\mathbf{W} \times \mathbf{H}) \cdot \mathbf{n} = \mathbf{W} \cdot (\mathbf{H} \times \mathbf{n})$ then (7.5.59) becomes

$$\begin{aligned} & \int_{\Omega_C} \mathbf{H} \cdot \nabla \times \mathbf{W} d\Omega - \int_{\Gamma_C} \mathbf{W} \cdot (\mathbf{H} \times \mathbf{n}) d\Gamma + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} d\Omega \\ & + \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \nabla V d\Omega = 0 \end{aligned} \quad (7.5.60)$$

where \mathbf{n} is the outward normal to the boundary Γ_C . Finally, reintroducing the definition of \mathbf{H} , rearranging and introducing the natural boundary condition for the magnetic potential, lead to the required form of the integral statement

$$\begin{aligned} & \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} d\Omega + \int_{\Omega_C} \nabla \times \mathbf{W} \cdot (\nu \cdot \nabla \times \mathbf{A}) d\Omega \\ & = - \int_{\Omega_C} \mathbf{W} \cdot \sigma \cdot \nabla V d\Omega + \int_{\Gamma_C} \mathbf{W} \cdot \mathbf{f}^H d\Gamma \end{aligned} \quad (7.5.61)$$

The electric potential equation (7.5.56) is transformed in a similar manner. Using the divergence theorem on the first two integrals in (7.5.56) leads to

$$\begin{aligned} & - \int_{\Omega_C} \nabla W \cdot (-\sigma \cdot \nabla V) d\Omega - \int_{\Omega_C} \nabla W \cdot \left(-\sigma \cdot \frac{\partial \mathbf{A}}{\partial t} \right) d\Omega \\ & = - \int_{\Gamma_C} W \left(-\sigma \cdot \frac{\partial \mathbf{A}}{\partial t} - \sigma \cdot \nabla V \right) \cdot \mathbf{n} d\Gamma \end{aligned} \quad (7.5.62)$$

This may be rearranged into a standard form, with the natural boundary conditions for the potential included to produce

$$\int_{\Omega_C} \nabla W \cdot \sigma \cdot \nabla V d\Omega + \int_{\Omega_C} \nabla W \cdot \sigma \cdot \frac{\partial \mathbf{A}}{\partial t} d\Omega = \int_{\Gamma_C} W f^J d\Gamma \quad (7.5.63)$$

Equations (7.5.61) and (7.5.63) are weak forms of the potential equations for electromagnetics and are suitable for use with a finite element approximation. Though not shown explicitly, the above formulation can be easily extended to include the case of motion of the conductor and/or more complex material behavior, e.g., remnant magnetization. To proceed with the finite element model, the region Ω_C is discretized into an assemblage of finite elements and the weighted integral statements are applied to each element. Within each element the vector magnetic potential and scalar electric potential are approximated by expansions of the form

$$\mathbf{A}(\mathbf{x}, t) = \Phi^T(\mathbf{x}) \mathbf{A}_x(t) \mathbf{e}_x + \Phi^T(\mathbf{x}) \mathbf{A}_y(t) \mathbf{e}_y + \Phi^T(\mathbf{x}) \mathbf{A}_z(t) \mathbf{e}_z \quad (7.5.64a)$$

$$V(\mathbf{x}, t) = \Psi^T(\mathbf{x}) \mathbf{V}(t) \quad (7.5.64b)$$

which are written here for the three-dimensional Cartesian case with the \mathbf{e}_i being unit vectors for the coordinate system; similar expressions can be constructed for the axisymmetric and two-dimensional cases. In Eqs. (7.5.64a,b) Φ and Ψ represent vectors of interpolation functions, \mathbf{A}_i and \mathbf{V} are vectors of nodal point unknowns and superscript T indicates a vector transpose. Note that the assumed approximations for the dependent variables are, as usual, semi-discrete with the spatial dependence being discretized through interpolation and the temporal dependence remaining continuous. For the Galerkin method the weight functions \mathbf{W} and W are selected to be the same functions as used to represent the variables. That is,

$$\mathbf{W}(\mathbf{x}) = \Phi(\mathbf{x}) \mathbf{e}_x + \Phi(\mathbf{x}) \mathbf{e}_y + \Phi(\mathbf{x}) \mathbf{e}_z \quad (7.5.65a)$$

$$W(\mathbf{x}) = \Psi(\mathbf{x}) \quad (7.5.65b)$$

Substituting the definitions in (7.5.64) and (7.5.65) into the weighted residual equations in (7.5.61) and (7.5.63) produces the following set of discrete equations for each element

$$\mathbf{M} \dot{\mathbf{A}} + \mathbf{K} \mathbf{A} + \mathbf{N} \mathbf{V} = \mathbf{F}_A \quad (7.5.66)$$

$$\mathbf{N}^T \dot{\mathbf{A}} + \mathbf{L} \mathbf{V} = \mathbf{F}_V \quad (7.5.67)$$

where the superposed dot indicates a time derivative. Equation (7.5.66) represents the three component equations for the magnetic potential.

The matrix system shown above is unsymmetric and is of an undesirable form from the standpoint of time integration. To restore symmetry, the following definition proposed by Chari, et al. [15] may be employed

$$\mathbf{V} \equiv \frac{\partial \mathbf{v}}{\partial t} = \dot{\mathbf{v}} \quad (7.5.68)$$

Using this definition Eqs. (7.5.66) and (7.5.67) can be rewritten as

$$\mathbf{M} \dot{\mathbf{A}} + \mathbf{K} \mathbf{A} + \mathbf{N} \dot{\mathbf{v}} = \mathbf{F}_A \quad (7.5.69)$$

$$\mathbf{N}^T \dot{\mathbf{A}} + \mathbf{L} \dot{\mathbf{v}} = \mathbf{F}_V \quad (7.5.70)$$

and in a completely assembled form

$$\begin{bmatrix} \mathbf{M} & \mathbf{N} \\ \mathbf{N}^T & \mathbf{L} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{A}} \\ \dot{\mathbf{v}} \end{Bmatrix} + \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{A} \\ \mathbf{v} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_A \\ \mathbf{F}_V \end{Bmatrix} \quad (7.5.71)$$

The component matrices defined in (7.5.71) are defined by the following integrals that arise from the weighted residual statements. The matrices are written here in terms of vector notation; the explicit forms involving derivatives on the element and the Jacobian for the transformation to the master element can be obtained by use of obvious substitutions from previous chapters

$$\begin{aligned}\mathbf{M} &= \int_{\Omega_e} \Phi \cdot \sigma \cdot \Phi^T d\Omega, & \mathbf{K} &= \int_{\Omega_e} \nabla \times \Phi \cdot \nu \cdot \nabla \times \Phi^T d\Omega \\ \mathbf{N} &= \int_{\Omega_e} \Phi \cdot \sigma \cdot \nabla \Psi^T d\Omega, & \mathbf{N}^T &= \int_{\Omega_e} \nabla \Psi \cdot \sigma \cdot \Phi^T d\Omega \\ \mathbf{L} &= \int_{\Omega_e} \nabla \Psi \cdot \sigma \cdot \nabla \Psi^T d\Omega, & \mathbf{F}_A &= \int_{\Gamma_e} \Phi \cdot \mathbf{f}^H d\Gamma \\ \mathbf{F}_M &= \int_{\Omega_e} \nabla \times \Phi \cdot \mathbf{M}_0 d\Omega, & \mathbf{F}_V &= \int_{\Gamma_e} \Psi f^J d\Gamma\end{aligned}\quad (7.5.72)$$

7.5.7.2 Gauge condition

When the magnetic potential must be made unique, the Coulomb gauge defined in (7.5.48) is added to the field equations as a constraint. From a computational standpoint there are several methods available for developing a discrete form of (7.5.48). One of the simplest implementations is the penalty method. The penalty method is most easily invoked by casting the original quasi-static field problem in terms of a functional to be minimized. To simplify the derivation the specified current form of the magnetostatics problem will be considered; the more general quasi-static or eddy current problem is developed in an analogous manner. Consider the static form of (7.5.29) or more precisely, Eq. (7.5.54)

$$\nabla \times (\nu \cdot \nabla \times \mathbf{A}) = \mathbf{J}_s \quad (7.5.73)$$

The functional associated with (7.5.73) is

$$I(\mathbf{A}) = \frac{1}{2} \int_{\Omega} \nabla \times \mathbf{A} \cdot \nu \cdot \nabla \times \mathbf{A} d\Omega - \int_{\Omega} \mathbf{J}_s \cdot \mathbf{A} d\Omega \quad (7.5.74)$$

A finite element approximation for \mathbf{A} may be used in (7.5.74), which when minimized with respect to the dependent variables produces a discrete system that is the same as the appropriate magnetostatics subset of the weak-form Galerkin equations in (7.5.71). To invoke the Coulomb gauge on this system, the functional in (7.5.74) is augmented with a least squares penalty term

$$I_C(\mathbf{A}) = I(\mathbf{A}) + \frac{\lambda}{2} \int_{\Omega} (\nabla \cdot \mathbf{A})^2 d\Omega \quad (7.5.75)$$

where λ is the penalty parameter. When a finite element approximation, such as (7.5.64) is used in the augmented functional in (7.5.75), and variations are taken with respect to the components of \mathbf{A} , the result is a discrete system of the following form

$$(\mathbf{K} + \lambda \mathbf{K}_{\text{div}}) \mathbf{A} = \mathbf{F}_A + \mathbf{F}_J \quad (7.5.75)$$

where most of the matrices and vectors are defined in (7.5.72) and the penalty term is defined by

$$\mathbf{K}_{\text{div}} = \int_{\Omega_e} \nabla \cdot \boldsymbol{\Phi} \nabla \cdot \boldsymbol{\Phi}^T d\Omega \quad (7.5.76)$$

This process is very similar to the penalty method used to enforce incompressibility in the viscous flow problem. The penalty enforcement of the Coulomb gauge requires the construction of an additional matrix that is added to the appropriate terms in the magnetic vector potential equation. The penalty parameter λ is set to a large number with the free space reluctivity being a good choice ($\lambda = 1/\mu_0 = \nu_0$). In order to avoid an over-constrained system, the penalty matrix must be singular. The standard method for achieving this situation is to reduce the order of the quadrature rule used to evaluate the integral in (7.5.76).

7.5.7.3 Static field equations

The finite element equations for the simplified static field problems are developed in the same manner as outlined in the previous sections. For some situations, the equations are merely subsets of the more complex quasi-static problem.

The basic equation for electrostatics is either (7.5.50) or (7.5.51) depending on the type of material. If combined, the relevant equation is

$$\nabla \cdot (\epsilon \cdot \nabla V) = -\rho \nabla \cdot \mathbf{P}_0 \quad (7.5.77)$$

The weak form of this equation is given by

$$\int_{\Omega} \nabla W \cdot \epsilon \cdot \nabla V d\Omega = - \int_{\Omega} W \rho d\Omega + \int_{\Omega} \nabla W \cdot \mathbf{P}_0 d\Omega + \int_{\Gamma} W f^D d\Gamma \quad (7.5.78)$$

where the natural boundary condition is a specification of the electric flux (gradient of the potential) normal to the boundary. Let the electric potential be approximated by the steady form of the finite element representation given in (7.5.64b)

$$V(\mathbf{x}) = \boldsymbol{\Psi}^T(\mathbf{x}) \mathbf{V} \quad (7.5.79)$$

Using the Galerkin definition and the scalar weight function in (7.5.65b), Eq. (7.5.78) leads to the matrix equation

$$\mathbf{L}_\epsilon \mathbf{V} = \mathbf{F}_\rho + \mathbf{F}_P + \mathbf{F}_D \quad (7.5.80)$$

where

$$\begin{aligned} \mathbf{L}_\epsilon &= \int_{\Omega_e} \nabla \boldsymbol{\Psi} \cdot \epsilon \cdot \nabla \boldsymbol{\Psi}^T d\Omega, & \mathbf{F}_\rho &= \int_{\Omega_e} \boldsymbol{\Psi} \rho d\Omega \\ \mathbf{F}_P &= \int_{\Omega_e} \nabla \boldsymbol{\Psi} \cdot \mathbf{P}_0 d\Omega, & \mathbf{F}_D &= \int_{\Gamma_e} \boldsymbol{\Psi} f^D d\Gamma \end{aligned} \quad (7.5.81)$$

The steady current flow problem is a subset of the general quasi-static formulation and can be defined immediately as

$$\mathbf{L}\mathbf{V} = \mathbf{F}_V \quad (7.5.82)$$

where the matrix and vector are defined in (7.5.72).

The magnetostatics problem is also a subset of the general quasi-static problem and corresponds to the equations for a free space region with specified currents. The relevant finite element equations are derived from (7.5.69)

$$\mathbf{KA} = \mathbf{F}_A + \mathbf{F}_J \quad (7.5.83)$$

where the vector \mathbf{F}_J is a weighted integral over the element volume of the specified current density.

7.5.8 Solution Methods – EM Fields

The finite element equations for the general quasi-static electromagnetics problem, as given in Eq. (7.5.71), represent a set of ordinary differential equations not unlike the equation sets that were encountered in the heat conduction and viscous flow formulations. Because of this similarity, virtually all of the solution methods used in the transport problems can be used in the electromagnetics problem without modification.

For static problems, the equations in (7.5.71) reduce to nonlinear algebraic equations that can be linearized and solved via iteration using the Picard or Newton methods. The general time-dependent case is well suited to implicit time integration methods such as backward Euler or the trapezoid rule. Both constant time step and adaptive time step, predictor/corrector methods, have been used successfully with these types of integrators.

The only problem type that differs significantly from the transport equations is the time harmonic field problem. If the nodal point variables are defined through a phasor representation with frequency ω [as was done for the continuum case in Eq. (7.5.34)], the equations in (7.5.69) and (7.5.70) become

$$i\omega\mathbf{MA}_0 + \mathbf{KA}_0 + i\omega\mathbf{Nv}_0 = \mathbf{F}_{A_0} \quad (7.5.84)$$

$$i\omega\mathbf{N}^T\mathbf{A}_0 + i\omega\mathbf{Lv}_0 = \mathbf{F}_{v_0} \quad (7.5.85)$$

or in the combined matrix form

$$\begin{bmatrix} i\omega\mathbf{M} & i\omega\mathbf{N} \\ i\omega\mathbf{N}^T & i\omega\mathbf{L} \end{bmatrix} \begin{Bmatrix} \mathbf{A}_0 \\ \mathbf{v}_0 \end{Bmatrix} + \begin{bmatrix} \mathbf{K} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{A}_0 \\ \mathbf{v}_0 \end{Bmatrix} = \begin{Bmatrix} \mathbf{F}_A \\ \mathbf{F}_V \end{Bmatrix} \quad (7.5.86)$$

where the unknowns are now complex and defined by $V_0 = (V^R + iV^I)$ and $\mathbf{A}_0 = (\mathbf{A}^R + i\mathbf{A}^I)$. Recall also that $V_0 = v_0 = i\omega v_0$.

Through the use of the phasor representation, the time harmonic problem has been reduced to a steady, linear matrix problem; the matrix coefficients and nodal point variables are complex.

The linear algebra problem associated with each of the steady, time-dependent or harmonic solution methods may be solved using either direct matrix solvers or iterative solvers of the conjugate gradient or Krylov type. Note that the matrix systems are usually unsymmetric. The gauge condition alters the matrix structure and may cause convergence difficulties for iterative methods. The complex coefficients in the time harmonic case can be used directly in solvers set up for complex arithmetic. An alternative is to separate the complex matrix into real and imaginary parts and solve twice the number of equations using only real coefficients.

7.6 Coupled Problems in Mechanics

7.6.1 Introduction

The equations from the three previous sections may be combined in a variety of ways to describe a large class of problems in mechanics. The fact that there is such a large spectrum of problems implies that meaningful discussions of coupling will be difficult to accomplish in a general way. Therefore, we will revert to a simple and expeditious method of description that considers the pairwise combinations of five mechanics areas: heat conduction (HC), nonisothermal viscous flow (VF), quasi-static solid mechanics (SM), electric fields (EF), and magnetic fields (MF). The symmetric coupling matrix shown in Figure 7.6.1 is used to organize and focus the discussion. Note that at the intersection of each pair of mechanics areas, one or more of the usual types of coupled problems are listed. No attempt is made to be thorough in this listing and only the most common interactions are indicated. In the matrix and in the following sections, some of the possible data dependencies for each interaction will be described, the typical strength of the interaction will be cited, and the types of algorithms that may be used will be suggested. Since this is a book on fluid mechanics and heat transfer, the focus will be on the first two rows of the matrix.

	1 Heat Conduction/ Radiation	2 Viscous/ Porous Flow	3 Quasi- Static Solids	4 Electric Fields	5 Magnetic Fields
1 Heat Conduction/ Radiation		1-2 Conjugate Heat Transfer B (S)	1-3 Thermal Stress V.P.G. (W-S)	1-4 Resistive Heating V.P. (W-S)	1-5 Inductive Heating V.P. (W-S)
2 Viscous/ Porous Flow	2-1		2-3 Fluid/Solid Interaction B.G. (S)	2-4 EHD Electro- Rheology V.P. (S-W)	2-5 MHD Inductive Stirring V.P. (W-S)
3 Quasi- Static Solids	3-1	3-2		3-4 Electro- Striction Piezoelectric B.P.G.(S-W)	3-5 Magnetic Stress V.G. (W-S)
4 Electric Fields	4-1	4-2	4-3		4-5 Eddy Currents V.P. (S)
5 Magnetic Fields	5-1	5-2	5-3	5-4	

Types of coupling:

- B – Boundary conditions/surface flux
- V – Volume terms; G – Geometry
- P – Property/constitutive dependence

Strength of coupling:

- S – Strong
- W – Weak

Figure 7.6.1: Matrix of possible mechanics interactions.

7.6.2 Heat Conduction – Viscous Fluid Interactions 1 & 2

For completeness of the exposition, some of the coupled problem situations from previous chapters are reviewed here in summary form. Consider the interaction of heat conduction and radiation, and conduction and chemical reaction from Chapter 3. These are strongly coupled problems with conduction/radiation being a boundary condition interaction and possibly a thermophysical property interaction if the emissivity is a function of temperature. The conduction/chemical reaction problem is primarily a volume source interaction on the conduction side of the interaction and a (kinetics) property interaction for the reaction equations. Because of the strong coupling, the preferred method of solution would be a simultaneous solution of each pair of equations.

The completely coupled solution process is often not practical in these two cases because of the characteristics of the radiation and chemical reaction equations. In the conduction/radiation case, the matrix structure of the radiation equation is full due to the view factor (surface-to-surface) coupling while the conduction matrix is sparse. The size of the radiation matrix and its precipitous growth with increase in mesh resolution often make the combined matrix problem not tractable. The combined conduction/reaction problem suffers mainly from a problem of disparate time scales and stiffness in the reaction equations. The excessively small time steps required in the chemistry solution are an unnecessary computational penalty for the thermal diffusion problem. Cyclic or operator splitting methods may not be optimal from a coupling perspective, but are practical from an overall algorithm point of view as detailed in Chapter 3.

The conjugate problem that couples solid body conduction with a nonisothermal flow is another strongly coupled problem with an interaction at the boundary. This class of problems is typically solved as a fully coupled matrix problem. The addition of conducting regions (scalar equation) to a primarily viscous flow problem (vector equation) presents no real increase in computational burden. Time constants for the two equation sets are usually of the same order of magnitude so that integration methods may follow the smaller time scale without too much of a penalty and coupling is straightforward. Note that it is possible to treat the conjugate problem in a decoupled or cyclic manner. In this case, the temperature and flux at the common boundary are used as the coupling variables. One variable is used as a boundary condition while the second variable is used to measure the convergence or agreement with the other equation set. Though feasible, this is not a recommended solution method for this type of problem.

7.6.3 Heat Conduction – Quasi-Static Solid Interactions 1 & 3

Thermal stress problems are one of the most heavily studied and commonly encountered types of coupled mechanics problems. In most cases the coupling is relatively weak, with the thermal field providing a thermal strain through the mechanical constitutive relation; in some cases a temperature dependence of the parameters in the constitutive equation adds to the interaction. When the interaction is limited to these effects, the problem can be easily solved via a one-way or decoupled procedure. With Eq. (7.3.7) representing the finite element heat conduction problem

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} = \hat{\mathbf{F}} \quad (7.6.1)$$

and Eq. (7.4.22) defining the mechanical equilibrium

$$\mathbf{K}(\mathbf{u}, \mathbf{T})\mathbf{u} = \mathbf{F} \quad (7.6.2)$$

the solution procedure is fairly evident. The conduction problem, being independent of any mechanical variables, is solved first over the time interval of interest. Temperature data are passed to the solid mechanics equations and these are solved for the displacement, strain, and stress fields over the temperature history. Since (7.6.2) is a quasi-static model, increments of the structural load is substituted for a time integration procedure. The temperature field at any given load step can be found by interpolation of the finite element solution. This type of problem could also be solved through a fully coupled procedure though this is not a computationally attractive alternative. Since the thermal diffusion process has the smaller time constant (compared to the infinite time constant for the solid mechanics problem), the thermal problem must be solved a fairly large number of times over the time interval. A fully coupled algorithm would thus force the solid mechanics problem to be solved many more times than would normally be required to achieve an accurate stress solution. The additional solutions of (7.6.2) make this approach ill-advised.

The coupling between Eqs. (7.6.1) and (7.6.2) increases in strength when mechanical dissipation is important and/or very large deformations occur in the structure such that the geometric definition of the heat conduction problem is altered. Mechanical dissipation appears as a volumetric source term in the conduction equation and is directly dependent on the stress and strain rate fields. That is, the force vector in (7.6.1) is $\hat{\mathbf{F}}(\sigma, \dot{\epsilon})$. Depending on the dissipation rate the problem may be either weakly or strongly coupled. If the heat generation rate is slow and temperature effects in the solid negligible, the solid mechanics problem could be solved over the loading history of interest to produce a dissipation history. The dissipation history could then be used to solve the conduction problem for the temperature response of the region. As the dissipation rate increases and temperature dependence increases, the coupling is more pronounced and Eqs. (7.6.1) and (7.6.2) must be solved more implicitly. A fully coupled method can be used, though a cyclic algorithm is more cost effective. Again, the conduction problem sets the time scale (time step) for the integration; the solid mechanics problem is solved after each thermal time step or whenever the fields change sufficiently to require an update in the dissipation. This decision on the frequency of solution for a subordinate process is one of the more difficult judgments to make in the use of cyclic or staggered solution strategies.

The coupling between (7.6.1) and (7.6.2) is very strong when large changes in geometry occur in the solid and the boundary conditions in the thermal problem are affected. If radiation is part of the thermal problem geometric changes imply changes in the radiation view factors and a redistribution of surface flux. Also, if deformation leads to new contacts between surfaces or the separation of previously contacting surfaces, the thermal conditions may be significantly altered. This type of coupling may be effectively treated with a cyclic algorithm, though the structural solution and geometrical updates would usually have to be computed at every thermal time step or whenever deformation was significant. The recomputing of radiation view factors for a dynamic geometry is a significant computational burden and makes this type of problem quite challenging.

7.6.4 Heat Conduction - Electric Field Interactions 1 & 4

Resistive or Joule heating occurs when energy is dissipated by an electric current flowing through a conductor. This type of problem would be represented by the conduction equation in (7.3.7)

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} = \hat{\mathbf{F}}(\mathbf{J}) \quad (7.6.3)$$

and the steady current flow equation (7.5.52) and its finite element model (7.5.82)

$$\mathbf{L}(\mathbf{T})\mathbf{V} = \mathbf{F}_V \quad (7.6.4)$$

The variable dependencies are shown in (7.6.3) and (7.6.4). The volume heating in (7.6.3) is given by $Q_J = \sigma^{-1}\mathbf{J}^2$ and \mathbf{J} is related to the gradient of the electric potential, \mathbf{V} , which is the unknown in (7.6.4). The electrical conductivity is usually a function of temperature, thus making the diffusion operator in (7.6.4) dependent on \mathbf{T} .

If these dependencies are both present, Eqs. (7.6.3) and (7.6.4) can be most easily solved as a fully coupled system. An implicit time integration procedure applied to (7.6.3) results in a generally nonlinear matrix problem as described in Chapter 3. This matrix can be combined with (7.6.4) to represent the two scalar equations. At each time step (or iteration, if the problem is steady), the combined nonlinear problem can be solved using either the Picard iteration if the nonlinearities are fairly mild or Newton's method for stronger nonlinear behavior. This problem can also be solved via a decoupled or cyclic procedure. Because (7.6.3) is time dependent, the conduction problem is the master process and sets the time scale for the problem. At a given time step, the known temperature field can be passed to the current flow equation, the temperature-dependent conductivity evaluated, and the current density and Joule heating computed. The Joule heating is transferred back to the conduction equation to allow the next update on the temperature field. This process may be repeated until convergence occurs at the current time; in many cases the changes that occur over a time step are sufficiently small that only one iteration through the data exchange is necessary to maintain acceptable accuracy.

7.6.5 Heat Conduction – Electromagnetic Field Interactions 1 & 4 & 5

Isolated magnetic fields do not generate any significant interactions with a thermal field, though the full coupling of both electric and magnetic fields with temperature effects describe a number of engineering processes of importance. Induction heating for melting and surface treatment (hardening) are commonly used processes that depend on this type of coupling. Eddy current analysis for the performance of electromagnetic devices, such as motors and generators, is another source of coupled problems. In the present description the focus will be on induction heating where extremes in the interaction are observed; the less demanding thermal performance problems will be subsets of this type of coupling.

To describe the nonisothermal eddy current problem, the conduction equation (7.3.7) is required

$$\mathbf{M}\dot{\mathbf{T}} + \hat{\mathbf{K}}\mathbf{T} = \hat{\mathbf{F}}(\mathbf{J}) \quad (7.6.5)$$

as is the quasi-static, electromagnetic system defined by the field problem in Eqs. (7.5.29) and (7.5.30) and the corresponding finite element model in equations (7.5.69) and (7.5.70)

$$\mathbf{M}(\mathbf{T})\dot{\mathbf{A}} + \mathbf{K}(\mathbf{T}, \mathbf{B})\mathbf{A} + \mathbf{N}(\mathbf{T})\dot{\mathbf{v}} = \mathbf{F}_A \quad (7.6.6)$$

$$\mathbf{N}^T(\mathbf{T})\dot{\mathbf{A}} + \mathbf{L}(\mathbf{T})\dot{\mathbf{v}} = \mathbf{F}_V \quad (7.6.7)$$

The variable dependencies are shown in Eqs. (7.6.5)–(7.6.7), which are written for the general case where the electrical problem is potential or voltage driven. When the problem is current driven, Eq. (7.6.7) is no longer needed and the $\dot{\mathbf{v}}$ term in (7.6.6) is known and associated with the known current density. These subcases were covered previously in Section 7.5.3. The volume heating in (7.6.5) is again given by the Joule heating relation

$$Q_J = \sigma^{-1} \mathbf{J}^2$$

and \mathbf{J} is related to the combination of the gradient of the electric potential, \mathbf{V} , if it is present, and the time derivative of the magnetic vector potential [see Eq. (7.5.28) and following]. The time derivative term is the induced or eddy current due to the fluctuating magnetic field. The dependencies in Eqs. (7.6.6) and (7.6.7) correspond to the possible variation of the electrical conductivity with temperature and the magnetic permeability with both temperature and the magnetic flux density.

In ferromagnetic materials the variation of the magnetic permeability is a major complicating factor in the analysis since it is a hysteretic function of \mathbf{B} and \mathbf{H} which is scaled by the temperature. The magnetic permeability may vary by three orders of magnitude over typical magnetic field strengths (at moderate temperatures) while above the Curie temperature the magnetic permeability reverts to a constant equal to free space permeability. A final complication in this type of coupled problem stems from the limited penetration into the material of the induced current fields and Joule heating when the applied fields are moderate to high frequencies. This skin depth effect implies that the simulation regions that are critical to the heat transfer part of the problem do not necessarily overlap all of the electromagnetic regions of importance.

In many applications the electromagnetic fields are time harmonic and the phasor representation of (7.6.6) and (7.6.7) [Eqs. (7.5.84) and (7.5.85)] are the appropriate finite element equations with similar functional dependencies. Even though the electromagnetic problem is nonlinear, a phasor representation can be used with single frequency driving fields if material properties are treated appropriately. This method is illustrated in [19]. Though the coupling is very strong, the appropriate solution strategy for Eqs. (7.6.5)–(7.6.7), or their time harmonic equivalents, is a cyclic procedure due to the vastly different time scales for most applications. The diffusion time constant for the heat conduction problem is significantly longer than equilibration of the induced electromagnetic fields; the electromagnetic fields are virtually constant over the diffusion time interval even if they are periodic. The heat conduction problem, therefore, is the master process and sets the time integration procedure. At a given time step, the current temperature field is passed to the electromagnetic problem. If the time-dependent forms of (7.6.6) and (7.6.7) are used, the fields are integrated forward to the end of the time interval using

an appropriately small time step. The Joule heating history is computed from the time-dependent fields and integrated to provide an energy rate over the time step for each resistively heated finite element. When the time harmonic forms of (7.6.6) and (7.6.7) are used, the procedure is similar except that a time independent, often nonlinear field problem must be solved and the resulting complex variables must be manipulated to generate the Joule heating. After the electromagnetic fields are updated, the new Joule heating data are used as a source term for the next conduction step. Note that in some applications, the geometry of the problem may change with time. This is not usually a difficulty because both the conduction and electromagnetics problems can be solved in a Lagrangian reference frame making geometric updates straightforward in the coupled situation. When radiation is a consideration, changing geometries will increase the computational work for the heat transfer solution step.

Even though the electromagnetic problem is the subordinate process and is simply providing the “loads” for the thermal problem it is, in this case, the more difficult problem. The inclusion of the gauge condition (7.5.48) makes the solution of the magnetic vector potential equation very difficult in much the same way as the incompressibility constraint hinders the solution of the viscous flow problem. In addition, the large changes in magnetic permeability with both field strength and temperature, plus the hysteretic nature of the property variation, generate strong material nonlinearities that inhibit convergence. Finally, though the time harmonic case is simplified in the time domain, the use of complex variables increases significantly the computational burden due to the matrix and variable vector sizes.

7.6.6 Viscous Flow – Quasi-Static Solid Interactions 2 & 3

Fluid structure interaction problems are quite common and this type of coupling has been well studied. In many simulations the fluid model can be simplified to a potential flow where only pressure loads on the structure are of interest and fluid separation is not an important feature of the flow. The coupling of the fluid potential equation with the solid mechanics problem will not be considered here as the main interest remains the viscous flow problem. The major difficulties in coupled viscous fluid and solid mechanics problems stem from the inherently different coordinate descriptions used in the two fields and the possible variations in defining a dominant physical process.

Because the preferred description for solid mechanics is Lagrangian and the usual description for fluids is an Eulerian coordinate system, some compromise must be found for a useful coupled problem description. The standard solution is to work in the usual Lagrangian reference frame for the solid and specify the fluid motion in an Arbitrary Lagrangian Eulerian (ALE) framework, as was outlined in Section 4.10.3. This approach has the advantage of maintaining an accurate definition of the fluid/solid interface which is the primary coupling mechanism in this type of problem. The possible large deformation of the solid is well captured by this approach and the use of appropriate strain measures in the solid mechanics formulation. Also, the ALE method with a mesh moving scheme coupled to the interface motion (which is determined from the solid mechanics problem), provides a robust and accurate algorithm for the fluid motion. Many of the details associated with ALE methods were described in Section 4.10.3 and will not be repeated here.

The second area of difficulty is to determine the primary physical process so that a solution algorithm for the coupled system can be constructed. Unfortunately, the fluid motion can drive the solid deformation or the solid motion and deformation can drive the fluid motion. Since neither process is always dominant a fully coupled algorithm is suggested.

To describe the fluid structure interaction problem the viscous flow model from equations (7.3.9) and (7.3.10) is required,

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (7.6.8)$$

$$\mathbf{M}\dot{\mathbf{u}} + \mathbf{C}(\mathbf{v}^m)\mathbf{v} + \mathbf{K}\mathbf{v} - \mathbf{Q}\mathbf{P} = \mathbf{F}(\mathbf{v}^m) \quad (7.6.9)$$

which is written here in its isothermal form. Coupling with a nonisothermal flow is certainly possible, in which case the energy equation from (7.3.11) would be added to the above and a heat conduction equation would be required for the solid region. This complication adds little to the overall algorithm and therefore will not be explicitly considered. The quasi-static solid mechanics problem is again given by Eq. (7.4.22)

$$\mathbf{K}(\mathbf{v}, \mathbf{v}^m, \mathbf{T})\mathbf{v} = \mathbf{F}(\mathbf{P}, \mathbf{u}) \quad (7.6.10)$$

For ease of notation it is assumed that the solid mechanics problem is written in terms of velocities; the use of the same matrix and vector symbols in (7.6.9) and (7.6.10) does not imply that these are the same matrices. The variable dependencies are indicated in (7.6.8)–(7.6.10) and are seen to be a strong function of the mesh movement \mathbf{v}^m (interface geometry) and the fluid loads on the structure.

As noted above, the most general solution procedure for this type of problem is a fully coupled method in which Eqs. (7.6.8)–(7.6.10) are solved in a single matrix problem. A time integration scheme would be applied to the fluid equations while the static solid equations would have the appropriate loads applied for the current time. The Picard or Newton's method could be used to converge the solution at the current time. The fluid and solid mechanics regions do not overlap; the entire domain is meshed by a single finite element discretization. If a pseudo-structural mesh movement scheme is employed as part of the ALE method, then solid mechanics equations are solved over the entire domain. The mesh movement equations may be fully coupled with the fluid and solid equations or may be decoupled and used to update the fluid mesh region in a subcycle. For efficiency, this fully coupled method assumes that the time scales for the fluid and solid regions are not too different, which they probably cannot be, due to the strong coupling at the boundary. This algorithm does pay a penalty in the matrix solution because the fluid equations are nonsymmetric while the solid equations are symmetric and could be solved with less computational effort.

7.6.7 Viscous Flow – Electric Field Interactions 2 & 4

There are a number of problems that involve a viscous fluid and an electric field, most of which are unfamiliar to the applied mechanics community. Electrohydrodynamics involves the motion of a fluid due to an electric pressure which is proportional to the square of the applied electric field. The fluid may be either conducting or a dielectric. The process of flow electrification involves a dielectric fluid and the convection of charge from the double layers that form near boundaries. This type of problem is

common in electrochemical devices such as batteries. A more familiar problem, and one that will be considered here, is the flow of an electrorheological fluid. In this case an applied electric field causes a large increase in the viscosity of the fluid, which can act as a control mechanism for the flow. The electrorheological application is not a particularly difficult type of coupled problem since the coupling is essentially in one direction with the electric field influencing the flow.

The equations for this type of problem include the isothermal, viscous flow equations from (7.3.9) and (7.3.10) or

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (7.6.11)$$

$$\mathbf{M}\dot{\mathbf{u}} + \mathbf{C}\mathbf{v} + \mathbf{K}(\mathbf{E})\mathbf{v} - \mathbf{Q}\mathbf{P} = \mathbf{F} \quad (7.6.12)$$

and the potential equation for the electric field from Eq. (7.5.82)

$$\mathbf{L}\mathbf{V} = \mathbf{F}_V \quad (7.6.13)$$

The viscous term in Eq. (7.6.12) is a function of the electric field, which is computed from the gradient of the electric potential, the unknown in (7.6.13). The master process is the viscous flow problem which is integrated in time by standard methods. When a voltage is applied in some region of the problem, the potential equation must be solved for the instantaneous electric field. These field data are then transferred to the flow equation to allow the viscosity function to be evaluated for the next time step. The process is continued as long as a voltage is applied to the problem. The subordinate process in this case involves a series of steady solutions with the boundary condition (voltage) having a time variation.

7.6.8 Viscous Flow – Electromagnetic Field Interactions 2 & 4 & 5

The topical area of magnetohydrodynamics (MHD) brings together the descriptions of nonisothermal, fluid mechanics and the electromagnetics of conducting materials. The MHD field includes a variety of problem types many of which are quite complex especially with regard to coupling and property variations. Rather than attempt a superficial coverage of general MHD problems, we will narrow the focus here to a class of problems that is important in material processing and manufacturing applications. Processes such as high current melting and inductive stirring fall into this problem category. The major assumption involved in this type of problem is that the magnetic Reynolds number is small. The magnetic Reynolds number is usually defined as $Re_m = UL\sigma\mu_m$, where σ is the electrical conductivity, μ_m is the magnetic permeability, and U and L are a representative flow velocity and length scale, respectively. The magnetic Reynolds number represents a ratio of magnetic convection to magnetic diffusion. For many common processes involving liquid metals, the magnetic Reynolds number is small, which implies that the magnetic field lines are unaffected by the flow field. The opposite extreme of a high magnetic Reynolds number ensures that the magnetic field lines are strongly convected by the flow field. This limit is typical of fusion applications and problems in astrophysics.

With the small magnetic Reynolds number assumption, the equations required for the MHD problem include the nonisothermal, viscous flow equations

$$-\mathbf{Q}^T \mathbf{v} = \mathbf{0} \quad (7.6.13)$$

$$\mathbf{M}\dot{\mathbf{u}} + \mathbf{C}\mathbf{v} + \mathbf{K}\mathbf{v} - \mathbf{Q}\mathbf{P} + \mathbf{B}\mathbf{T} = \mathbf{F}(\mathbf{J}, \mathbf{B}) \quad (7.6.14)$$

$$\mathbf{N}\dot{\mathbf{T}} + \mathbf{D}\mathbf{T} + \mathbf{L}\mathbf{T} = \mathbf{G}(\mathbf{J}) \quad (7.6.15)$$

and the potential forms of the electromagnetic equations

$$\mathbf{M}(\mathbf{T})\dot{\mathbf{A}} + \mathbf{K}(\mathbf{T})\mathbf{A} + \mathbf{N}(\mathbf{T})\dot{\mathbf{v}} = \mathbf{F}_A \quad (7.6.16)$$

$$\mathbf{N}^T(\mathbf{T})\dot{\mathbf{A}} + \mathbf{L}(\mathbf{T})\dot{\mathbf{v}} = \mathbf{F}_V \quad (7.6.17)$$

In the flow equations (7.6.13)–(7.6.15), the main electromagnetic coupling is through the resistive or Joule heating in the energy equation (which is proportional to the square of the current density) and the Lorentz body force in the momentum equation (which is proportional to the current density and the magnetic flux density). The electromagnetic equations are coupled to the flow problem primarily through temperature-dependent properties; a change of phase in the flow region may require some adjustment in geometry and boundary conditions to account for expansion or contraction of the fluid region. The small magnetic Reynolds number assumption produces the reduced coupling of the electromagnetics to the flow problem. If this assumption is not invoked, the magnetic equation (7.6.16) would have an additional term that depends on the fluid velocity.

Equations (7.6.13)–(7.6.17) form a strongly coupled set when all of the dependencies are present. Though a fully coupled algorithm is attractive from the convergence point of view, the very large size of the system and the generally differing time scales make such an approach impractical. In many applications, the electromagnetic fields are slowly varying or steady and the flow process will set the time scale and be the dominant process. The fact that the fluid and thermal “loads” come from the electromagnetics, while only properties vary in the other coupling direction, also argues for this choice of master and slave process. The standard staggered algorithm would thus proceed with the current temperature field being transferred to the electromagnetics equations (7.6.16) and (7.6.17). A time integration or steady solution for the electric and magnetic fields would produce the needed Joule heating and Lorentz forces. Transferring these data back to the nonisothermal flow equations would allow the next fluid and thermal fields to be computed. In the limit of temperature-independent electromagnetic properties and steady EM fields, Eqs. (7.6.16) and (7.6.17) reduce to a magnetostatics description, which only needs to be solved once for resistive heating and body forces. Slowly varying electric and magnetic fields could also be treated as a series of steady states that may be precomputed and made available to the flow problem as a loading history.

7.6.9 Quasi-Static Solid – Electromagnetic Field Interactions 3 & 4 & 5

The interaction of solid mechanics with electromagnetics will not be treated here in detail because it is outside the scope of the text. It is sufficient to note that these types of interactions can produce coupled problems that are similar in type and complexity to those described in the previous sections. Electric fields may cause stresses and deformations in a solid (electrostriction) and deformation may induce an electrical response (piezoelectric effect). These types of interactions are primarily through the constitutive relations for the material. Also, a strong magnetic field may produce a magnetic stress (magnetostriiction) as described by the Maxwell stress tensor. The algorithms for coupling these types of interactions would certainly be similar to the fluid and thermal procedures outlined above.

7.7 Implementation of Coupled Algorithms

A primary feature of coupled field problems in mechanics is the almost endless variety of possible interactions. This diversity plus the accompanying distribution of workable numerical algorithms, as illustrated in the previous sections, complicates the construction of finite element software. Though this text is not oriented toward the descriptions of code architectures and finite element implementation issues, it is important to discuss some general aspects of coupled problem implementation.

The first choice in implementing a coupled or multiphysics solution method is whether the algorithm will be in a single code or multiple software packages. For strongly coupled problems that will be solved in a fully coupled method, the choice is obviously a single code. In many ways this is the simplest type of implementation since only one matrix problem needs to be considered and all the equations are solved at once. Finite element interpolation order for the various fields, equation ordering within the matrix, type of integration and/or iterative method demand consideration but are not major impediments to implementation. The major drawback to this approach is that the code does one and only one type of coupled problem.

A second approach to coupling involves a single code or software package that is composed of modules for the different types of physical phenomena that are to be modeled. The coupling or required data exchanges are then orchestrated from a driver routine. This type of design would work for either strongly or weakly coupled problems since any strongly coupled interaction can be solved iteratively. The interactions in this case are predefined in terms of what data will be transferred from one process to another; new types of interactions require code modifications that may be quite extensive if the alterations are in constitutive behavior rather than boundary or source terms. An advantage in this design is the sharing of finite element infrastructure between modules. An element library, solver package, and data management routines can be made common between physics modules and reduce code maintenance. Since the individual modules may be run independently, the overall utility of the software is increased significantly over the previous, fully coupled approach. The disadvantage of the multiple module implementation is the size of the overall code, the number of modules that may be required for general applications, and designing enough flexibility into the driver to anticipate the numerous types and strengths of interactions that may be needed.

A final methodology that strongly resembles the multiple module concept introduces the idea of coupling separately maintained mechanics codes into an interacting package. No driver routine is needed though the code containing the dominant mechanics process will act as the master routine. The secondary mechanics codes will act as slave processes and provide data on the demand of the master. For such a design to be feasible, the individual codes must be sufficiently general to make proper use of any externally supplied data and return needed quantities to the master. Also, since no assumptions about which physical process will be dominant in any particular application, all codes should be constructed to function in either the master or slave role. The communication between individual codes could be achieved through data file transfers. Another alternative is the use of message passing software developed for parallel computing environments. Message passing utilities, such as the PVM (Parallel Virtual Machine) libraries [21] or the MPI (Message Passing Interface) libraries [22], allow the memory to memory

transfer of data, while each code is running on its own processor. The advantage of the independent code design is that the very substantial investment in stand alone mechanics code development can be readily applied to coupled problems with relatively minor software modifications. Stand alone mechanics codes also tend to contain more extensive capabilities and more robustness for an application than a module in a more monolithic code. The seamless coupling of such specialized codes provides an increase in overall capability. The drawbacks to this design are the need to set some standards for data transfer protocols and develop some general interface routines that use either file reads and writes or the message passing utilities.

In addition to the overall architecture, there are a number of algorithm details that must be resolved and implemented during the development of a coupled problem or multiphysics capability. Many of these items have been mentioned throughout this chapter but the major issues will be collected here as a series of questions to summarize the subject.

1. How many physical processes will be allowed in any simulation? Though coupling of only two processes have been described above, there are a variety of important problems in which three physical phenomena can be coupled.
2. What types of coupling will be allowed? Will the data dependencies be in source terms, constitutive relations, boundary conditions, geometric changes, or all of these areas?
3. What types of data will be exchanged between processes? Will the coupling be limited to nodal variables, or will element and global data also be permitted in the exchange?
4. How will the dominant or master process be selected? Will this be predetermined so that only certain types of physics may drive the simulation or will any process be selectable as the master?
5. What assumptions will be made regarding the domains for the different physics? Will these areas be forced to coincide or will they be completely arbitrary in extent and overlap? Will the mesh discretizations be the same for each physical process or will data interpolation be required?
6. How will the solution procedures for each process be synchronized? Will only similar time dependencies, both steady or time dependent, be allowed in the coupling or will any useful combination be permitted? How will the time integration processes in each process be allowed to influence each other? How will the frequency of data exchange be determined?

As was outlined in the individual discussions of the various types of coupled heat transfer and fluid mechanics problems, the answers to the above questions can vary considerably. In any given situation the answers are fairly obvious but when planning a general capability the decisions are more complicated. Though complete generality in multiphysics simulations has not as yet been achieved, considerable progress has been made in linking the most important types of interactions together to obtain more realistic finite element solutions.

7.8 Numerical Examples

7.8.1 Introduction

The coupled problems illustrated in this section are relatively complex engineering analyses that are intended to demonstrate some of the multiphysics areas described in this chapter. Due to space limitations, each problem is simply outlined and some representative results described. The finite element codes used in these computations and the method of code coupling vary from example to example. In the first example of a thermal stress problem, the COYOTE [23] heat transfer code was connected via subroutine calls to the quasi-static solid mechanics code, JAS [24]. The second and fourth examples involving electromagnetics employed individual codes linked by the PVM [21] message passing utilities. The second problem of induction heating used the thermal code COYOTE [23] and the electromagnetics code TORO [25], while the remelting problem was solved with the flow code NACHOS II [26] connected to TORO. The third example was solved with the flow code GOMA [27], which has a solid mechanics and ALE capability included as part of a fully coupled algorithm.

7.8.2 Thermal-Stress Example

The thermal-structural interaction example is a geometrically simple problem with a fairly complex material response. A cylindrical canister is filled with a foam-like material; located within the foam-like material is a solid of rectangular cross section. The canister is subjected to a thermal boundary condition that is representative of a heat source at one end of the canister. As the temperature of the container rises, the foam-like material begins to decompose, producing an off-gas and the eventual disappearance of the foam-like material. The container is pressurized by the gas production causing deformation of the canister wall and changes in the geometry for the thermal analysis. Figure 7.8.1 shows a meshed, cut away view of the canister and its contents at the initial time.

A full three-dimensional simulation of this problem was produced using a thermal analysis code as a master process and coupled to the quasi-static, solid mechanics code through a subroutine interface. A chemical kinetic description was used to detail the heat and mass transfer due to the decomposition of the foam-like material. The chemistry was solved in conjunction with the temperature field via an operator splitting technique. As the foam-like material completed the decomposition reaction in each element, the element was removed from the simulation (element death). Mass transfer from the decomposition reaction was accounted for in the expanding clear space of the container through use of a bulk node (see Section 3.11.4). Radiation inside the container was included in the analysis with view factors being constantly updated due to element death and changes in the shape of the container walls. The pressure computed in the bulk node was passed to the solid mechanics code for use as a boundary condition on the container walls. The temperature field was also transferred to the solid mechanics code for use in the temperature-dependent, elastic-plastic constitutive model of the container wall. The foam-like material was not modelled in the structural computation because of its very limited load carrying capacity. An updated, deformed geometry was transferred back to the thermal analysis code at the completion of each solution step.

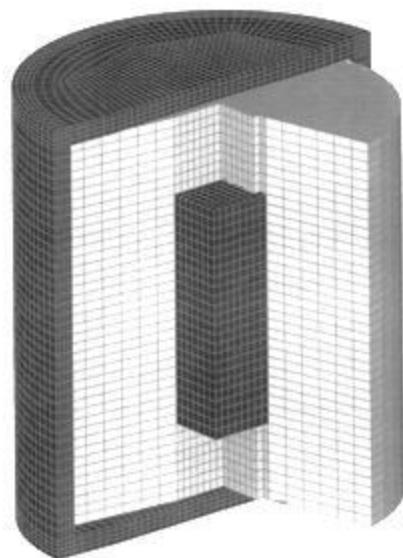


Figure 7.8.1: Schematic and mesh for heated canister.

The deformed geometries for the canister at several times during the heating process are shown in Figure 7.8.2. The foam-like material begins to disappear (element death) at 250 seconds after heating begins and creates a clear space at the bottom of the canister. Visible deformation begins at approximately 300 seconds and continues to a time of 450 seconds after which additional deformations are relatively small. The deformations in Figure 7.8.2 are scaled by a factor of 1.5.

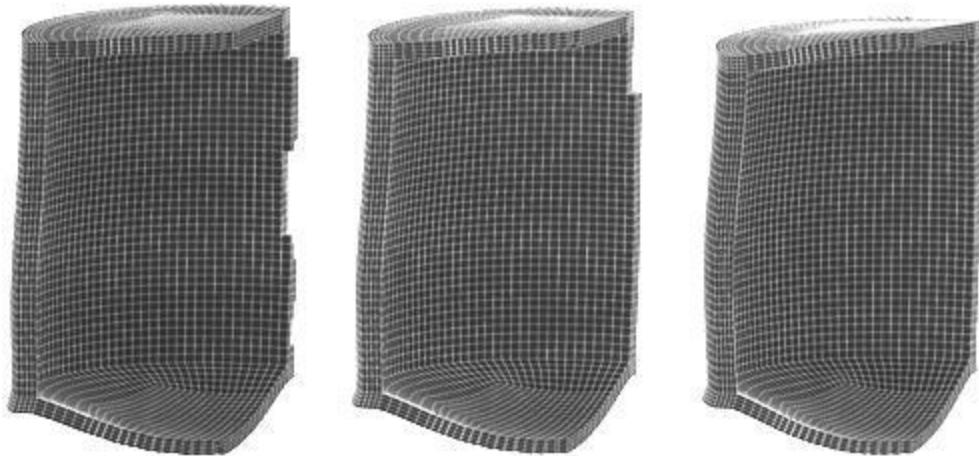


Figure 7.8.2: Deformed geometry for heated/pressurized canister.

The state of the foam within the canister is illustrated in the cut away plots in Figure 7.8.3 which are viewed from the heated end of the container. The recession of the foam-like material is easily seen in this figure, as only the currently active elements are plotted; an undeformed configuration of the canister is used for reference. At the last time shown, the block buried in the foam-like material is just starting to be exposed. Though the mesh discretization is not excessively refined, the complexity of the physical processes modeled required that a parallel computation be utilized to reduce the analysis time. This analysis is due to R. G. Schmitt (personal communication).

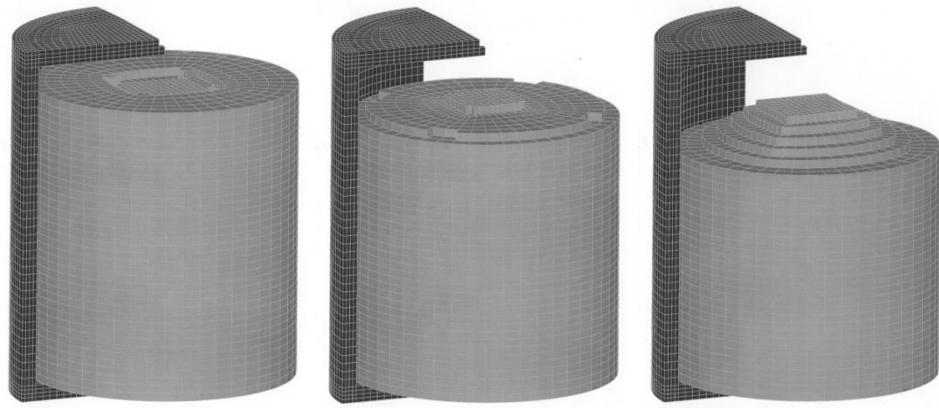


Figure 7.8.3: Material decomposition and removal for heated/pressurized canister; currently active elements are shown.

7.8.3 Thermal-Electromagnetic Example

Induction heating is used extensively in manufacturing processes, especially in the surface treatment (hardening) of ferromagnetic materials. In general terms, induction hardening consists of a part placed in the vicinity of a alternating current coil. The time varying coil current induces an eddy current in the part which in turn produces a resistive heating and temperature rise in the part. The high frequency of the applied coil current and shallow penetration of the magnetic field (skin depth effect) in the part, limit the thermal effect to the part surface. Complexities in this process include strongly temperature-dependent electromagnetic properties, a strongly nonlinear magnetic permeability and the relative motion between the part and the coil when a realistic industrial process is simulated.

Shown in Figure 7.8.4 is a schematic of a simplified induction hardening process in which a cylindrical part is heated within an annular coil of square cross section. The finite element mesh used for the axisymmetric analysis is shown in Figure 7.8.5. The time harmonic electromagnetics problem was solved over the domain shown in the figure where the current density was specified for the coil. The thermal problem was solved over the cylindrical part and the coil, each of which was subjected to

Joule heating; the air space was not included in the thermal analysis. Radiation between the part and coil could be included in the analysis but was omitted for this demonstration. For this application the thermal problem is the master process since the time constant for thermal diffusion is significantly longer than the time constant for induction. The thermal problem was integrated over the heating cycle of 5.4 seconds using an implicit integration method. At every time step, the current temperature field was transferred to the electromagnetics code via PVM and the (nonlinear) time harmonic eddy current problem solved for the Joule heating in the part and coil. For the present example, the coil current was 8 kA and was run at a frequency of 7.6 kHz. These data were returned to the thermal code via PVM and used in the next temperature solution. At the conclusion of the heating cycle the part was subjected to a spray quench which was applied via a convective heat transfer boundary condition on the vertical surface of the part.

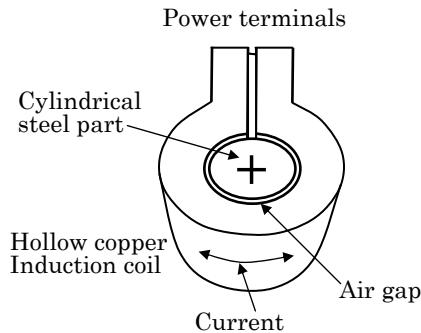


Figure 7.8.4: Schematic of induction hardening problem.

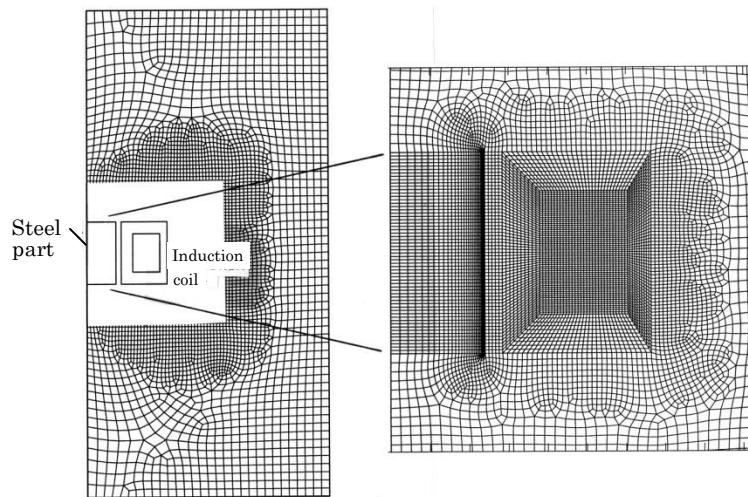


Figure 7.8.5: Finite element mesh for axisymmetric induction hardening problem.

Figure 7.8.6 shows computed and measured temperature histories for various locations within the midplane of the cylinder. The agreement between model and experiment is reasonably good and shows the rapid rise in surface temperature. Note that the surface temperature exceeds the Curie temperature at which point the magnetic permeability falls to a free space value. At quench, the surface temperature drops immediately while the internal temperatures continue to rise due to conduction.

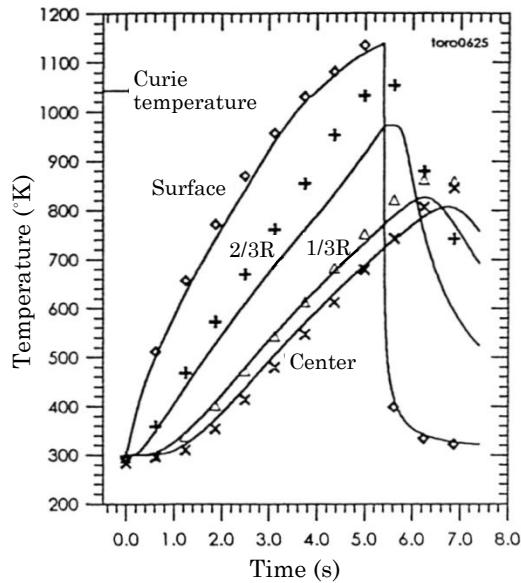


Figure 7.8.6: Predicted and measured temperature histories for induction heated cylindrical part.

In the real analysis setting, the thermal problem is augmented by a reaction kinetics solution that describes the phase transformation of the material and the hardness of the processed part. Ultimately, the coupled solution described here is also passed to a quasi-statics solid mechanics code that evaluates residual stresses and deformation due to the heat treatment. Further details on this problem are available in [28].

The same type of problem with a longer cylinder that translates along its axis, through the coil, was solved. A sliding surface was placed within the (air space) mesh between the coil and the cylinder. Along the slide surface a multipoint constraint was employed to enforce continuity of the magnetic vector potential. The mesh inside the slide surface moved with the translating cylinder while the mesh outside the slide surface remained stationary with the coil. The time-dependent motion of the cylinder was specified. The thermal problem required no constraint since the air was not included in the analysis. If radiation had been included in the analysis, continual updating of the view factors would be necessary to account for the motion of the cylinder.

Figure 7.8.7 contains a series of contour plots which illustrate the temperature field at several times during the motion of the rod through the coil. Motion of the cylinder begins 0.5 second after the coil current is applied and continues for 1

second after heating is terminated; the coil is energized for a period of 5 seconds. The process is sufficiently rapid so that the heat loss from the cylinder is minimal and no thermal boundary conditions were applied; no quench process was modeled in this example.

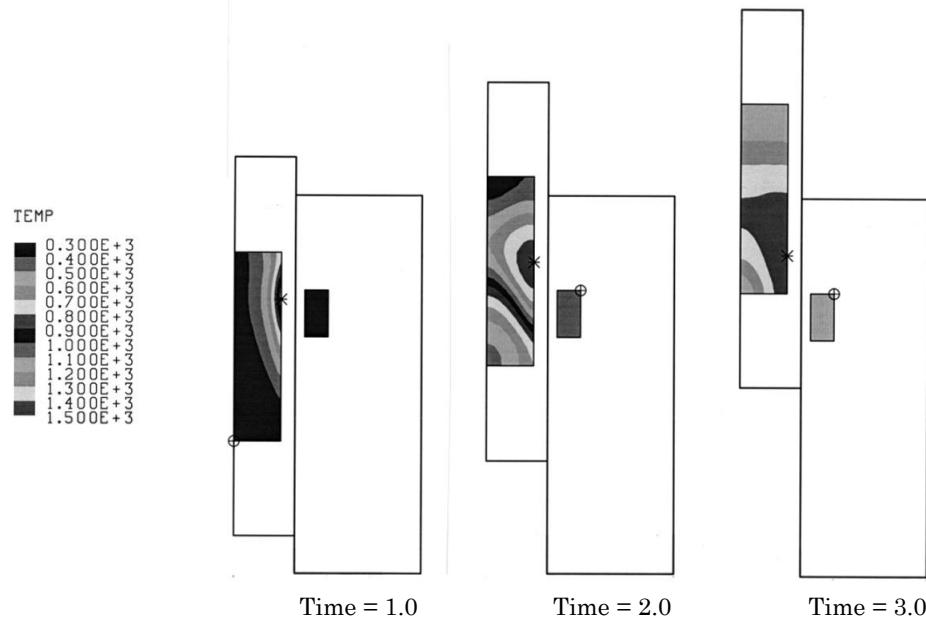


Figure 7.8.7: Temperature contours for a cylinder moving through an alternating current coil.

7.8.4 Fluid-Solid Interaction Example

As an example of the interaction between a viscous fluid and a solid, the problem of creating a drop by pushing fluid through an orifice is considered. A cross section of a typical axisymmetric geometry is shown in Figure 7.8.8 along with an initial mesh for the simulation. A piston is located in a small fluid reservoir and is moved toward the orifice at an almost constant rate. Fluid is pushed through the orifice forming an axisymmetric drop (no gravity). The boundary behind the piston is an inflow boundary that simulates the connection to a large reservoir. During the piston motion, the piston and reservoir wall are slightly deformed due to the relative thinness of the solid sections and the fluid pressure.

The drop formation problem was simulated using a fully coupled ALE technique to track the motion of the fluid free surface (with surface tension) and motion/deformation of the solid regions. Due to the large geometric change in the fluid region, mesh distortion became unacceptable at several times during the transient and remeshing of the fluid region was required. The solid was treated as an elastic material.

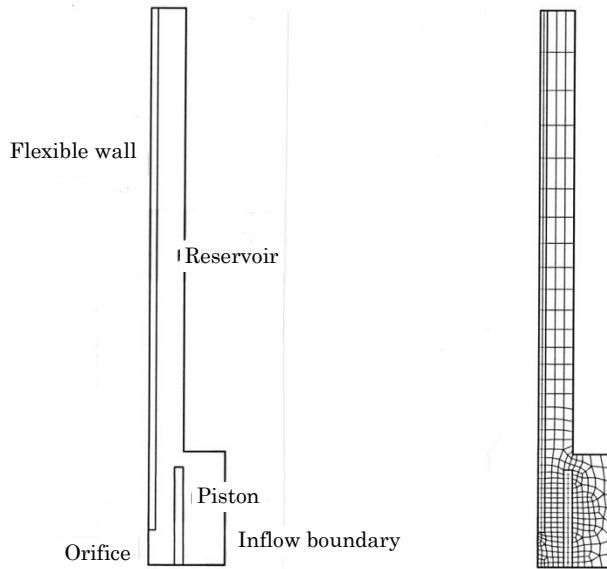


Figure 7.8.8: Schematic and initial finite element mesh for drop formation problem.

Shown in Figure 7.8.9 is a series of deformed mesh plots that illustrate the motion of the piston and the formation of a fluid drop. Necking of the drop is apparent in the second plot and reaches incipient pinch-off by the last plot. No mechanism for pinch-off is available in this method and the simulation was terminated at the last time shown. The deflection of the reservoir wall is not visible in the unmagnified plots of Figure 7.8.9; the deformation is shown at one time in Figure 7.8.10 where the displacements have been magnified by a factor of five. The deflection of the wall is obvious since the wall was initially parallel with the piston. Small deflections of the piston were observed early in the transient as the piston started to accelerate the fluid; these deflections were not as large as the bending of the reservoir wall.

7.8.5 Fluid-Electromagnetic Example

The last example in this section is a simulation of the vacuum arc remelt (VAR) process that involves nonisothermal viscous fluid flow, flow in a porous medium, solidification and Joule heating, and Lorentz forces from a magnetostatic field. The problem and solution techniques are described in detail in [29] and will only be summarized here. Figure 7.8.11 shows a schematic of the VAR process and a typical finite element mesh for the axisymmetric analysis of the solidifying ingot. In the VAR process, a high current is passed through a consumable electrode, creating a metal vapor plasma arc between the electrode and the melt pool contained within the crucible. The arc provides the energy for melting the small electrode.

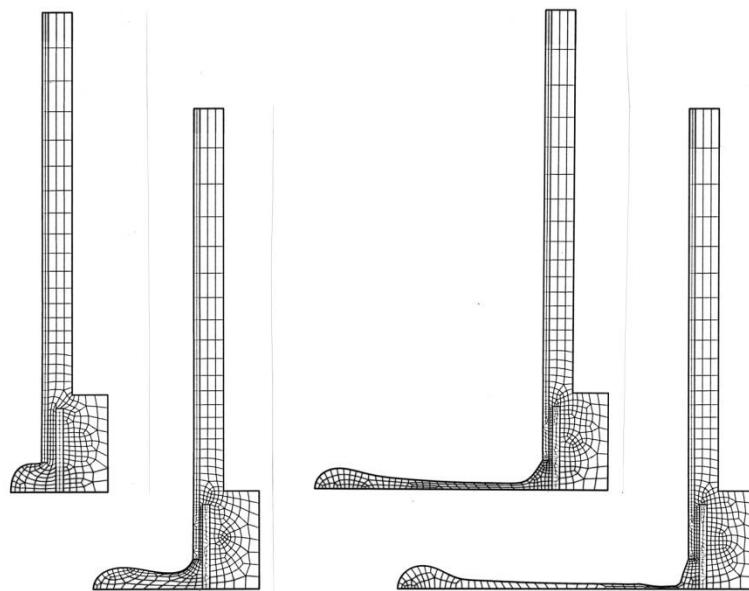


Figure 7.8.9: Finite element solution for drop formation problem. Remeshing occurs at three times during the simulation.

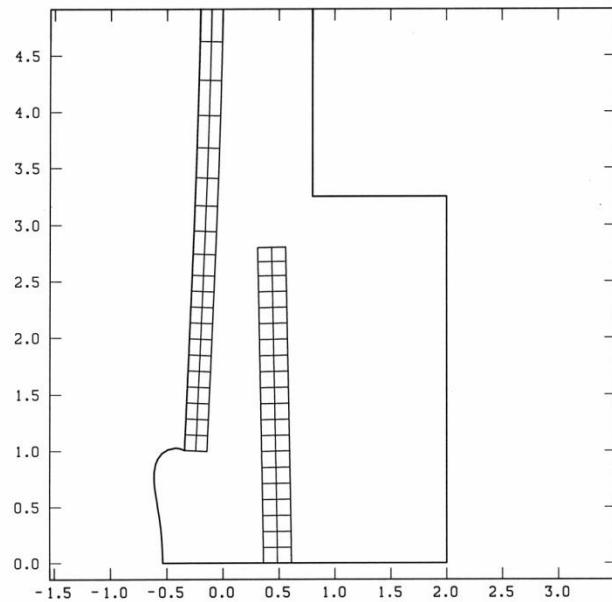


Figure 7.8.10: Reservoir wall deformation during drop formation; the displacements are magnified by a factor of five.

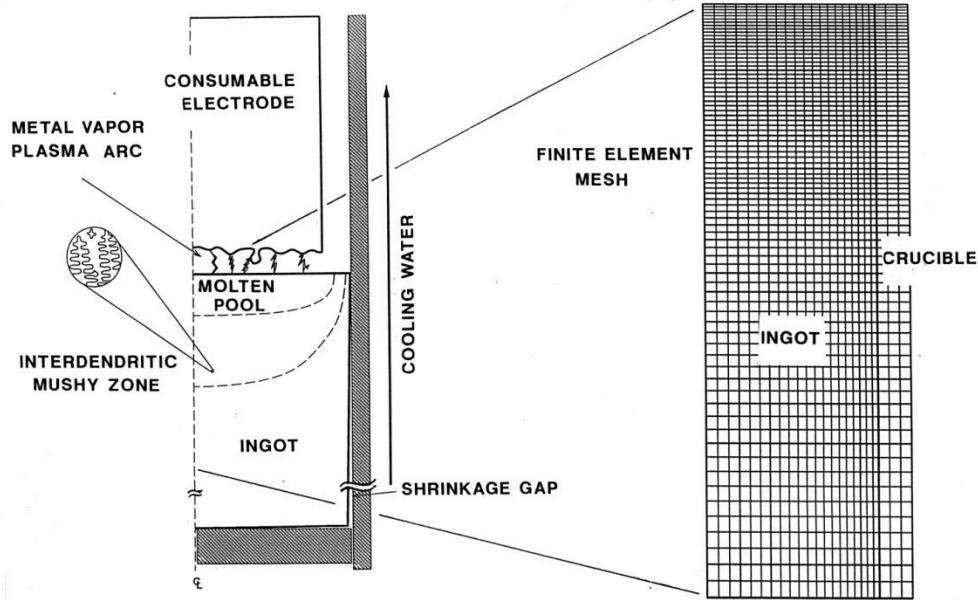


Figure 7.8.11: Schematic for vacuum arc remelt process and typical finite element mesh for crucible.

The objective is to create a larger ingot with better and more uniform metallurgical properties. In the finite element simulation of the solidifying ingot, the melt pool is represented as a viscous incompressible fluid with significant buoyancy forces; the mushy zone, which is typically present in alloy solidification, is modeled as a porous media using the Darcy–Brinkman equations of Section 4.4. Solid body conduction describes heat flow in the solidified portion of the ingot. The melt pool fluid is a conducting medium and the interaction of the current and magnetic field produces body forces and resistive heating within the crucible. A low magnetic Reynolds number is typical for this flow and the magnetostatics equations from Section 7.5 provide the appropriate description of the fields. The electromagnetic problem is weakly coupled to the flow problem since the properties are temperature and phase dependent. Boundary conditions are fairly complex and represent mass, energy, and current input from the arc and interface conditions with the crucible wall. As the ingot solidifies it shrinks in diameter and pulls away from the wall. This effect is modeled by increasing the resistance to heat transfer and electrically insulating the wall below the shrinkage point. A coupled solid mechanics solution would be required to eliminate this model parameter.

The algorithm used to solve this type of problem is a cyclic procedure with the fluid mechanics and heat transfer code being the master process; the magnetostatics code was invoked whenever an update on the volume heating or body force was required. Because time-independent solutions were of primary interest, a solution strategy involving zeroeth order continuation and combinations of the Picard and Newton iteration for the flow problem was adopted. The solution process was

delicate with small increments in surface heat flux from the arc being used to “advance” the solution while buoyancy and Lorentz forces were scaled to maintain a balance between these opposing effects. Data transfer between the two finite element solutions occurred at user-specified intervals and was processed by the PVM utilities. Contour plots for the main variables in a typical VAR simulation are shown in Figure 7.8.12.

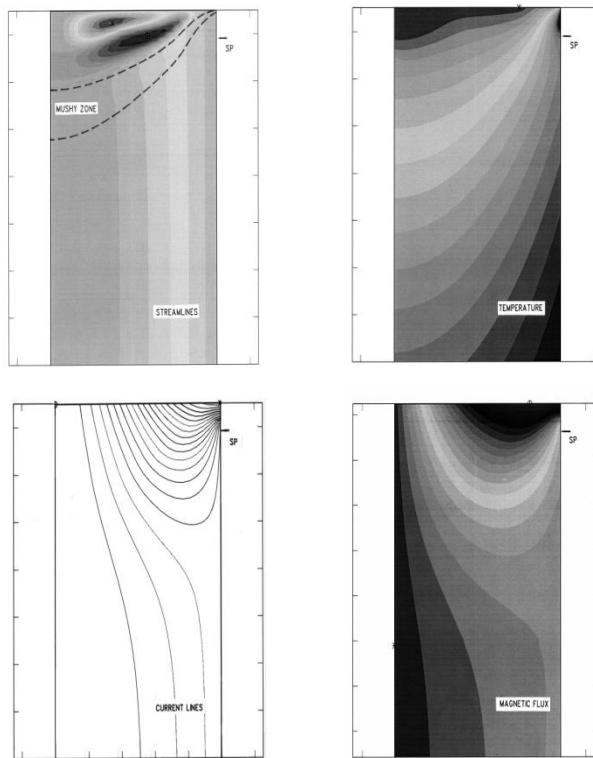


Figure 7.8.12: Contour plots for a typical VAR solution with buoyancy and Lorentz effects included.

Of primary interest is the stream function plot that illustrates the motion of the liquid metal from its small inflow at the top surface of the melt pool, through the clockwise rotation of the electromagnetically driven, Lorentz cell, the counterclockwise rotation of the thermally driven cell, through the porous layer, and finally as a solid body translation with the solid ingot. The temperature field shows some influence of the cell structure and the separation of the ingot from the wall. The electromagnetic fields are also strongly influenced by the wall boundary conditions. These results are for a fairly large current applied to the ingot. As the current is reduced, the thermal cell grows in size and limits the Lorentz cell to a thin region at the top of the melt pool. Without the Lorentz force, very large thermal cell solutions with unrealistically large melt pools are predicted. Additional details on modeling of the VAR process can be found in [29,30].

References for Additional Reading

1. O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, *The Finite Element Method: Its Basis and Fundamentals*, 6th ed., Elsevier, New York (2005).
2. R. D. Cook, D. S. Malkus, and M. E. Plesha, *Concepts and Applications of Finite Element Analysis*, John Wiley & Sons, New York (1989).
3. K. J. Bathe, *Finite Element Procedures*, Prentice-Hall, Englewood Cliffs, NJ (1996).
4. M. A. Crisfield, *Non-linear Finite Element Analysis of Solids and Structures, Volume 1 - Essentials*, John Wiley & Sons, Chichester, U.K. (1991).
5. M. A. Crisfield, *Non-Linear Finite Element Analysis of Solids and Structures, Vol. 2: Advanced Topics*, John Wiley, Chichester, UK (1997).
6. T. Belytschko, W. K. Liu, and B. Moran, *Nonlinear Finite Elements for Continua and Structures*, John Wiley, Chichester, UK (2000).
7. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).
8. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis*, Oxford University Press, Oxford (2004).
9. J. N. Reddy, *An Introduction to Continuum Mechanics with Applications*, Cambridge University Press, New York (2008).
10. J. Bonet and R. D. Wood, *Nonlinear Continuum Mechanics for Finite Element Analysis*, Cambridge University Press, New York (1997).
11. L. E. Malvern, *Introduction to the Mechanics of a Continuous Medium*, Prentice-Hall, Englewood Cliffs, New Jersey (1969).
12. ABAQUS Theory Manual, Version 5.8, Hibbit, Karlsson & Sorenson, Inc., Pawtucket, Rhode Island (1998).
13. J. N. Reddy, *Energy Principles and Variational Methods in Applied Mechanics*, 2nd ed., John Wiley & Sons, New York (2002).
14. W. F. Hughes and F.J. Young, *The Electromagnetodynamics of Fluids*, John Wiley & Sons, New York (1966).
15. J. D. Jackson, *Classical Electrodynamics*, Second Edition, John Wiley & Sons, New York (1975).
16. J. Jin, *The Finite Element Method in Electromagnetics*, John Wiley & Sons, New York (1993).
17. M. N. O. Sadiku, *Numerical Techniques in Electromagnetics*, CRC Press, Boca Raton, Florida (1992).
18. K. J. Binns, P. J. Lawrence, and C. W. Trowbridge, *The Analytical and Numerical Solution of Electric and Magnetic Fields*, John Wiley & Sons, New York (1992).
19. P. P. Silvester and P. L. Ferrari, *Finite Elements for Electrical Engineers*, Third Edition, Cambridge University Press, Cambridge, U.K. (1996).
20. M. V. K. Chari, A. Konrad, M. A. Palmo, and J. D'Angelo, "Three-Dimensional Vector Potential Analysis for Machine Field Problems," *IEEE Transactions on Magnetics*, **MAG-18**, 436–446 (1982).
21. A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. Sunderam, *PVM: Parallel Virtual Machine, A User's Guide and Tutorial for Network Parallel Computing*, MIT Press, Cambridge, Massachusetts (1996).
22. W. Gropp, E. Lusk, and A. Skjellum, *Using MPI, Portable Parallel Programming with the Message Passing Interface*, MIT Press, Cambridge, Massachusetts (1995).

23. D. K. Gartling, R. E. Hogan, and M. W. Glass, "COYOTE - A Finite Element Computer Program for Nonlinear Heat Conduction Problems," Sandia National Laboratories Report, SAND94-1173 and SAND94-1179, Albuquerque, New Mexico (2000).
24. M. L. Blanford, "JAS3D A Multi-Strategy Iterative Code fro Solid Mechanics Analysis, User's Instructions, Sandia National Laboratories Report, in progress, Albuquerque, New Mexico (2000).
25. D. K. Gartling, "TORO - A Finite Element Computer Program for Nonlinear Quasi-Static Problems in Electromagnetics," Sandia National Laboratories Report, SAND95-2472 and SAND96-0903, Albuquerque, New Mexico (2000).
26. D. K. Gartling, "NACHOS II - A Finite Element Computer Program for Incompressible Flow Problems," Sandia National Laboratories Report, SAND86-1816 and SAND86-1817, Albuquerque, New Mexico (1986).
27. P. R. Schunk, P. A. Sackinger, R. R. Rao, K. S. Chen, R. A. Cairncross, T. A. Baer, and D. A. Labreche, "GOMA 2.0 - A Full-Newton Finite Element Program for Free and Moving Boundary Problems with Coupled Fluid/Solid Momentum, Energy, Mass, and Chemical Species Transport: User's Guide," Sandia National Laboratories Report, SAND97-2404, Albuquerque, New Mexico (1998).
28. D. R. Adkins, D. K. Gartling, J. B. Kelly, and P. M. Kahle, "TORO II Simulations of Induction Heating in Ferromagnetic Materials," in *Proceedings of the 1st International Induction Heat Treating Symposium*, Indianapolis, Indiana (1997).
29. D. K. Gartling and P. A. Sackinger, "Finite Element Simulation of Vacuum Arc Remelting," *International Journal for Numerical Methods in Fluids*, **24**, 1271-1289 (1997).
30. F. J. Zanner and L. A. Bertram, "Vacuum Arc Remelting - An Overview," in *Proceedings of the 8th International Conference on Vacuum Metallurgy*, **1**, 512-552 (1985).

Parallel Processing

8.1 Introduction

Parallel processing is still a relatively new facet of computing, with a significant history in computational mechanics of less than two decades. However, the technology is a certainty and will continue to have a major impact on computational mechanics for the foreseeable future. Many of the currently popular computing systems are based on the sequential processing, von Neumann architecture, in which instructions are executed in sequence, one at a time, on a single data element. For computational algorithms that have a sequential character, this type of processing is optimal. However, most computational problems are not strictly sequential, but have processes that are independent of each other (e.g., element matrices), and could therefore be executed simultaneously. For these parallel tasks or algorithms, sequential processing is obviously not optimal. In the past, increases in computer performance have been primarily achieved through increases in speed of the central processing unit (CPU). However, as the rate of increase in CPU speed slows down, and costs for complex CPUs increase, computer developers have turned to alternative architectures to achieve high computing performance. A logical choice is to use multiple processors and exploit the parallelism inherent in most computational algorithms. This hardware development has led to the rapidly growing field of parallel computing.

This chapter provides a brief introduction to the topic of parallel processing and its relation to finite element algorithms. In previous chapters we have mostly avoided discussions of finite element implementation since there are many ways to achieve the required algorithmic result. Some texts [1–4] have illustrated the finite element implementation process by providing detailed descriptions and listings of source codes for a variety of applications. Appendix A provides a minimal introduction of this type for some two-dimensional problems of interest in this text. The implementation of finite element procedures in a parallel computing environment involves a substantial increase in complexity with many algorithms requiring a heavy dependence on areas in computer science. A good introduction to parallel computing is found in [5].

In the next section an introductory description of parallel systems is provided. We will then outline some of the major algorithmic areas that must be addressed if a finite element procedure is going to be adapted to or designed for a parallel computer. It is not our intention to be extremely detailed in this chapter but it is important to call attention to methods and code structures that have been tested and put into use in engineering simulations. Likewise, it is important to point out where finite element methods and parallel processing are the most conflicted and a substantial code development effort may be required.

8.2 Parallel Systems

8.2.1 Classification

The standard terminology used for classifying parallel computers is the scheme proposed by Flynn [6]. Even though this scheme does not include many new developments in classification [7], it still forms a good basis. Following are the four classes in the scheme proposed by Flynn [6]:

1. Single Instruction Single Data (SISD)
2. Single Instruction Multiple Data (SIMD)
3. Multiple Instruction Single Data (MISD)
4. Multiple Instruction Multiple Data (MIMD).

An instruction stream is the sequence of instructions executed by a processing element (PE) and a data stream is the sequence of data on which instructions are performed.

The sequential computers (Micro and Mini computers) fall under the class of the SISD system. They have a single processing element executing instructions in sequence on a single stream of data. The SIMD systems have many processing elements and execute the same set of instructions in each of the processing elements but on different data streams. Vector computers fall under this class. The MISD computers execute a different set of instructions in each processing element but on the same data stream. These systems are useful in applications like signal processing and are often termed pipeline processors. The MIMD machines are the most general type among the parallel processing systems. They have many processing elements executing their own sets of instructions on different data streams.

Parallel processing systems can be further classified based on the characteristics listed below. Most of the parallel machines of interest in computational mechanics come under the MIMD or SIMD class of computers. The classification discussed here is relevant primarily to these systems.

8.2.1.1 Granularity of the processing elements

Granularity of the system can be either fine or coarse, where a coarse grain system has fewer processors than the fine grain system. A coarse grain system typically operates with a few (say, 100) powerful processing elements, while a fine grain system would have many (say, 1,000) ordinary processing elements. The boundary between fine and coarse grain computers is not rigidly defined. Systems with more than a 1,000 processors are usually termed *massively parallel* computers.

8.2.1.2 Topology of interconnections

The topology of interconnections is the manner in which the individual processing elements are connected. In terms of hardware, the connections may be arranged as a ring, tree, pipeline, or a hypercube. With the advent of new systems and with a virtual communication facility, the programmer can assume that each processor is connected to every other processor. The system will take care of the data communication through the intermediate processors based on the actual physical topology. Efficiency of such a system will be high if the actual and virtual layouts are identical. Figure 8.1.1 shows the commonly used interconnect topologies. These types of processor topologies are common to multiprocessors, i.e., computers with all of the processors located in a single system.

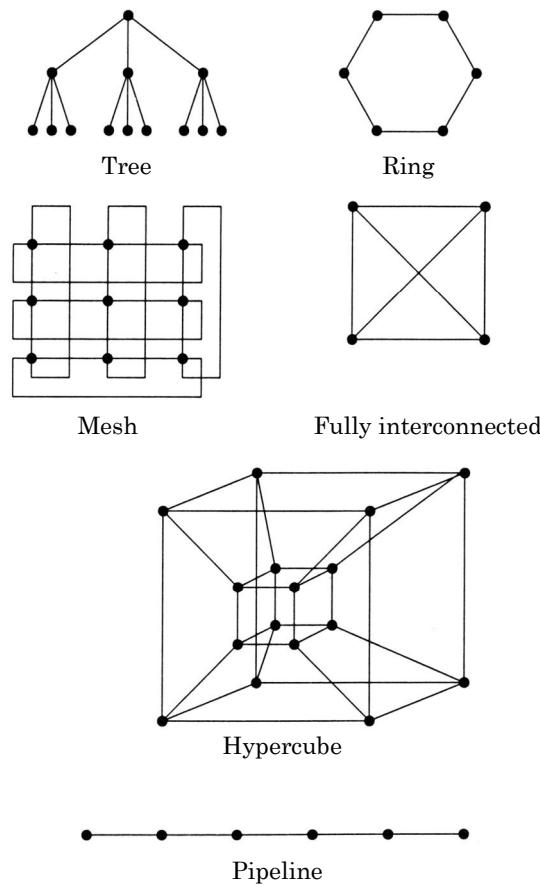


Figure 8.1.1: Topology of interconnections.

Networks of single processor computers can be arranged to provide a parallel computing system; a heterogeneous network of multiprocessor machines can be used to form a virtual parallel machine. These types of networked architectures are usually referred to as *distributed computing*. The major difference between a parallel or massively parallel computer and a distributed computing system is the variability of the processing unit. On a multiprocessor every processor is essentially the same, whereas in a distributed (virtual) machine each processor on the network could be very different in terms of speed and capability. Likewise, data formats may vary significantly between processors on a network. These differences lead to software portability issues and concerns with the basic parallel paradigm, language selection, and communication protocols.

8.2.1.3 Distribution of control across the processing elements

The control of the system may be with a single processor, which directs the system, as is often the case in SIMD machines. Also, in MIMD machines, even though a master (or main) processor is identified, each processor operates on its own, independently. However, interprocessor communication demands synchronous operation.

8.2.1.4 Memory access

Each processing element can have its own local memory and/or share the global memory of the system. Both types of architectures have advantages and disadvantages. In a local memory system, synchronous communication becomes vital and deadlocks/delays in communication may cause drastic deterioration of the system efficiency. In a shared memory system, processing elements compete with each other in addressing the memory locations. Bottlenecks in communication remain one of the major problems in a shared memory system. Again, the most common architectures used in computational mechanics are the shared memory MIMD machines and the distributed memory MIMD computers with the latter being the most popular. A detailed discussion of these characteristics is available in [5-8].

8.2.2 Languages and Communication Utilities

There are many types of parallelism in computing. For some applications, special computer languages or language extensions have been developed to simplify program development. A *data parallel* model seeks to exploit parallel capabilities in the performance of even simple arithmetic tasks such as a vector-matrix product. In this case, corresponding components of the matrix and vector are distributed to different processors for multiplication and returned for summation. Microtasking or multitasking of this type originally required compiler directives to indicate sections of the code that could be spread across multiple processors. Currently, languages such as High Performance (HP) Fortran [8] and Fortran 90 [10] implicitly support data parallel constructs. For systems based on transputers [11], the language of choice is Occam, which was developed specifically for this type of processor [11,12].

The usual approach to parallel computing of mechanics problems is a domain decomposition or partitioning paradigm with a single program, multiple data (SPMD) software implementation. In this situation, the problem data are divided or partitioned equally among the available processors. A copy of the finite element mechanics code runs on each processor and computes its locally assigned part of the problem. Crucial to the success of this paradigm is the requirement to efficiently communicate (transfer) data between processors such that the global problem can be solved from its distributed pieces. The two traditional methods for processor to processor communication on standard MIMD machines are the use of shared memory or the use of an explicit message passing utility. Message passing has become something of a standard because of its generality, especially considering the difficulties associated with distributed computing.

In the standard SPMD implementation, no special languages or extensions are required. The use of ANSI standard Fortran 77 and C is widespread. Finite element programs require a mixture of data processing procedures, some of which are best suited to C or Fortran 90 constructs. Other parts of the code are optimally constructed using the unit stride facilities of Fortran. Most current codes are a mixture of the two languages. The use of Object Oriented designs and the C++ language [13] is gaining in popularity.

Utilization of a distributed memory parallel system requires the ability to perform two basic functions: create tasks and set communication channels between processors. Portable, system independent libraries with standardized interfaces have

been developed to accomplish these functions. The two most popular packages of this type are the Parallel Virtual Machine (PVM) library [14] and the Message Passing Interface (MPI) software [15]. Other systems, such as Linda [16] and P4 [17] exist but do not have the overall popularity of PVM and MPI. PVM is the older of the two communication packages and is best suited for systems with a few processors and an architecture that operates with a master processor and a number of slave processors. Distributed computing architectures with heterogeneous network processors are well aligned with PVM. The MPI software has found the widest utility and is heavily used on multiprocessor machines and networks, as well as massively parallel machines. MPI is especially convenient for the SPMD models found in most mechanics applications. Some language extensions that support task and channel functionality are also available, though not widely used. The CC++ [18] language is an extension of C++ and supports parallelism from within the language; Fortran M [19] is an extension of Fortran with similar functionality.

8.2.3 Performance

The performance of a parallel code is generally estimated by its parallel efficiency, which indicates how much is gained by solving the problem in parallel. The best way to estimate this is to compare it with a sequential version of the same code. This can be done in two different ways [20].

8.2.3.1 Algorithmic efficiency

This is defined as the ratio of CPU time taken for running the parallel code on N_p processing elements to the time taken when the code is run on a single processing element. The sequential code used here is essentially a parallel code run on a single processor. Hence the sequential code will also have the delays in communication reflected in its timing. This will indicate the parallel efficiency of the algorithm rather than the problem. The formula for algorithmic efficiency is

$$\eta_{alg} = \frac{t_{par}}{t_{seq1}N_p} 100 \quad (8.2.1)$$

where t_{par} is the time taken to execute the program on a parallel machine with N_p processors and t_{seq1} is the time taken to run the same code on a single processor.

8.2.3.2 Actual/beneficial efficiency

Another measure, the actual or beneficial efficiency is defined as the ratio of CPU time taken for running the parallel code on N_p processing elements to the time taken by the fastest sequential code run on one processing element (t_{seq2}). This efficiency will reflect the parallel efficiency of the problem, or in other words it will reflect both the level of parallelism in the problem under consideration and the efficiency of the parallel algorithm used. It indicates the actual benefit derived by resorting to parallel processing. It is given by the following equation:

$$\eta_{act} = \frac{t_{par}}{t_{seq2}N_p} 100 \quad (8.2.2)$$

The actual efficiency will always be less than or equal to the algorithmic efficiency. If these efficiencies are less than 50%, then it is not usually worth considering the

case. Sometimes a speed-up ratio is also used to rate a program. This is just the ratio of time taken by the parallel program to the time taken by the sequential program. The speed-up ratio should be greater than one. The speed-up factor can be calculated by multiplying the efficiency by $N_p/100$.

8.2.3.3 Scalability

Another important issue for performance is scalability. It is expected that systems with increasing numbers of processors will continue to be developed. Software that maintains performance with an increasing number of processors is therefore important for effective use of a parallel system. Early definitions of scalability sought to decrease computational time by increasing the number of processors on a fixed work load. It was believed that the serial portions of the algorithm and communication costs would ultimately limit performance as the number of processors increased. This is a statement of Amdahl's law which has as its focus a decreasing work load per processor. A current view of scalability asks for a constant work load per processor, in which case the overall work load would increase with the number of processors. In this situation, any degradation in performance can be assigned to communication cost.

8.3 FEM and Parallel Processing

8.3.1 Preliminary Comments

Previous chapters have outlined the steps/procedures for finite element algorithms for a number of specific problem types. The implementation of all of these finite element methods follows a series of generic steps. In the first part of this section, these generic finite element procedures will be discussed in the context of a parallel computing implementation. A later section will describe some of the issues that arise due to specific computational and modeling techniques such as multipoint constraints and code coupling.

As noted previously, the most common parallel architecture of interest in the computational mechanics community is the MIMD architecture with a shared memory, or more preferably, with a distributed memory. The MIMD architecture argues for a decomposition or partitioning strategy when considering approaches to finite element solutions in a parallel environment. Note that here we are considering only the solution to a single finite element model; other, more coarse grained levels of parallelism are possible when considering multiphysics applications. If the computer has N processors, then the finite element mesh is decomposed into N roughly equal groups of (contiguous) nodes or elements with each group assigned to a processor. Each processor computes the finite element algorithm for its group of nodes or elements with interprocessor communication required for completion of the algorithm at processor boundaries where nodes or elements are shared. Input and output from the process may be handled through a special, dedicated front end processor, by one of the N processors that is assigned the duty, or through all the processors and a parallel I/O hardware system. Often the finite element software is structured such that a copy of the executable code is running on each processor; at appropriate points in the execution, the processors are synchronized and data are exchanged between processors. This is the SPMD model mentioned previously.

As sketched here, the domain decomposition or partitioning approach to parallel finite element solutions is the most common paradigm currently in use. The discussion to follow will be oriented toward this type of procedure. Note that the term domain decomposition is also used to describe parallel solution methods for large systems of equations. The two areas are obviously related but not necessarily coincident. We use the term here only to describe an approach to parallel implementation of finite element methods.

8.3.2 Generic FEM Steps

A finite element algorithm or program is naturally divided into the following major steps or units:

- External preprocessing
- Internal preprocessing
- Solution processing
 - Element matrix building
 - Matrix solving
 - Solution control
- Internal postprocessing
- External postprocessing

The first and last processing steps usually occur outside of the basic finite element program in specialized mesh generators and graphics packages. However, these external processes must be included when discussing a parallel application because they play a significant role in the problem setup. Each of these processing steps is considered individually in the next several sections.

8.3.3 External Preprocessing

Preprocessing for a finite element model usually refers to the generation of the element mesh from some type of geometric description of the problem region. In many cases the problem geometry resides in a computer-aided design (CAD) file. Mesh generation sets the location of the nodal points (global coordinates) and defines the elements in terms of the nodes through construction of the nodal connectivity. The connectivity is an ordered list relating the nodes in an element to the global node numbering in the assembled model. Elements in the mesh are normally assigned material i.d.s at this point and element surfaces and nodes are flagged for boundary conditions. The output from this step is a file, in some standard format, that provides all the required data for each element in the mesh.

For parallel applications, preprocessing involves several additional steps. The distribution of the finite element computations to a number of processors is accomplished through domain decomposition or mesh partitioning algorithm. Within this procedure, the overall problem is divided into N sets of elements for execution on N processors of the parallel platform. Efficient execution of the parallel problem demands that the computational work on each processor be as equal as possible and that the communication between processors be minimized. This part of the process is referred to as “load-balancing.” Various software packages, such as Chaco [21] and Parmetis [22], have been developed to perform this type of mesh decomposition and processor load-balancing. The mesh decomposition algorithm

is often based on viewing the mesh as a graph and utilizing sophisticated graph partitioning methods. Recursive bisection and octree methods have also been used as partitioning procedures. In each case, the partitioning method must consider the general case where the number of unknowns per node and per element may vary between elements. This situation is usually handled by weighting the vertices of the graph to reflect the work associated with a node or element. Also, partitioning may be nodal or element based. In a node-based decomposition, nodes are assigned to processors and elements may be split between processors. An element-based decomposition assigns elements to a processor with nodes then being shared between processors. The type of needed decomposition and load balance is thus dependent on the architecture of the finite element code.

The domain decomposition and load-balancing software typically functions by reading the mesh generation file, decomposing the mesh for the stated number of processors, and producing its own output file or files. The form of the partitioned output again depends on the structure of the finite element code. If the finite element code is designed to have a master process and a number of slave processes, the domain decomposition data can be stored in a single scalar file. This file would contain the load balance data needed by the master process to distribute the node, element, and communication data to the individual processors. Finite element codes designed to run with a copy of the code on each processor (the SPMD model) require the decomposition data in a set of parallel files. Each of the parallel files would contain some global information plus the nodal, element, and communication data specific to a processor.

Figure 8.3.1 shows the finite element mesh of a casting with a partial gating system that was partitioned for use in a heat conduction/radiation simulation using eight processors. The original geometry was meshed using a standard mesh generator with the output being processed by a graph partitioning method within the Chaco code [21]. Additional utilities were used to reformat the output file into a series of eight parallel files that were subsequently read by a heat transfer code. Note that the original mesh contained 3,297 elements (6,661 nodes) and the element-based partitioning yielded 412 elements on seven processors and 413 elements on one processor. The number of nodes on each processor varied from 790 to 944.

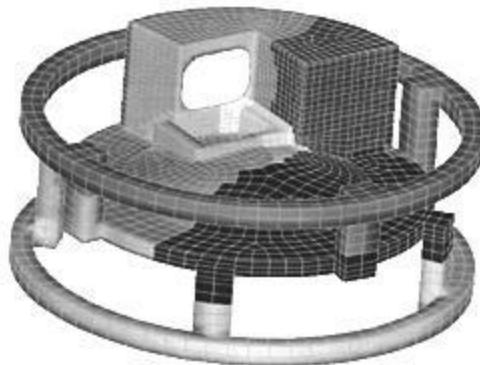


Figure 8.3.1: A finite element mesh showing a partitioning for use on a parallel computer.

In the course of this discussion, it has been implied that the mesh generation occurs on a serial computer and the decomposition and load-balancing follow as a utility process. In many situations this type of procedure is adequate. For extremely large and/or complex finite element models, it may not be possible to generate the mesh on a serial computer. Parallel mesh generation is an area of current investigation and will be required as simulations become more sophisticated. Also, static domain decompositions are adequate for static element topologies; adaptive meshing or remeshing will require dynamic load-balancing capabilities [23].

8.3.4 Internal Preprocessing

The first steps within the finite element code usually entail the reading of input data, reading of mesh data, the allocation of memory for the problem and the setting of execution and control pointers and flags. These same steps are required in a parallel code though the source of the data may be different. As noted in the previous section, mesh data may be available from a parallel input file for some types of codes or may be passed from the master processor and a serial file. Input data that specify material properties, boundary condition values, solution control, and postprocessing requirements usually come from a single, serial file and must be broadcast to the appropriate processors. Processor communication utilities, such as MPI, must be initialized. Assuming a distributed memory machine, local memory on a processor is allocated based on the number of elements or nodes assigned to the processor. Print or text output from the code is generally written from a single processor, even if the code design does not specifically include a master or front-end processor. At the conclusion of initialization and data checking, the processors are synchronized and the first step of the solution is ready for processing.

8.3.5 Solution Processing

A solution step for a finite element method consists of two major parts – element matrix building and solution of the global matrix system. For time-dependent simulations or nonlinear steady problems this step will be repeated for each time step and/or each iteration. The solution of the equations is usually the most time-consuming part of the analysis and can benefit substantially from a parallel implementation. For time-dependent solutions obtained via explicit methods, no matrix solution is required though a parallel implementation is still very effective.

8.3.5.1 Element matrices

The finite element method lends itself naturally to parallel computing especially in the equation building part of the algorithm. The usual structure for a serial finite element code contains a loop over the elements during which element matrices are computed (via numerical quadrature) and added to the global matrix. In most cases, the evaluation of the matrices for an individual element is independent of other elements. Hence, generation of element matrices can be carried out simultaneously, i.e., in parallel. The parallel finite element code, therefore, has a loop over all the elements on the processor, where element matrices are computed and added into the part of the global matrix represented on the processor. This architecture presumes that an element-based decomposition has been used in the mesh partitioning so that all the data for an element reside on the processor. The part of the global matrix that is assembled on each processor represents a combination of both fully summed

equations and partially summed equations for nodes shared between processors. At the completion of the element loop, this submatrix is passed to the parallel matrix solver for further processing and solution. Note that the element-based decomposition is the natural choice for element construction since an element is the primary data entity.

In the case where the partitioning is node based, equation construction is slightly more complex. The looping is over the nodes on the processor. An inverse connectivity is used to determine which elements (and nodes) are connected to the node. The integration of the shape functions for the nodal equations will require coordinate data from other processors and therefore communication between processors. At the completion of the loop, a submatrix will be available to the parallel solution routine. This type of decomposition is well suited for most parallel, iterative matrix solution methods but is not well aligned with finite element data processing.

8.3.5.2 Matrix solvers

The matrix solver used for a parallel code implementation can be either iterative or direct; the serial versions of these two types of solvers are discussed in Appendix B. Direct solvers obtain the solution to the system of simultaneous equations in a fixed number of steps. On the other hand, iterative solvers do not converge to the solution in a fixed number of iterations. The number of iterations required depends on the mathematical structure of the system of equations being solved. Nonlinearity of the problem and the diagonal dominance of the system of equations affect the convergence rate. In some cases, it may not even be possible to obtain a solution using an iterative solver, especially without a preconditioning of the coefficient matrix. The major advantage of most iterative solvers is that they are amenable to the element-by-element (EBE) or block EBE solution approach. The system of equations is solved without direct assembly of the element matrices. These algorithms have very high parallel efficiency when compared with the direct solvers.

Most direct solvers use a Gauss elimination scheme as their base algorithm. This algorithm consists of two steps, the forward elimination and the backward substitution. The forward elimination can be programmed in parallel with a good parallel efficiency, but with a significant increase in solver complexity. The parallel forward elimination is much like a substructure process where the interior nodes (fully summed equations) on a processor are eliminated, leaving the border nodes (partially summed equations). For efficiency, the summation and subsequent elimination of groups of border nodes should be redistributed among all the processors. The continuation of this process to its conclusion involves substantial data tracking and communication. Finally, the backward substitution is sequential in nature, and this will decrease the overall parallel efficiency of the solver. Also, the forward elimination will be less efficient for banded systems.

The Gauss–Jordan solver is a less efficient solver in a sequential mode, and it is rarely used in sequential programming. If N is the number of unknowns, Gauss elimination performs $O(N^3/3)$ multiplications and the same number of additions during the elimination operation (multiplication and division and addition and subtraction are taken as identical operations). The back substitution scheme involves $O(N^2/2)$ multiplications and additions. On the other hand, the

Gauss–Jordan solver performs $O(N^3/2)$ multiplications and additions during the elimination step, and there is no back substitution. Since there are more operations in the Gauss–Jordan method than the Gauss elimination scheme, it is not preferred; but the issues take different priorities in parallel processing. The Gauss–Jordan algorithm is devoid of the back substitution step, which is highly sequential. This is a major advantage. Also, more computation is performed for a unit of communication in the Gauss–Jordan method than the Gauss elimination scheme. This will result in an improved parallel efficiency [23]. For a complete system, Gauss–Jordan will break even with Gauss elimination, even in terms of CPU time, with only a few processors. However, for a banded system, a greater number of processors will be required to achieve parity.

Among the many iterative solvers, like the Jacobi iteration method and the Gauss–Seidel method, the Conjugate Gradient Method (CGM) is very efficient and effective for many finite element equation systems. In theory, it will yield convergent solution in less than N iterations, where N is the number of unknowns. In practical computations, round-off errors will affect convergence and it often requires more than N iterations to achieve a solution. Hence, restarting procedures have to be used. Even with this penalty, it is faster than the Gauss–Seidel iteration method by an order of magnitude.

The conjugate gradient method is often used for minimizing a function. It uses the gradient of the function to minimize it, without evaluating the function itself. This method was originally developed for solving systems of simultaneous equations. The original method can solve only symmetric, positive-definite systems. Using an appropriate preconditioning matrix, other systems can also be solved. Details of the conjugate gradient method with preconditioning can be found in [24] as well as in Appendix B.

In developing a parallel implementation for an iterative matrix solution method, two key features are exploited. Algorithms like the conjugate gradient method involve matrix–vector products and vector dot products. These steps can be easily split into parallel operations by expressing the product of a matrix with a vector as the sum of the products of their components. The key idea is

$$[A]\{x\} = \left[\sum_{i=1}^{N_{elem}} [A_i] \right] \{x\} \quad (8.3.1)$$

$$[A]\{x\} = \sum_{i=1}^{N_p} \left[\sum_{j=M_{i1}}^{M_{i2}} [A_j] \right] \{x\} \quad (8.3.2)$$

where N_p is the number of processors and N_{elem} is the number of elements. The values of M_{i1} and M_{i2} can be calculated in different ways. Only the matrix $[A]$ is expressed as a sum and not the load vector. Therefore, the load vector generated by the processors has to be added to obtain the final vector, which involves some communication. If the load vector is also expressed as a sum, the process will be even more complicated. Also, it will be computationally intensive with a heavy communication load. This can be readily seen from the following equations:

$$A \cdot x = (A_1 + A_2)(x_1 + x_2) = A_1x_1 + A_1x_2 + A_2x_1 + A_2x_2 \quad (8.3.3)$$

$$A \cdot x = \left[\sum_{i=1}^N A_i \right] \left[\sum_{j=1}^N x_j \right] = \sum_{i=1}^N \sum_{j=1}^N A_i x_j \quad (8.3.4)$$

If there are N values of A and x , each value of A is associated with all the N components of x . This places a requirement on the processors to know all the values, and there is a larger number multiplications as well. It is therefore not advisable to express the vectors involved in the matrix-vector product as a sum of its element contributions.

The other product frequently encountered is a vector dot product. Consider two vectors $\{x\}$ and $\{y\}$ of dimension N . The dot product can be written as

$$\{x\} \cdot \{y\} = \sum_{i=1}^N x_i y_i \quad (8.3.5)$$

$$\{x\} \cdot \{y\} = \sum_{i=1}^{N_p} \left[\sum_{j=M_{i3}}^{M_{i4}} x_j y_j \right] \quad (8.3.6)$$

The values of M_{i3} and M_{i4} are calculated from the element connectivity of the elements present in the i th processor. Care should be taken to make sure that a particular node is identified with only one processor in calculating the dot product, even though it may be associated with many processors. This is due to the fact that the vectors are assembled and they are global. Depending upon the particular algorithm under consideration, the methodology can be restructured.

Although the matrix solution process occupies a large majority of the execution time in a finite element code, it is no longer necessary to understand the implementation details of the various matrix solvers. A number of software projects have produced solver libraries with convenient, standardized interfaces that can be used easily with many finite element formulations. Many of these libraries have both direct and iterative matrix solvers and usually have available a number of preconditioning options. Typical of these solver packages are LAPACK [25] for shared memory machines and ScaLAPACK [26], PETSC [27] and Aztec [28] for distributed memory architectures. The use of well-documented solver libraries is an effective method for quickly getting a finite element algorithm running in parallel. As noted above, much of a finite element computation is naturally parallel and easy to code. The matrix solution is the difficult part of the parallel implementation and involves most of the “computer science” aspects of a finite element code. Though solver libraries may be conveniently used for expediting program development, it is important to understand the relationship between various matrix solver methods and characteristics of the finite element equations (matrices).

8.3.5.3 Solution control

At the conclusion of a matrix solution, various computations must be completed that influence the overall progression of the algorithm. In particular, norms on the solution field must be computed and checked to determine if convergence has occurred. Also, for time-dependent simulations tests for reaching steady state are computed and a new time step may be required. Each of these computations requires

some communication between processors though these tasks are minor compared to the data exchange that occurs within the matrix solver.

8.3.6 Internal Postprocessing

At the completion of the finite element solution, some postprocessing of the data may be required. The computation of auxiliary quantities, such as flux data or globally integrate values, may occur at the end of each time step or nonlinear iteration or at the completion of the steady state solution. In a serial algorithm these data are again processed within a loop on the elements. The parallel implementation of this type of computation is similar though some communication may be required.

If fluxes are computed at the integration points of an element, then the computation is completely parallel if the decomposition was element based. When continuous flux data are required, the usual process is to extrapolate the integration point fluxes to the nodes and average the nodal data between elements attached to the node. This type of averaging requires some interprocessor communication. Using communication utilities such as MPI makes the computation straightforward. Quantities that are integrated over surfaces or volumes likewise require data exchange between processors but nothing outside the scope of MPI-like capabilities.

Output of postprocessing data may be accommodated by either of the two methods used to input data. For many systems, data would be collected on the master processor or a designated processor and written to a printed output file and/or a postprocessing file. Codes using the SPMD model would more likely collect data on a single processor for printing while using parallel output files for postprocessing data. The parallel files are usually concatenated into a single file prior to use with a graphics package.

8.3.7 External Postprocessing

External postprocessing refers to the use of specialized graphics programs to investigate the solution fields generated by the finite element code. These software packages have a variety of methods for graphically displaying data, manipulating data, and probing the computed fields. For applications generated by a parallel algorithm the concerns related to postprocessing stem mainly from the size of the computed data sets. It is a significant challenge to graphically render nodal or element data from time-dependent finite element meshes containing several million elements. Not only is rendering time an issue but temporary and permanent storage of the data sets is a formidable problem. Work on parallel graphics software is in progress and will be mandatory as finite element simulations continue to increase in size and complexity.

8.3.8 Other Parallel Issues

It is clear from the previous section that most of the parallel issues in a standard finite element algorithm are contained within the matrix or linear algebra solution procedure. However, there are some finite element modeling capabilities that complicate the normal parallel implementation based on mesh partitioning. As described in Chapter 3, contact and multipoint constraint boundary conditions require data external to the current element. Algorithms that require the staggered or cyclic solution of multiple equation sets also add some complexity to a parallel implementation. These topics are considered briefly in the following sections.

8.3.8.1 Nonlocal data

The efficient use of a domain decomposition method requires that data used by a processor be local to the processor. This is the usual situation when an element-based decomposition is used and the looping is over the elements on a processor. Nonlocal or off-processor data can occur in the implementation of certain types of boundary conditions, such as node-to-surface contact and multipoint constraints (see Section 3.11.3). Implementation of these types of conditions requires two steps: the location of the slave node on the master surface and the construction of an equation relating the slave node to the nodes on the master surface. When the slave node is on a different processor from the master surface, additional interprocessor communication is required during both steps of the procedure. Note that preprocessing data is not available for establishing these communication paths since the boundary condition may be dynamic and change location during the course of the solution.

The search for the location of one or more (slave) nodes within a list of master elements is a procedure that occurs in several computational situations. The transfer (interpolation) of data from one mesh to another requires this type of procedure as does the implementation of the boundary conditions mentioned above. In a parallel setting the local search process is most effectively done after the slave nodes and master elements that are geometrically close are redistributed to common processors. This amounts to a slave node-based repartitioning of the data. An effective method for load balancing this problem is a recursive bisection algorithm. After completion of the local, on-processor search procedure, the location data for each slave must be returned to its original processor.

The actual implementation of the multipoint constraint requires a second step in which a constraint matrix is constructed between the slave node and the nodes on the master surface. The contact boundary condition requires this same type of construction if the implementation is fully implicit; an explicit formulation for contact can be constructed without this second step. After the constraint equations are constructed on the processor containing the slave node, the constraint coefficients must be distributed to the processors containing the master surface nodes and added to their portion of the global matrix. These interprocessor communication paths are flagged during the search process since they may change with time and are not included as part of the original mesh partition.

Another type of nonlocal data occurs for modeling features such as a bulk node (see Section 3.11.4). Here, the equation for a bulk node is not part of the original mesh partition and in fact is owned by all the processors. Because the bulk node may be connected to a large number of elements (and nodes) located on different processors, updating of the bulk node equation occurs on every processor. This local update is followed by a global accumulation of data and integration of the bulk node equation on each processor. The bulk node parameters are then available to each processor for use in boundary conditions on the processor.

8.3.8.2 Multiphysics simulations

The use of cyclic or staggered solution methods for multiphysics or multiple systems of equations may introduce additional complexities into the parallel implementation. The two issues of main concern are load balancing and data transfer between

equation sets. When the equation sets are solved on the same element topology (and same domain decomposition), these problems do not occur.

As a specific example of the load-balancing problem, consider the fairly common situation of solving the coupled heat conduction, enclosure radiation problem or the heat conduction, chemical kinetics problem outlined in Chapter 3. These types of problems would normally be solved with a single software package, so that code coupling is not a complicating factor. A recommended method in both cases is the staggered or alternating solution of the two equation sets as a function of time. In a parallel implementation, the finite element conduction problem is partitioned across the N processors and solved by the standard method. However, when the second equation set is solved it is probable that a number of processors will be idle because its computational domain does not necessarily coincide with the conduction domain. In the radiation case, the view factors and radiosity problem are computed for a subset of element surfaces. In the chemical kinetics problem it is usual for only one material in the problem to be reactive, while the other materials require no chemistry solution. Note that it is quite possible to have both radiation and chemistry in the same problem in which case none of the equation domains are coincident.

The solution to this dilemma is of course to have individual decompositions for each equation set in the problem. The individual partitions may be generated externally if the equation sets are complex, e.g., two or more partial differential equation models. For less complex secondary equation systems, the partitioning may be done internally with a simple algorithm. In the case of radiation view factors or chemistry on an element, the number of equations or elements divided by the number of processors provides an adequate partition. This simple method works in these cases because there is no interaction between the equations within the set. Regardless of the method used for partitioning the second equation set, some additional interprocessor communication is required. In essence, the communication relationship between the partitions must be established to allow the exchange of dependent variables between the sets of equations.

The previous load-balancing discussion centered on the situation encountered when a single finite element code was required to handle multiple equation sets. It was assumed that the variables were defined at the same spatial locations (nodes and elements) in each equation, though the spatial extent of the domains did not have to coincide. The coupling of independent finite element codes, as described in Chapter 7, can present another difficulty for parallel implementation. When the coupled codes use independent finite element meshes and different domain decompositions, the transfer of variable data from one mesh to the other involves added complexity. The basic problem is similar to the multipoint constraint problem described above. To transfer nodal data from mesh A to mesh B, the locations of the mesh B nodes within mesh A must be found. Once located, the nodal data for mesh B can be found from shape function interpolation on mesh A. The search procedure is most effectively done in parallel by redistributing the data across the processors such that groups of mesh B nodes and mesh A elements are geometrically close. A recursive bisection procedure is fairly optimal for this process. Once the interpolated nodal data are computed, the variables are transferred back to the original processor so that the next solution step in the algorithm can be completed. In general, the search

and interpolation process must be completed in each direction if the two solution fields are mutually dependent. When the geometries are static the search need only be done once, though the interpolation would have to be carried out at each data exchange step. The complexity of this process obviously increases substantially for each coupled code added to the simulation.

8.4 Summary

The future of computational mechanics will be influenced to a great extent by the developments in parallel processing. Computational fluid dynamics, in particular, can and will benefit significantly from the developments in parallel computing. Currently, most of the parallel algorithms are the parallel versions of existing sequential algorithms. It may be more useful to conceive and develop algorithms which are fundamentally parallel. Other than CFD, problems involving artificial vision, signal processing, neural networks, optimization, and general finite element analysis will benefit with the advances in parallel computing. The scope is much broader than what is mentioned here.

References for Additional Reading

1. C. Taylor and T. G. Hughes, *Finite Element Programming of the Navier-Stokes Equations*, Pineridge Press, Swansea, U.K. (1981).
2. T. J. R. Hughes, *The Finite Element Method, Linear Static and Dynamic Finite Element Analysis*, Prentice Hall, Englewood Cliffs, New Jersey (1987).
3. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).
4. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis*, Oxford University Press, Oxford (2004).
5. I. Foster, *Designing and Building Parallel Programs*, Addison-Wesley, Reading, Massachusetts (1995).
6. M. J. Flynn, “Very High Speed Computing Systems,” *Proceedings of IEEE*, **54**, 1901–1909 (1966).
7. B. Davidson, “A Parallel Processing Tutorial,” *IEEE Antennas and Propagation Society Magazine*, **32**, 6–19 (1990).
8. M. R. Hord, *Parallel Supercomputing in SIMD Architecture*, CRC Press, Boca Raton, Florida (1990).
9. C. Koebel, D. Loveman, R. Schreiber, G. Steele, and M. Zosel, *The High Performance Fortran Handbook*, MIT Press, Cambridge, Massachusetts (1994).
10. J. Adams, W. Brainerd, J. Martin, B. Smith, and J. Wagener, *The Fortran 90 Handbook*, McGraw-Hill, New York (1992).
11. A. Carling, *Parallel Processing: the Transputer and Occam*, Sigma Press, London, U.K. (1992).
12. J. Galletly, *Occam 2*, Krieger Publishing, Melbourne, Florida (1990).
13. J. Barton and L. Nackman, *Scientific and Engineering C++*, Addison-Wesley, Reading, Massachusetts (1994).
14. A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, and V. Sunderam, *PVM: Parallel Virtual Machine, A User’s Guide and Tutorial for Network Parallel Computing*, MIT Press, Cambridge, Massachusetts (1996).

15. W. Gropp, E. Lusk, and A. Skjellum, *Using MPI, Portable Parallel Programming with the Message Passing Interface*, MIT Press, Cambridge, Massachusetts (1995).
16. N. Carriero and D. Gelernter, "Linda in Context," *Communications of the ACM*, **32**, 444–458 (1989).
17. R. Butler and E. Lusk, "Monitors, message, and clusters: The *p4* parallel programming system," *Parallel Computing*, **20**, 547–564 (1994).
18. P. Sivilotti and P. Carlin, "A Tutorial for CC++," California Institute of Technology Technical Report, CS-TR-94, Pasadena, California (1994).
19. I. Foster, R. Olson, and S. Tuecke, "Programming in Fortran M," Argonne National Laboratory, Mathematics and Computer Sciences Division Technical Report, ANL-93/26, Argonne, Illinois (1993).
20. T. L. Freeman, *Parallel Numerical Algorithms*, Prentice Hall, Englewood Cliffs, New Jersey (1992).
21. B. Hendrickson and R. Leland, "The Chaco User's Guide - Version 2.0," Sandia National Laboratories Report, SAND94-2692, Albuquerque, New Mexico (1994).
22. G. Karypis and V. Kumar, "Parmetis: Parallel Graph Partitioning and Sparse Matrix Ordering Library," University of Minnesota, Department of Computer Science Technical Report, 97-060, Minneapolis, Minnesota (1997).
23. J. J. Modi, *Parallel Algorithms and Matrix Computations*, Oxford University Press, Oxford, U.K. (1988).
24. O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge, U.K. (1996).
25. E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbau, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, "LAPACK Users' Guide, Second Edition," Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania (1994).
26. L. S. Blackford, J. Choi, A. Cleary, E. D'azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R. C. Whaley, "ScaLAPACK Users' Guide," Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania (1997).
27. S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith, "PETSc 2.0 Users Manual," Argonne National Laboratory Report, ANL-95/11 - Revision 2.0.28, Argonne, Illinois (2000).
28. S. Hutchinson, L. Prevost, J. Shadid, C. Tong, and R. Tuminaro, "Official Aztec User's Guide, Version 2.1," Sandia National Laboratories Report, SAND99-8801, Albuquerque, New Mexico (1999).

Appendix A

Computer Program *FEM2DHT*

A.1 Introduction

The computer program *FEM2DHT* is a finite element analysis program for the solution of two-dimensional heat transfer and viscous, incompressible fluid flow problems. The program is a modification of the computer program FEM2D from the book by Reddy [1]. The program is discussed here to aid the readers with the basic computer implementation aspects of the finite element method. The program is educational in nature and it does not contain powerful pre- or post-processors found in a commercial finite element software.

A.2 Heat Transfer and Related Problems

The heat transfer part of the program is based on a slightly general form of equation (2.2.1) on page 44 of this book ($a_{11} = k_{xx}$ and $a_{22} = k_{yy}$):

$$-\left[\frac{\partial}{\partial x} \left(a_{11} \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y} \left(a_{22} \frac{\partial u}{\partial y} \right) \right] + a_{00}u = f \quad \text{in } \Omega \quad (\text{A.2.1})$$

with the boundary conditions [c.f., equation (2.2.5a,b)]

$$u = \hat{u}(s) \quad \text{on } \Gamma_u \quad (\text{A.2.2a})$$

$$\left(a_{11} \frac{\partial u}{\partial x} n_x + a_{22} \frac{\partial u}{\partial y} n_y \right) + \beta(u - u_\infty) = q \quad \text{on } \Gamma_q \quad (\text{A.2.2b})$$

where Γ_u and Γ_q are disjoint portions of the boundary Γ such that $\Gamma = \Gamma_u \cup \Gamma_q$. For a heat transfer problem, $a_{11} = k_{xx}$ and $a_{22} = k_{yy}$ denote conductivities in the x and y directions, respectively, $f(x, y)$ denotes the known internal heat generation per unit volume, (n_x, n_y) denote the direction cosines of the unit normal vector $\hat{\mathbf{n}}$ on the boundary, β denotes the convective heat transfer coefficient, and u_∞ is the ambient temperature.

Equation (A.2.1) arises in a variety of other fields and the program *FEM2DHT* can be used to analyze them as long as the governing equation of the problem solved is a special case of Eq. (A.2.1). For example, slow flows of inviscid fluids are often described in terms of either the *velocity potential* ϕ

$$v_x \equiv -\frac{\partial \phi}{\partial x}, \quad v_y \equiv -\frac{\partial \phi}{\partial y} \quad (\text{A.2.3})$$

or the *stream function* ψ

$$v_y \equiv -\frac{\partial \psi}{\partial x}, \quad v_x \equiv \frac{\partial \psi}{\partial y} \quad (\text{A.2.4})$$

Here (v_x, v_y) denote components of the velocity vector \mathbf{v} . Thus, the program *FEM2DHT* can be used to solve the inviscid flow problems in terms of the velocity potential or the stream function, and then post-compute the velocity field as per Eqs. (A.2.3) and (A.2.4).

A.3 Flows of Viscous Incompressible Fluids

The finite element model of viscous, incompressible flows used in the *FEM2DHT* program is the *reduced integration penalty model* discussed in Section 4.3.3. The flows considered here have no inertia and are therefore termed Stokes flows. The governing equations of the model are a combination of the continuity equation (4.3.1) and momentum equations (4.3.2):

$$-\frac{\partial}{\partial x} \left(2\mu \frac{\partial v_x}{\partial x} \right) - \frac{\partial}{\partial y} \left[\mu \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right) \right] - \gamma \frac{\partial}{\partial x} \left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} \right) = 0 \quad (\text{A.3.1a})$$

$$-\frac{\partial}{\partial x} \left[\mu \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right) \right] - \frac{\partial}{\partial y} \left(2\mu \frac{\partial v_y}{\partial y} \right) - \gamma \frac{\partial}{\partial y} \left(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} \right) = 0 \quad (\text{A.3.1b})$$

with the boundary conditions

$$v_x = \hat{v}_x \text{ on } \Gamma_{v_x} \text{ or } \mathcal{T}_1 = \hat{\mathcal{T}}_1 \text{ on } \Gamma_{\mathcal{T}_1} \quad (\text{A.3.3a})$$

$$v_y = \hat{v}_y \text{ on } \Gamma_{v_y} \text{ or } \mathcal{T}_2 = \hat{\mathcal{T}}_2 \text{ on } \Gamma_{\mathcal{T}_2} \quad (\text{A.3.3b})$$

where Γ_{v_x} and $\Gamma_{\mathcal{T}_1}$, and Γ_{v_y} and $\Gamma_{\mathcal{T}_2}$ are disjoint portions of the boundary Γ such that $\Gamma = \Gamma_{v_x} \cup \Gamma_{\mathcal{T}_1}$ and $\Gamma = \Gamma_{v_y} \cup \Gamma_{\mathcal{T}_2}$. The boundary stress components \mathcal{T}_1 and \mathcal{T}_2 are given by

$$\mathcal{T}_1 = \left(2\mu \frac{\partial v_x}{\partial x} - P \right) n_x + \mu \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right) n_y \quad (\text{A.3.4a})$$

$$\mathcal{T}_2 = \mu \left(\frac{\partial v_x}{\partial y} + \frac{\partial v_y}{\partial x} \right) n_x + \left(2\mu \frac{\partial v_y}{\partial y} - P \right) n_y \quad (\text{A.3.4b})$$

In equations (A.3.3a) and (A.3.3b) γ denotes the *penalty parameter*, and in equations (A.3.4a) and (A.3.4b) P denotes the pressure.

A.4 Description of the Input Data

The following table describes the input data required for the solution of heat transfer and related problems and viscous fluid flow problems using program *FEM2DHT*. Sample input data files are included. For additional examples, the reader may consult the book by Reddy [1, Chapter 13].

Table A.4.1: Description of input variables to computer program *FEM2DHT*.*** Data Card 1**

TITLE - Title of the problem being solved (80 characters)

*** Data Card 2**

ITYPE - Problem type

ITYPE = 0, Heat transfer (and like) problems

ITYPE = 1, Viscous incompressible flow problems

IGRAD - Flag for computing the gradient of the solution

IGRAD = 0, No postprocessing is required

IGRAD > 0, Postprocessing is required

When ITYPE=0 and IGRAD=1, the gradient is computed as in Eq. (A.3); for ITYPE = 0 and IGRAD > 1, the gradient is computed as in Eq. (A.4).

ITEM - Indicator for transient analysis:

ITEM = 0, Steady-state analysis is required

ITEM > 0, Transient analysis is required

*** Data Card 3**

IELTYP - Element type used in the analysis:

IELTYP= 0, Triangular elements

IELTYP> 0, Quadrilateral elements

NPE - Nodes per element:

NPE = 3, Linear triangle (IELTYP=0)

NPE = 4, Linear quadrilateral (IELTYP>0)

NPE = 6, Quadratic triangle (IELTYP=0)

NPE = 8 or 9, Quadratic quadrilateral (IELTYP>0)

MESH - Indicator for mesh generation by the program:

MESH = 0, Mesh is not generated by the program

MESH = 1, Mesh is generated by the program for rectangular domains (by MSH2DR)

MESH > 1, Mesh is generated by the program for general domains (by MSH2DG)

NPRNT - Indicator for printing certain output:

NPRNT = 0, Not print array NOD, element matrices, or global matrices

NPRNT = 1, Print array NOD and Element 1 matrices:

[ELK] and {ELF}

NPRNT = 2, Print array NOD and assembled matrices, [GLK] and {GLF}

NPRNT > 2, Combination of NPRNT=1 and NPRNT=2

*** Data Card 4:** SKIP the card if MESH.EQ.1

NEM - Number of elements in the mesh when the user inputs the mesh or the mesh is generated by MSH2DG

NNM - Number of nodes in the mesh when the user inputs the mesh or the mesh is generated by MSH2DG

* **Data Card 5** SKIP the card if MESH.NE.0; otherwise, the card is
read in a loop on the number of elements (N=1, NEM)
NOD(N,I)-Connectivity for the Nth element (I=1,NPE)

* **Data Card 6** SKIP the card if MESH.NE.0
GLXY(I,J)-Global x and y coordinates of Ith global node in
the mesh (J=1, x-coordinate, J=2, y-coordinate)
Loops on I and J are: ((J=1,2), I=1,NNM); the NNM
pairs of (x,y)--coordinates are read sequentially

_____ The next FOUR data cards are read in subroutine MSH2DG _____
* * * * * SKIP Cards 7, 8, 9, and 10 unless MESH.GT.1 * * * * *

* **Data Card 7**
NRECL - Number of line records to be read in the mesh
* **Data Card 8** Read the following variables NRECL times
NOD1 - First global node number of the line segment
NODL - Last global node number of the line segment
NODINC - Node increment on the line
X1 - The global x-coordinate of NOD1
Y1 - The global y-coordinate of NOD1
XL - The global x-coordinate of NODL
YL - The global y-coordinate of NODL
RATIO - Ratio of the first to the last element lengths

* **Data Card 9**
NRECEL - Number of rows of elements to be read in the mesh
* **Data Card 10** Read the following variables NRECEL times
NEL1 - First element number of the row
NELL - Last element number of the row
IELINC - Increment of element number in the row
NODINC - Increment of global node number in the row
NPE - Number of nodes in each element
NODE(I) - Connectivity array of the first element in the row

* **Data Card 11** SKIP the card if MESH.NE.1
NX - Number of elements in the x-direction
NY - Number of elements in the y-direction
* **Data Card 12** SKIP the card if MESH.NE.1
X0 - The x-coordinate of global node 1
DX(I) - The x-dimension of the Ith element (I=1,NX)
* **Data Card 13** SKIP the card if MESH.NE.1
Y0 - The y-coordinate of global node 1
DY(I) - The y-dimension of the Ith element (I=1,NY)

* **Data Card 14**
NSPV - The number of specified primary variables

* **Data Card 15** SKIP the card if NSPV.EQ.0
ISPV(I,J)-Node number and LOCAL degree of freedom number of the I-th specified
primary variable:
ISPV(I,1)=Node number; ISPV(I,2)=Local DOF number
The loops on I and J are: ((J=1,2),I=1,NSPV)

* **Data Card 16** SKIP the card if NSPV.EQ.0

VSPV(I) - Specified value of the Ith primary variable

* **Data Card 17**

NSSV - Number of (nonzero) specified secondary variables

* **Data Card 18** SKIP the card if NSSV.EQ.0

ISSV(I,J)-Node number and LOCAL degree of freedom number of
the Ith specified secondary variable:

ISSV(I,1)=Node number; ISSV(I,2)=Local DOF number

The loops on I and J are: ((J=1,2),I=1,NSSV)

* **Data Card 19** SKIP the card if NSSV.EQ.0

VSSV(I) - Specified value of the Ith secondary variable
(I=1,NSSV)

Data Cards 20 - 26 are for HEAT TRANSFER PROBLEMS (ITYPE = 0) only

* **Data Card 20** SKIP the card if ITYPE.NE.0

A10 |
A1X | Coefficients of the differential equation
A1Y | a11 = A10 + A1X*X + A1Y*Y

* **Data Card 21** SKIP the card if ITYPE.NE.0

A20 |
A2X | Coefficients of the differential equation
A2Y | a22 = A20 + A2X*X + A2Y*Y

* **Data Card 22** SKIP the card if ITYPE.NE.0

A00 - Coefficient of the differential equation

* **Data Card 23** SKIP the card if ITYPE.NE.0

ICONV - Indicator for convection boundary conditions
ICONV = 0, No convection boundary conditions
ICONV > 0, Convection boundary conditions present

* **Data Card 24** SKIP the card if ITYPE.NE.0 or ICONV.EQ.0

NBE - Number elements with convection

* **Data Card 25** SKIP the card if ITYPE.NE.0 or ICONV.EQ.0

The following cards are read for each I, I=1,NBE

IBN(I) - Ith element number with convection

BETA(I) - Film coefficient for convection on Ith element

TINF(I) - Ambient temperature of the Ith element

* **Data Card 26** SKIP the card if ITYPE.NE.0 or ICONV.EQ.0

INOD(I,J)- Local node numbers of the side with convection

(J=1,2; for quadratic elements, give end nodes)

Loops on I and J are: ((J=1,2), I=1,NBE)

_____ Data Card 27 is for VISCOUS FLUID FLOWS (ITYPE = 1) only _____

* **Data Card 27** SKIP the card if ITYPE.EQ.0

AMU - Viscosity of the fluid

PENLTY - Penalty parameter used

_____ Remaining data cards are for ALL problem types _____

*** Data Card 28**

F0 |
FX | Coefficients to define the source term:
FY | $f = F0 + FX*x + FY*y$

_____ Cards 29 thru 32 are for TRANSIENT ANALYSIS only _____

*** Data Card 29** SKIP the card if ITEM.EQ.0

C0 | Coefficients defining the temporal parts of the
CX | differential equations:
CY | $CT = C0 + CX*x + CY*y$

*** Data Card 30** SKIP the card if ITEM.EQ.0

NTIME - Number of time steps for the transient solution
NSTP - Time step number at which the source is removed
INTVL - Time step interval at which to print the solution
INTIAL - Indicator for nature of initial conditions:
INTIAL=0, Zero initial conditions are used
INTIAL>0, Non-zero initial conditions are used

*** Data Card 31** SKIP the card if ITEM.EQ.0

DT - Time step used for the transient solution
ALFA - Parameter in the alfa-family of time approximation
used:
ALFA=0, The forward difference scheme (C.S.)@
ALFA=0.5, The Crank-Nicolson scheme (stable)
ALFA=2/3, The Galerkin scheme (stable)
ALFA=1, The backward difference scheme (stable)
@C.S.=Conditionally Stable; For all schemes with
ALFA < 0.5, the time step DT is restricted to:
DT < 2/[MAXEGN*(1-2*ALFA)], where MAXEGN is the
maximum eigenvalue of the discrete problem.
EPSLN - A small parameter to check if the solution has
reached a steady state

*** Data Card 32** SKIP the card if ITEM or INTIAL.EQ.0

GLU(I) - Vector of initial values of the primary variables
(I=1,NEQ, NEQ=Number of nodal values in the mesh)

Example 1: Convective heat transfer in a square region (see Figure A.4.1) is solved using 2×2 mesh of nine-node rectangular elements. The conductivities are taken to be $a_{11} = a_{22} = 10 \text{ W}/(\text{m } ^\circ\text{C})$. Echo of the input data along with the output from program *FEM2DHT* is given below and on the next couple of pages.

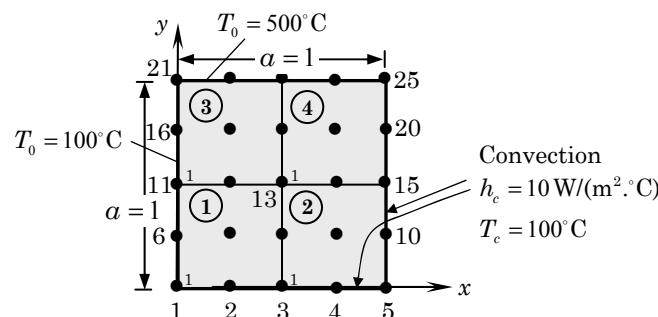


Figure A.4.1: Domain, boundary conditions, and mesh used for the convective heat transfer problem in Example 1.

Input data and output for Example 1: _____

Example 1: CONVECTIVE heat transfer in a square region

```

0 1 0 0          ITYPE,IGRAD,ITEM,NEIGN
2 9 1 0          IEL, NPE, MESH, NPRNT
2 2              NX, NY
0.0 0.5 0.5      X0, DX(1), DX(2)
0.0 0.5 0.5      Y0, DY(1), DY(2)
9                NSPV
1 1 6 1 11 1 16 1 21 1 22 1 23 1 24 1 25 1      ISPV(I,J)
100.0 100.0 100.0 500.0 500.0 500.0 500.0 500.0 500.0  VSPV(I)
0                NSSV
10.0 0.0 0.0     A10, A1X, A1Y
10.0 0.0 0.0     A20, A2X, A2Y
0.0              A00
1                ICONV
4                NBE
1 10.0 100.0 2 10.0 100.0 2 10.0 100.0 4 10.0 100.0    IBM,BETA,TINF
1 2 1 2 2 3 2 3 INOD(I,J)
0.0 0.0 0.0       F0, FX, FY

```

Example 1: CONVECTIVE heat transfer in a square region

OUTPUT FROM PROGRAM *FEM2DHT* BY J. N. REDDY

ANALYSIS OF A HEAT TRANSFER PROBLEM

COEFFICIENTS OF THE DIFFERENTIAL EQUATION:

```

Coefficient, A10 .....= 0.1000E+02
Coefficient, A1X .....= 0.0000E+00
Coefficient, A1Y .....= 0.0000E+00
Coefficient, A20 .....= 0.1000E+02
Coefficient, A2X .....= 0.0000E+00
Coefficient, A2Y .....= 0.0000E+00
Coefficient, A00 .....= 0.0000E+00

```

CONVECTIVE HEAT TRANSFER DATA:

Number of elements with convection, NBE .= 4

Elements, their LOCAL nodes and convective
heat transfer data:

Ele. No.	End Nodes	Film Coeff.	T-Infinity
----------	-----------	-------------	------------

1	1 2	0.10000E+02	0.10000E+03
2	1 2	0.10000E+02	0.10000E+03
2	2 3	0.10000E+02	0.10000E+03
4	2 3	0.10000E+02	0.10000E+03

CONTINUOUS SOURCE COEFFICIENTS:

```

Coefficient, F0 .....= 0.0000E+00
Coefficient, FX .....= 0.0000E+00
Coefficient, FY .....= 0.0000E+00

```

***** A STEADY-STATE PROBLEM is analyzed *****

*** A mesh of QUADRILATERALS is chosen by user ***

FINITE ELEMENT MESH INFORMATION:

Element type: 0 = Triangle; >0 = Quad.).....= 2
 Number of nodes per element, NPE= 9
 No. of primary deg. of freedom/node, NDF= 1
 Number of elements in the mesh, NEM= 4
 Number of nodes in the mesh, NNM= 25
 Number of equations to be solved, NEQ= 25
 Half bandwidth of the matrix GLK, NHBW ...= 13
 Mesh subdivisions, NX and NY= 2 2
 No. of specified PRIMARY variables, NSPV ...= 9

Node	x-coord.	y-coord.	Speci. primary & secondary variables (0, unspecified; >0, specified)	
			Primary DOF	Secondary DOF
1	0.0000E+00	0.0000E+00	1	0
2	0.2500E+00	0.0000E+00	0	0
3	0.5000E+00	0.0000E+00	0	0
4	0.7500E+00	0.0000E+00	0	0
5	0.1000E+01	0.0000E+00	0	0
6	0.0000E+00	0.2500E+00	1	0
7	0.2500E+00	0.2500E+00	0	0
8	0.5000E+00	0.2500E+00	0	0
9	0.7500E+00	0.2500E+00	0	0
10	0.1000E+01	0.2500E+00	0	0
11	0.0000E+00	0.5000E+00	1	0
12	0.2500E+00	0.5000E+00	0	0
13	0.5000E+00	0.5000E+00	0	0
14	0.7500E+00	0.5000E+00	0	0
15	0.1000E+01	0.5000E+00	0	0
16	0.0000E+00	0.7500E+00	1	0
17	0.2500E+00	0.7500E+00	0	0
18	0.5000E+00	0.7500E+00	0	0
19	0.7500E+00	0.7500E+00	0	0
20	0.1000E+01	0.7500E+00	0	0
21	0.0000E+00	0.1000E+01	1	0
22	0.2500E+00	0.1000E+01	1	0
23	0.5000E+00	0.1000E+01	1	0
24	0.7500E+00	0.1000E+01	1	0
25	0.1000E+01	0.1000E+01	1	0

NUMERICAL INTEGRATION DATA:

Full quadrature (IPDF x IPDF) rule, IPDF = 3
 Reduced quadrature (IPDR x IPDR), IPDR = 2
 Quadrature rule used in postproc., ISTR = 2

SOLUTION :

Node	x-coord.	y-coord.	Temperature
1	0.00000E+00	0.00000E+00	0.10000E+03
2	0.25000E+00	0.00000E+00	0.11380E+03
3	0.50000E+00	0.00000E+00	0.13273E+03
4	0.75000E+00	0.00000E+00	0.13350E+03
5	0.10000E+01	0.00000E+00	0.11697E+03
6	0.00000E+00	0.25000E+00	0.10000E+03
7	0.25000E+00	0.25000E+00	0.14268E+03
8	0.50000E+00	0.25000E+00	0.17135E+03
9	0.75000E+00	0.25000E+00	0.17561E+03
10	0.10000E+01	0.25000E+00	0.15241E+03
11	0.00000E+00	0.50000E+00	0.10000E+03
12	0.25000E+00	0.50000E+00	0.18453E+03
13	0.50000E+00	0.50000E+00	0.24111E+03
14	0.75000E+00	0.50000E+00	0.24392E+03
15	0.10000E+01	0.50000E+00	0.21538E+03
16	0.00000E+00	0.75000E+00	0.10000E+03
17	0.25000E+00	0.75000E+00	0.29562E+03
18	0.50000E+00	0.75000E+00	0.34509E+03
19	0.75000E+00	0.75000E+00	0.35267E+03
20	0.10000E+01	0.75000E+00	0.30641E+03
21	0.00000E+00	0.10000E+01	0.50000E+03
22	0.25000E+00	0.10000E+01	0.50000E+03
23	0.50000E+00	0.10000E+01	0.50000E+03
24	0.75000E+00	0.10000E+01	0.50000E+03
25	0.10000E+01	0.10000E+01	0.50000E+03

The orientation of gradient vector is measured from the positive x-axis
x-coord. y-coord. -a11(du/dx) -a22(du/dy) Flux Mgntd Orientation

0.1057E+00	0.1057E+00	-0.9881E+03	-0.5666E+03	0.1139E+04	-150.17
0.1057E+00	0.3943E+00	-0.2681E+04	-0.7906E+03	0.2795E+04	-163.57
0.3943E+00	0.1057E+00	-0.8220E+03	-0.1403E+04	0.1626E+04	-120.37
0.3943E+00	0.3943E+00	-0.1632E+04	-0.2457E+04	0.2950E+04	-123.59
0.6057E+00	0.1057E+00	-0.1830E+03	-0.1566E+04	0.1577E+04	-96.67
0.6057E+00	0.3943E+00	-0.2553E+03	-0.2872E+04	0.2884E+04	-95.08
0.8943E+00	0.1057E+00	0.8494E+03	-0.1499E+04	0.1723E+04	-60.46
0.8943E+00	0.3943E+00	0.1152E+04	-0.2709E+04	0.2944E+04	-66.97
0.1057E+00	0.6057E+00	-0.7097E+04	-0.1718E+04	0.7302E+04	-166.39
0.1057E+00	0.8943E+00	-0.5095E+04	-0.1272E+05	0.1370E+05	-111.83
0.3943E+00	0.6057E+00	-0.2009E+04	-0.4743E+04	0.5151E+04	-112.95
0.3943E+00	0.8943E+00	-0.7522E+03	-0.6432E+04	0.6476E+04	-96.67
0.6057E+00	0.6057E+00	-0.4085E+03	-0.4237E+04	0.4257E+04	-95.51
0.6057E+00	0.8943E+00	-0.2876E+03	-0.5918E+04	0.5925E+04	-92.78
0.8943E+00	0.6057E+00	0.1909E+04	-0.3846E+04	0.4293E+04	-63.60
0.8943E+00	0.8943E+00	0.1194E+04	-0.6904E+04	0.7006E+04	-80.19

Example 2: Flow of an incompressible viscous fluid in a slider bearing (see Figure A.4.2) is analyzed using a mesh of six nine-node elements. The mesh used is to illustrate the input data, and it is not representative of the mesh required to obtain an accurate solution. The input data and an edited output from program *FEM2DHT* are given below.

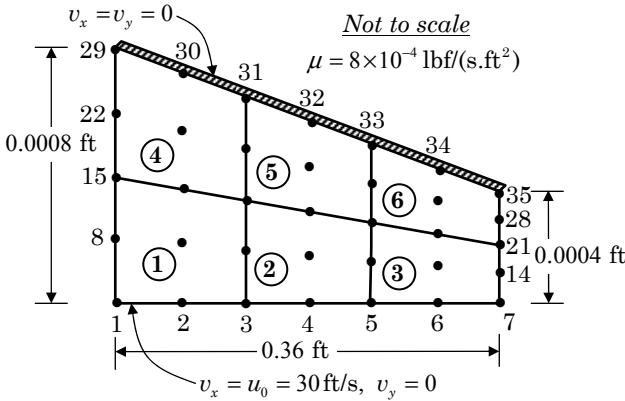


Figure A.4.2: Domain and mesh used for the flow of a viscous incompressible fluid in a slider bearing (Example 2).

Input data and output for Example 2:

```

Example 2: Flow of a viscous fluid in a slider bearing
1 1 0          ITYPE,ISTRS,ITEM
1 9 2 0        IELTYP,NPE,MESH,NPRNT
6 35           NEM, NNM
5              NRECL
1 7 1 0.0 0.0   0.36 0.0    1.0  NOD1,NODL,NODINC,...
8 14 1 0.0 5.0E-5 0.36 5.0E-5 1.0
15 21 1 0.0 1.0E-4 0.36 1.0E-4 1.0
22 28 1 0.0 3.5E-4 0.36 1.5E-4 1.0
29 35 1 0.0 8.0E-4 0.36 4.0E-4 1.0
2              NRECEL
1 3 1 2 9     1 3 17 15 2 10 16 8 9 NEL1,NELL,..
4 6 1 2 9     15 17 31 29 16 24 30 22 23
28             NSPV
1 1 1 2 2 1 2 2 3 1 3 2 4 1 4 2 5 1 5 2
6 1 6 2 7 1 7 2 29 1 29 2 30 1 30 2 31 1 31 2
32 1 32 2 33 1 33 2 34 1 34 2 35 1 35 2 ISPV(I,J)
30.0 0.0 30.0 0.0 30.0 0.0 30.0 0.0 30.0 0.0
30.0 0.0 30.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0      VSPV(I)
0              NSSV
8.0E-4 8.0E12  AMU, PENLTY
0.0 0.0 0.0     F0, FX, FY

```

Example 2: Flow of a viscous fluid in a slider bearing

OUTPUT FROM PROGRAM *FEM2DHT* BY J. N. REDDY

A VISCOUS INCOMPRESSIBLE FLOW IS ANALYZED

PARAMETERS OF THE FLUID FLOW PROBLEM:

Viscosity of the fluid, AMU= 0.8000E-03
 Penalty parameter, PENLTY= 0.8000E+13

CONTINUOUS SOURCE COEFFICIENTS:

Coefficient, F0= 0.0000E+00
 Coefficient, FX= 0.0000E+00
 Coefficient, FY= 0.0000E+00

***** A STEADY-STATE PROBLEM is analyzed *****
 *** A mesh of QUADRILATERALS is chosen by user ***

FINITE ELEMENT MESH INFORMATION:

Element type: 0 = Triangle; >0 = Quad.).....= 1
 Number of nodes per element, NPE= 9
 No. of primary deg. of freedom/node, NDF= 2
 Number of elements in the mesh, NEM= 6
 Number of nodes in the mesh, NNM= 35
 Number of equations to be solved, NEQ= 70
 Half bandwidth of the matrix GLK, NHBW ..= 34
 No. of specified PRIMARY variables, NSPV ..= 28

Node	x-coord.	y-coord.	Speci. primary & secondary variables (0, unspecified; >0, specified)			
			Primary DOF	Secondary DOF		

1	0.0000E+00	0.0000E+00	1	2	0	0
2	0.6000E-01	0.0000E+00	1	2	0	0
3	0.1200E+00	0.0000E+00	1	2	0	0
4	0.1800E+00	0.0000E+00	1	2	0	0
5	0.2400E+00	0.0000E+00	1	2	0	0
6	0.3000E+00	0.0000E+00	1	2	0	0
7	0.3600E+00	0.0000E+00	1	2	0	0
8	0.0000E+00	0.5000E-04	0	0	0	0
9	0.6000E-01	0.5000E-04	0	0	0	0
10	0.1200E+00	0.5000E-04	0	0	0	0
11	0.1800E+00	0.5000E-04	0	0	0	0
12	0.2400E+00	0.5000E-04	0	0	0	0
13	0.3000E+00	0.5000E-04	0	0	0	0
14	0.3600E+00	0.5000E-04	0	0	0	0
15	0.0000E+00	0.1000E-03	0	0	0	0
16	0.6000E-01	0.1000E-03	0	0	0	0
17	0.1200E+00	0.1000E-03	0	0	0	0
18	0.1800E+00	0.1000E-03	0	0	0	0
19	0.2400E+00	0.1000E-03	0	0	0	0
20	0.3000E+00	0.1000E-03	0	0	0	0
21	0.3600E+00	0.1000E-03	0	0	0	0
22	0.0000E+00	0.3500E-03	0	0	0	0

23	0.6000E-01	0.3167E-03	0	0	0	0
24	0.1200E+00	0.2833E-03	0	0	0	0
25	0.1800E+00	0.2500E-03	0	0	0	0
26	0.2400E+00	0.2167E-03	0	0	0	0
27	0.3000E+00	0.1833E-03	0	0	0	0
28	0.3600E+00	0.1500E-03	0	0	0	0
29	0.0000E+00	0.8000E-03	1	2	0	0
30	0.6000E-01	0.7333E-03	1	2	0	0
31	0.1200E+00	0.6667E-03	1	2	0	0
32	0.1800E+00	0.6000E-03	1	2	0	0
33	0.2400E+00	0.5333E-03	1	2	0	0
34	0.3000E+00	0.4667E-03	1	2	0	0
35	0.3600E+00	0.4000E-03	1	2	0	0

NUMERICAL INTEGRATION DATA:

Full quadrature (IPDF x IPDF) rule, IPDF = 3

Reduced quadrature (IPDR x IPDR), IPDR = 2

Quadrature rule used in postproc., ISTR = 2

SOLUTION:

Node	x-coord.	y-coord.	Velocity, u	Velocity, v
1	0.00000E+00	0.00000E+00	0.30000E+02	0.00000E+00
2	0.60000E-01	0.00000E+00	0.30000E+02	0.00000E+00
3	0.12000E+00	0.00000E+00	0.30000E+02	0.00000E+00
4	0.18000E+00	0.00000E+00	0.30000E+02	0.00000E+00
5	0.24000E+00	0.00000E+00	0.30000E+02	0.00000E+00
6	0.30000E+00	0.00000E+00	0.30000E+02	0.00000E+00
7	0.36000E+00	0.00000E+00	0.30000E+02	0.00000E+00
8	0.00000E+00	0.50000E-04	0.26008E+02	-0.45304E-02
9	0.60000E-01	0.50000E-04	0.25983E+02	0.22683E-02
10	0.12000E+00	0.50000E-04	0.26085E+02	-0.46442E-02
11	0.18000E+00	0.50000E-04	0.26308E+02	0.21300E-02
12	0.24000E+00	0.50000E-04	0.26982E+02	-0.50555E-02
13	0.30000E+00	0.50000E-04	0.28270E+02	0.14714E-02
14	0.36000E+00	0.50000E-04	0.31533E+02	-0.68447E-02
15	0.00000E+00	0.10000E-03	0.22292E+02	0.10406E-01
16	0.60000E-01	0.10000E-03	0.22259E+02	-0.51970E-02
17	0.12000E+00	0.10000E-03	0.22421E+02	0.10016E-01
18	0.18000E+00	0.10000E-03	0.22847E+02	-0.57272E-02
19	0.24000E+00	0.10000E-03	0.23933E+02	0.86438E-02
20	0.30000E+00	0.10000E-03	0.26237E+02	-0.81796E-02
21	0.36000E+00	0.10000E-03	0.31469E+02	0.26246E-02
22	0.00000E+00	0.35000E-03	0.87401E+01	0.14297E-01
23	0.60000E-01	0.31667E-03	0.10255E+02	-0.80107E-02
24	0.12000E+00	0.28333E-03	0.12079E+02	0.12563E-01
25	0.18000E+00	0.25000E-03	0.14350E+02	-0.99138E-02
26	0.24000E+00	0.21667E-03	0.17197E+02	0.97258E-02
27	0.30000E+00	0.18333E-03	0.20956E+02	-0.13382E-01
28	0.36000E+00	0.15000E-03	0.25982E+02	0.38986E-02
29	0.00000E+00	0.80000E-03	0.00000E+00	0.00000E+00
30	0.60000E-01	0.73333E-03	0.00000E+00	0.00000E+00

31	0.12000E+00	0.66667E-03	0.00000E+00	0.00000E+00
32	0.18000E+00	0.60000E-03	0.00000E+00	0.00000E+00
33	0.24000E+00	0.53333E-03	0.00000E+00	0.00000E+00
34	0.30000E+00	0.46667E-03	0.00000E+00	0.00000E+00
35	0.36000E+00	0.40000E-03	0.00000E+00	0.00000E+00

x-coord.	y-coord.	sigma-x	sigma-y	sigma-xy	pressure
0.2536E-01	0.2113E-04	-0.2325E+04	-0.2325E+04	-0.6464E+02	0.2325E+04
0.2536E-01	0.7887E-04	-0.2324E+04	-0.2324E+04	-0.5930E+02	0.2324E+04
0.9464E-01	0.2113E-04	-0.8666E+04	-0.8666E+04	-0.6391E+02	0.8666E+04
0.9464E-01	0.7887E-04	-0.8666E+04	-0.8666E+04	-0.5883E+02	0.8666E+04
0.1454E+00	0.2113E-04	-0.1311E+05	-0.1311E+05	-0.6235E+02	0.1311E+05
0.1454E+00	0.7887E-04	-0.1311E+05	-0.1311E+05	-0.5731E+02	0.1311E+05
0.2146E+00	0.2113E-04	-0.1771E+05	-0.1771E+05	-0.5387E+02	0.1771E+05
0.2146E+00	0.7887E-04	-0.1771E+05	-0.1771E+05	-0.5184E+02	0.1771E+05
0.2654E+00	0.2113E-04	-0.1858E+05	-0.1858E+05	-0.4341E+02	0.1858E+05
0.2654E+00	0.7887E-04	-0.1859E+05	-0.1859E+05	-0.4380E+02	0.1859E+05
0.3346E+00	0.2113E-04	-0.9608E+04	-0.9608E+04	-0.2520E+00	0.9608E+04
0.3346E+00	0.7887E-04	-0.9607E+04	-0.9607E+04	-0.1734E+02	0.9607E+04
0.2536E-01	0.1753E-03	-0.2323E+04	-0.2323E+04	-0.4786E+02	0.2323E+04
0.2536E-01	0.5632E-03	-0.2323E+04	-0.2323E+04	-0.1607E+02	0.2323E+04
0.9464E-01	0.1590E-03	-0.8666E+04	-0.8666E+04	-0.4855E+02	0.8666E+04
0.9464E-01	0.5025E-03	-0.8667E+04	-0.8667E+04	-0.2187E+02	0.8667E+04
0.1454E+00	0.1471E-03	-0.1311E+05	-0.1311E+05	-0.4832E+02	0.1311E+05
0.1454E+00	0.4580E-03	-0.1311E+05	-0.1311E+05	-0.2752E+02	0.1311E+05
0.2146E+00	0.1309E-03	-0.1771E+05	-0.1771E+05	-0.4730E+02	0.1771E+05
0.2146E+00	0.3973E-03	-0.1771E+05	-0.1771E+05	-0.3830E+02	0.1771E+05
0.2654E+00	0.1190E-03	-0.1859E+05	-0.1859E+05	-0.4532E+02	0.1859E+05
0.2654E+00	0.3529E-03	-0.1859E+05	-0.1859E+05	-0.4934E+02	0.1859E+05
0.3346E+00	0.1027E-03	-0.9605E+04	-0.9605E+04	-0.6196E+02	0.9605E+04
0.3346E+00	0.2922E-03	-0.9606E+04	-0.9606E+04	-0.7193E+02	0.9606E+04

A.5 Source Listings of Selective Subroutines

The computer implementation of the finite element formulations presented in Chapters 2 and 3 can be found in Chapter 13 of [1]. A complete listing of the FORTRAN source of FEM2D is also included in [1]. The program *FEM2DHT* consists of the following subroutines:

BOUNRY: Imposes applied boundary conditions on primary and secondary variables.

CONCT: Assembles element coefficient matrices and vectors.

ECHO: Echoes the input file.

ELKMF: Generates element coefficient matrices and vectors for rectangular (quadrilateral) elements.

ELKMFT: Generates element coefficient matrices and vectors for triangular elements.

MSH2DG: Generates finite element meshes for some general domains (see [1] for details).

MSH2DR: Generates finite element meshes for only rectangular domains (see [1] for details).

PSTPRC: Postprocessor

SHPRCT: Evaluates shape (or interpolation) functions and their global derivatives for rectangular (isoparametric) elements.

SHPTRI: Evaluates shape (or interpolation) functions and their global derivatives for triangular (isoparametric) elements.

SOLVER: Solves symmetric, banded system of algebraic equations.

TEMPORAL: Sets up coefficient matrices and column vectors for transient analysis.

To aid the reader in the computer implementation of the finite element formulations presented in Chapters 2 and 3, listings of subroutines ELKMFR, SHPRCT, and TEMPORAL from program *FEM2DHT*, which illustrate the computation of element matrices for quadrilateral elements, are included in the remaining pages of this appendix. The complete FORTRAN source of program *FEM2DHT* or *FEM2D* from [1] may be obtained for a small charge from the first author.

Reference for Additional Reading

1. J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw-Hill, New York (2006).

Listings of Subroutines *ELKMFR*, *SHPRCT*, and *TEMPORAL*

```

SUBROUTINE ELKMFR(NPE,NN,ITYPE)
C
C   Element calculations based on linear and quadratic rectangular
C   elements and isoparametric formulation are carried out for the
C   heat transfer and penalty model of fluid flow.
C
C
IMPLICIT REAL*8(A-H,O-Z)
COMMON/STF/ELF(18),ELK(18,18),ELM(18,18),ELXY(9,2),ELU(18),A1,A2
COMMON/PST/A10,A1X,A1Y,A20,A2X,A2Y,A00,C0,CX,CY,F0,FX,FY,
1      AMU,PENLTY
COMMON/SHP/SF(9),GDSF(2,9),SFH(16)
COMMON/PNT/IPDF,IPDR,NIPF,NIPR
DIMENSION GAUSPT(5,5),GAUSWT(5,5)
COMMON/IO/IN,ITT
DATA GAUSPT/5*0.0D0, -0.57735027D0, 0.57735027D0, 3*0.0D0,
2 -0.77459667D0, 0.0D0, 0.77459667D0, 2*0.0D0, -0.86113631D0,
3 -0.33998104D0, 0.33998104D0, 0.86113631D0, 0.0D0, -0.90617984D0,
4 -0.53846931D0, 0.0D0, 0.53846931D0, 0.90617984D0/
      DATA GAUSWT/2.0D0, 4*0.0D0, 2*1.0D0, 3*0.0D0, 0.55555555D0,
2 0.8888888D0, 0.55555555D0, 2*0.0D0, 0.34785485D0,
3 2*0.65214515D0, 0.34785485D0, 0.0D0, 0.23692688D0,
4 0.47862867D0, 0.56888888D0, 0.47862867D0, 0.23692688D0/

```

```

NDF = NN/NPE
NET=NPE
C
C Initialize the arrays
C
DO 120 I = 1,NN
ELF(I) = 0.0
DO 120 J = 1,NN
IF(ITEM.NE.0) THEN
    ELM(I,J)= 0.0
ENDIF
120 ELK(I,J)= 0.0
C
C Do-loops on numerical (Gauss) integration begin here. Subroutine
C SHPRCT (SHaPe functions for ReCTangular elements) is called here
C
DO 200 NI = 1,IPDF
DO 200 NJ = 1,IPDF
XI = GAUSPT(NI,IPDF)
ETA = GAUSPT(NJ,IPDF)
CALL SHPRCT (NPE,XI,ETA,DET,ELXY,NDF,ITYPE)
CNST = DET*GAUSWT(NI,IPDF)*GAUSWT(NJ,IPDF)
X=0.0
Y=0.0
DO 140 I=1,NPE
X=X+ELXY(I,1)*SF(I)
140 Y=Y+ELXY(I,2)*SF(I)
C
SOURCE=F0+FX*X+FY*Y
IF(ITEM.NE.0) THEN
    CT=C0+CX*X+CY*Y
ENDIF
IF(ITYPE.LE.0) THEN
    A11=A10+A1X*X+A1Y*Y
    A22=A20+A2X*X+A2Y*Y
ENDIF
C
II=1
DO 180 I=1,NET
JJ=1
DO 160 J=1,NET
S00=SF(I)*SF(J)*CNST
S11=GDSF(1,I)*GDSF(1,J)*CNST
S22=GDSF(2,I)*GDSF(2,J)*CNST
S12=GDSF(1,I)*GDSF(2,J)*CNST
S21=GDSF(2,I)*GDSF(1,J)*CNST
IF(ITYPE.EQ.0) THEN

```

472 THE FINITE ELEMENT METHOD IN HEAT TRANSFER AND FLUID DYNAMICS

```

C
C Heat transfer and like problems (i.e. single DOF problems):_____
C
    ELK(I,J) = ELK(I,J) + A11*S11 + A22*S22 + A00*S00
    IF(ITEM.NE.0) THEN
        ELM(I,J) = ELM(I,J) + CT*S00
    ENDIF
    ELSE
C
C Viscous incompressible fluids:_____
C Compute coefficients associated with viscous terms (full integ.)
C
        ELK(II,JJ)      = ELK(II,JJ)      + AMU*(2.0*S11 + S22)
        ELK(II+1,JJ)   = ELK(II+1,JJ)   + AMU*S12
        ELK(II,JJ+1)   = ELK(II,JJ+1)   + AMU*S21
        ELK(II+1,JJ+1)= ELK(II+1,JJ+1) + AMU*(S11 + 2.0*S22)
        IF(ITEM.NE.0) THEN
            ELM(II,JJ)      = ELM(II,JJ)      + CT*S00
            ELM(II+1,JJ+1)= ELM(II+1,JJ+1) + CT*S00
        ENDIF
    ENDIF
    ENDIF
160 JJ = NDF*j+1
C
C Source of the form fx = F0 + FX*X + FY*Y is assumed
C
    L=(I-1)*NDF+1
    ELF(L) = ELF(L)+CNST*SF(I)*SOURCE
180 II = NDF*I+1
200 CONTINUE
C
    IF(ITYPE.GT.0) THEN
C
C Use reduced integration to evaluate coefficients associated with
C penalty terms for flows of viscous incompressible fluids.
C
        DO 280 NI=1,IPDR
        DO 280 NJ=1,IPDR
        XI = GAUSPT(NI,IPDR)
        ETA = GAUSPT(NJ,IPDR)
        CALL SHPRCT (NPE,XI,ETA,DET,ELXY,NDF,ITYPE)
        CNST=DET*GAUSWT(NI,IPDR)*GAUSWT(NJ,IPDR)
C
        II=1
        DO 260 I=1,NPE
        JJ = 1
        DO 240 J=1,NPE
        S11=GDSF(1,I)*GDSF(1,J)*CNST
        S22=GDSF(2,I)*GDSF(2,J)*CNST
        S12=GDSF(1,I)*GDSF(2,J)*CNST
        S21=GDSF(2,I)*GDSF(1,J)*CNST
        ELK(II,JJ) = ELK(II,JJ) + PENLTY*S11
        ELK(II+1,JJ) = ELK(II+1,JJ) + PENLTY*S21
        ELK(II,JJ+1) = ELK(II,JJ+1) + PENLTY*S12
        ELK(II+1,JJ+1)= ELK(II+1,JJ+1) + PENLTY*S22
240    JJ=NDF*j+1
260    II=NDF*I+1
280    CONTINUE
    ENDIF

```

```

IF(ITEM.NE.0) THEN
C
C Compute the coefficient matrices of the final algebraic equations
C (i.e., after time approximation) in the transient analysis:_____
C
    CALL TEMPORAL(NN)
ENDIF
RETURN
END

```

```

SUBROUTINE SHPRCT(NPE,XI,ETA,DET,ELXY,NDF,ITYPE)
C _____
C
C The subroutine evaluates the interpolation functions (SF(I)) and
C their derivatives with respect to global coordinates (GDSF(I,J))
C for Lagrange linear & quadratic rectangular elements, using the
C isoparametric formulation. The subroutine also evaluates Hermite
C interpolation functions and their global derivatives using the
C subparametric formulation.
C
C SF(I).....Interpolation function for node I of the element
C DSF(J,I).....Derivative of SF(I) with respect to XI if J=1 and
C                 and ETA if J=2
C GDSF(J,I)....Derivative of SF(I) with respect to X if J=1 and
C                 and Y if J=2
C XNODE(I,J)..J-TH (J=1,2) Coordinate of node I of the element
C NP(I).....Array of element nodes (used to define SF and DSF)
C GJ(I,J).....Jacobian matrix
C GJINV(I,J)...Inverse of the jacobian matrix
C _____
C
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION ELXY(9,2),XNODE(9,2),NP(9),DSF(2,9),GJ(2,2),GJINV(2,2)
COMMON/SHP/SF(9),GDSF(2,9)
COMMON/IO/IN,ITT
DATA XNODE/-1.0D0, 2*1.0D0, -1.0D0, 0.0D0, 1.0D0, 0.0D0, -1.0D0,
*      0.0D0, 2*-1.0D0, 2*1.0D0, -1.0D0, 0.0D0, 1.0D0, 2*0.0D0/
DATA NP/1,2,3,4,5,7,6,8,9/
C
FNC(A,B) = A*B
IF(NPE.EQ.4) THEN
C
C LINEAR Lagrange interpolation functions for FOUR-NODE element
C
DO 10 I = 1, NPE
XP = XNODE(I,1)
YP = XNODE(I,2)
XI0 = 1.0+XI*XP
ETA0=1.0+ETA*YP
SF(I) = 0.25*FNC(XI0,ETA0)
DSF(1,I)= 0.25*FNC(XP,ETA0)
10   DSF(2,I)= 0.25*FNC(YP,XI0)
ELSE
IF(NPE.EQ.8) THEN

```

```

C
C QUADRATIC Lagrange interpolation functions for EIGHT-NODE element
C
DO 20 I = 1, NPE
NI = NP(I)
XP = XNODE(NI,1)
YP = XNODE(NI,2)
XI0 = 1.0+XI*XP
ETA0 = 1.0+ETA*YP
XI1 = 1.0-XI*XI
ETA1 = 1.0-ETA*ETA
IF(I.LE.4) THEN
SF(NI) = 0.25*FNC(XI0,ETA0)*(XI*XP+ETA*YP-1.0)
DSF(1,NI) = 0.25*FNC(ETA0,XP)*(2.0*XI*XP+ETA*YP)
DSF(2,NI) = 0.25*FNC(XI0,YP)*(2.0*ETA*YP+XI*XP)
ELSE
IF(I.LE.6) THEN
SF(NI) = 0.5*FNC(XI1,ETA0)
DSF(1,NI) = -FNC(XI,ETA0)
DSF(2,NI) = 0.5*FNC(YP,XI1)
ELSE
SF(NI) = 0.5*FNC(ETA1,XI0)
DSF(1,NI) = 0.5*FNC(XP,ETA1)
DSF(2,NI) = -FNC(ETA,XI0)
ENDIF
ENDIF
20 CONTINUE
ELSE
C
C QUADRATIC Lagrange interpolation functions for NINE-NODE element
C
DO 30 I=1,NPE
NI = NP(I)
XP = XNODE(NI,1)
YP = XNODE(NI,2)
XI0 = 1.0+XI*XP
ETA0 = 1.0+ETA*YP
XI1 = 1.0-XI*XI
ETA1 = 1.0-ETA*ETA
XI2 = XP*XI
ETA2 = YP*ETA
IF(I .LE. 4) THEN
SF(NI) = 0.25*FNC(XI0,ETA0)*XI2*ETA2
DSF(1,NI)= 0.25*XP*FNC(ETA2,ETA0)*(1.0+2.0*XI2)
DSF(2,NI)= 0.25*YP*FNC(XI2,XI0)*(1.0+2.0*ETA2)
ELSE
IF(I .LE. 6) THEN
SF(NI) = 0.5*FNC(XI1,ETA0)*ETA2
DSF(1,NI) = -XI*FNC(ETA2,ETA0)
DSF(2,NI) = 0.5*FNC(XI1,YP)*(1.0+2.0*ETA2)
ELSE
IF(I .LE. 8) THEN
SF(NI) = 0.5*FNC(ETA1,XI0)*XI2
DSF(2,NI) = -ETA*FNC(XI2,XI0)
DSF(1,NI) = 0.5*FNC(ETA1,XP)*(1.0+2.0*XI2)
ELSE
SF(NI) = FNC(XI1,ETA1)
DSF(1,NI) = -2.0*XI*ETA1
DSF(2,NI) = -2.0*ETA*XI1
ENDIF
ENDIF

```

```

          ENDIF
          ENDIF
30      CONTINUE
          ENDIF
          ENDIF
C
C Compute the Jacobian matrix [GJ] and its inverse [GJINV]
C
        DO 40 I = 1,2
        DO 40 J = 1,2
        GJ(I,J) = 0.0
        DO 40 K = 1,NPE
40    GJ(I,J) = GJ(I,J) + DSF(I,K)*ELXY(K,J)
C
        DET = GJ(1,1)*GJ(2,2)-GJ(1,2)*GJ(2,1)
        GJINV(1,1) = GJ(2,2)/DET
        GJINV(2,2) = GJ(1,1)/DET
        GJINV(1,2) = -GJ(1,2)/DET
        GJINV(2,1) = -GJ(2,1)/DET
C
C Compute the derivatives of the interpolation functions with
C respect to the global coordinates (x,y): [GDSF]
C
        DO 50 I = 1,2
        DO 50 J = 1,NPE
        GDSF(I,J) = 0.0
        DO 50 K = 1, 2
50    GDSF(I,J) = GDSF(I,J) + GJINV(I,K)*DSF(K,J)
        RETURN
        END

```

SUBROUTINE TEMPORAL(NN)

```

C
C
C The subroutine computes the algebraic equations associated with
C the parabolic differential equations by using the alfa-family of
C approximations. A constant source is assumed.
C
C
        IMPLICIT REAL*8(A-H,O-Z)
        COMMON/STF/ELF(18),ELK(18,18),ELM(18,18),ELXY(9,2),ELU(18),A1,A2
C
C The alfa-family of time approximation for parabolic equations
C
        DO 20 I=1,NN
        SUM=0.0
        DO 10 J=1,NN
        SUM =SUM+(ELM(I,J)-A2*ELK(I,J))*ELU(J)
10    ELK(I,J)=ELM(I,J)+A1*ELK(I,J)
20    ELF(I) =(A1+A2)*ELF(I)+SUM
        RETURN
        END

```

Appendix B

Solution of Linear Equations

B.1 Introduction

All of the steady-state iteration methods and most of the time-dependent solution methods described throughout the text lead ultimately to the point where at least one set (matrix) of linear algebraic equations must be solved. In general terms, the system

$$\tilde{\mathbf{K}}\mathbf{U}^* = \tilde{\mathbf{F}} \quad (\text{B.1.1})$$

must be solved where $\tilde{\mathbf{K}}$ is a banded, sparse matrix that may be either symmetric or unsymmetric depending on the characteristics of the partial differential equation describing the physical problem and perhaps the nonlinear solution algorithm. The solution vector (nodal point variables) \mathbf{U}^* may contain several degrees of freedom per node or a single variable, again depending on the specific problem. Other characteristics for $\tilde{\mathbf{K}}$, such as whether or not it is positive definite, and the size of its condition number, are also very dependent on the type of finite element model and the particular solution algorithm. However, these characteristics are extremely important, since they strongly influence the type of matrix solution procedure that can be successfully used for any given problem.

A linear set of equations can be solved by either a direct or iterative method. Direct solvers, like the Gauss elimination method, are often used to solve systems of algebraic equations. They provide the solution after a fixed number of steps and are less sensitive to the conditioning of the matrix. However, the main deficiency of the direct solvers is that they require the coefficient matrix be stored in an ordered format to enhance the matrix band structure and reduce storage requirements. In recent years, the direct solvers have been refined to reduce this deficiency through innovative data management techniques (e.g., frontal solvers, skyline solvers, and others; see Carey and Oden [1]). These improvements enable users to solve moderately large systems of equations efficiently. However, they have been found to be unsuitable for solving very large systems of equations (especially in three-dimensional problems) because they demand out-of-core storage of the equations and hence require large data transfers. In addition, direct methods are difficult to organize for efficient use on multiprocessor, parallel computers and are therefore seeing reduced utilization.

The limitations on CPU time and storage requirements make the use of direct solvers not economical, even on present-day high speed and high memory computers, and using direct solvers for complex problems with more than a quarter million equations is impractical. For large systems, iterative methods are more efficient in that they require less storage and CPU time while giving comparable accuracy in the solution. This is due to the fact that the global matrix formation may be avoided, and the major operation is the matrix-vector multiplication as compared to the matrix reduction in direct methods. Another advantage of the various iterative methods is that the solution algorithm can be effectively performed in parallel on an array of processors. Iterative methods are well suited to many of the linear algebraic equation sets generated by finite element models. However, the methodology is not universal in its application. The matrix problems arising from constrained finite element models, such as incompressible flow or quasi-static electromagnetics, present significant challenges to iterative methods with the major failing being the lack of a suitable preconditioner.

In the following sections a brief outline of some methods for the solution of linear algebraic equations will be given. Further details for this topic, which is outside the major focus of this book, can be found in references such as [2–4].

B.2 Direct Methods

By *direct methods*, we mean those in which simultaneous linear algebraic equations are solved exactly, assuming negligible computational round-off error, by successive elimination of variables and back substitution. The Gauss elimination method is a direct technique [2–4], and frontal [5] and skyline [6] solution methods are examples of direct solution methods that use the Gauss elimination technique efficiently. Direct solution methods involve a fixed number of steps to determine the solution. The direct techniques are useful when the number of equations involved is not too large. The number of operations (i.e., multiplication, division, etc.) for Gauss elimination is of the order $n^3/3 + O(n^2)$, where n denotes the number of operations.

For a comparable direct method, we refer to the *frontal* solution procedure [5]. Because it is faster than most direct solvers, it requires less core space as long as active variables can be kept in the core, and it allows for partial pivoting. An additional advantage is that no stringent node numbering scheme is needed, though a judicious element numbering helps to minimize the front width. Details of the frontal method will not be given here as it is a sophisticated implementation of the simple Gauss elimination method which is adequately studied in numerous texts. Implementation issues for the frontal method can be found in [5].

Due to the fact that the approximation functions are defined only within an element, the coefficient matrix in the finite element method is banded, i.e., $K_{ij} = 0$ for $j > i + m_b$, where m_b is the half bandwidth of the matrix $[K]$. This greatly reduces the number of operations, if we make note of the fact that elements outside the bandwidth are zero. Of course, the bandwidth size depends on the global node numbering. The skyline technique is one in which bandedness of the finite element equations is exploited by storing the row number m_j of the first nonzero element in column j . The variables $m_i, i = 1, 2, \dots, n$, define the skyline of the matrix. For additional details see Bathe [6].

B.3 Iterative Methods

B.3.1 General Comments

Among the various iterative methods, the Conjugate Gradient (CG) method [7] is most widely used because it is a finite step method (i.e., apart from round-off errors, the solution is achieved in a fixed number of iterations) and it can be used to determine the inverse. However, the number of iterations required depends on the condition number of the coefficient matrix (the condition number of the coefficient matrix is the ratio of the largest to smallest eigenvalue). The convergence of the conjugate gradient method, and iterative methods in general, can be improved by preconditioning and/or scaling the equations [8–10].

The limitations on storage can be overcome by solving the equations at the element level, i.e., using the Gauss–Seidel iteration idea for the set of variables associated with the element. This approach avoids assembly of element matrices to form the global coefficient matrix. This idea of using the element-by-element data structure of the coefficient matrix was first pointed out by Fox and Stanton [11] and Fried [12–14]. The phrase *element-by-element* refers to a particular data structure for finite element techniques wherein information is stored and maintained at the element level rather than assembled into a global data structure. In this method the matrix–vector multiplications are carried out at the element level and the assembly is carried out on the resultant vector. This idea proves to be very attractive when solving large problems, because the matrix–vector multiplication can be done in parallel on a series of processors. Another advantage of this method is that the resultant savings in storage, compared to direct solvers, allows solution of large problems on small computers. A review of the literature on element-by-element algorithms is presented in [15], and they have been investigated by numerous investigators [16–34].

In summary, for iterative solution methods the advantages of the element-by-element data structure over assembling the global coefficient matrix are

1. the need for formation and storage of a global matrix is eliminated, and therefore the total storage and computational costs are low,
2. the amount of storage is independent of the node numbering and mesh topology and depends on the number and type of elements in the mesh, and
3. the element-by-element solution algorithms can be vectorized for efficient use on supercomputers.

The major disadvantage of the element-by-element data structure is the limited number of preconditioners that can be formulated from the unassembled matrices. This becomes of critical importance when the linear system is not well-conditioned as in the mixed method, incompressible flow model.

B.3.2 Solution Algorithms

In this section, we review three iterative solvers from [15] that are applicable to nonsymmetric, positive definite equation systems that are typical of isothermal flow algorithms. The three iterative solution schemes used here are the Biorthogonal Conjugate Gradient method [9], the Lanczos ORTHORES [9], and the GMRES [31].

The conjugate gradient method for solving a system of equations can be interpreted as the search for the minimum of the energy E of the system. The energy

of the system is a minimum when the residual vector $\mathbf{r} = \tilde{\mathbf{F}} - \tilde{\mathbf{K}}\mathbf{U}^*$ vanishes. The algorithm for the biorthogonal conjugate gradient method (also known as two-term form of the steepest descent method, Lanczos/ORTHOMIN) for unsymmetric systems of equations [9,15] is given in Table B.3.1, and the steps involved in the Lanczos ORTHORES solution algorithm [9,15] are given in Table B.3.2.

Table B.3.1: Steps involved in using biorthogonal conjugate gradient method (Lanczos/ORTHOMIN solver).

Repeat the following steps for each nonlinear iteration:

I. *Initial Calculations*

- (1) Form the element stiffness matrix $\tilde{\mathbf{K}}^e$ and force vector $\tilde{\mathbf{F}}^e$.
- (2) Apply essential and/or natural boundary conditions, and modify $\tilde{\mathbf{K}}^e$ and $\tilde{\mathbf{F}}^e$.
- (3) Store the element matrices in $\bar{\mathbf{A}}$ (whose dimensions are nem, neleq, neleq).†
- (4) Store the inverse of the diagonal terms of the global system in \mathbf{W} ($W_{ii} = \sum_{e=1}^{\text{nem}} \tilde{K}_{ii}^{-1}$).
- (5) Assemble the global force vector.

II. *Preconditioning*

Form the preconditioned system of equations

$$\bar{\mathbf{K}}\bar{\mathbf{U}} = \bar{\mathbf{F}}; \quad \bar{\mathbf{K}} = \mathbf{W}^{-1/2}\tilde{\mathbf{K}}\mathbf{W}^{-1/2}, \quad \bar{\mathbf{U}} = \mathbf{W}^{1/2}\mathbf{U}^*, \quad \bar{\mathbf{F}} = \mathbf{W}^{-1/2}\tilde{\mathbf{F}}$$

III. *Lanczos ORTHOMIN Algorithm*

- (1) For known initial solution vector $\bar{\mathbf{U}}^0$, compute:

$$\begin{aligned} \mathbf{r}^0 &= \bar{\mathbf{F}} - \bar{\mathbf{K}}\bar{\mathbf{U}}^0, \quad \mathbf{P}^0 = \mathbf{r}^0, \quad \tilde{\mathbf{r}}^0 = \tilde{\mathbf{P}}^0 = \mathbf{r}^0, \\ \alpha_0 &= 0, \quad \lambda_0 = \frac{(\mathbf{r}^0, \tilde{\mathbf{r}}^0)}{(\bar{\mathbf{K}}\mathbf{P}^0, \tilde{\mathbf{r}}^0)}, \quad \bar{\mathbf{U}}^1 = \bar{\mathbf{U}}^0 + \lambda_0 \mathbf{P}^0 \end{aligned}$$

- (2) For each ORTHOMIN iteration $m = 1, 2, 3, \dots$, compute[§]:

$$\begin{aligned} \lambda_m &= \frac{(\mathbf{r}^m, \tilde{\mathbf{r}}^m)}{(\bar{\mathbf{K}}\mathbf{P}^m, \tilde{\mathbf{r}}^m)}, \quad \alpha_m = \frac{(\mathbf{r}^m, \tilde{\mathbf{r}}^m)}{(\mathbf{r}^{m-1}, \tilde{\mathbf{r}}^{m-1})}, \\ \mathbf{P}^m &= \mathbf{r}^m + \alpha_m \mathbf{P}^{m-1}, \quad \tilde{\mathbf{P}}^m = \tilde{\mathbf{r}} + \alpha_m \tilde{\mathbf{P}}^{m-1}, \\ \mathbf{r}^{m+1} &= \mathbf{r}^m - \lambda_m \bar{\mathbf{K}}\mathbf{P}^m, \quad \tilde{\mathbf{r}}^{m+1} = \tilde{\mathbf{r}}^m - \lambda_m \bar{\mathbf{K}}^T \tilde{\mathbf{P}}^{m-1}, \\ \bar{\mathbf{U}}^{m+1} &= \bar{\mathbf{U}}^m + \lambda_m \mathbf{P}^m \end{aligned}$$

- (3) Convergence criterion: $\|\bar{\mathbf{U}}^{m+1}\|/\|\mathbf{r}^0\| \leq 10^{-6}$
 - (4) If convergence criterion is satisfied $\bar{\mathbf{U}}^* = \mathbf{W}^{-1/2}\bar{\mathbf{U}}^{m+1}$
-

† nem = number of elements in the finite element mesh, neleq = number of element equations.

§ $(a, b) = \sum a_i b_i$.

The third iterative solver uses the GMRES solution algorithm. For an approximate solution of the form $\bar{\mathbf{U}}_0 + \mathbf{z}$, where $\bar{\mathbf{U}}_0$ is the initial guess vector and \mathbf{z} is a member of the Krylov space \mathcal{K} of dimension k , the GMRES algorithm determines the vector \mathbf{z} such that $\| \tilde{\mathbf{F}} - \tilde{\mathbf{K}}(\bar{\mathbf{U}}_0 + \mathbf{z}) \|$ is minimized, where $\| \cdot \|$ denotes the L_2 -norm. The Krylov space is given by $\mathcal{K} = \text{span}\{\bar{\mathbf{U}}_0, \tilde{\mathbf{K}}\bar{\mathbf{U}}_0, \tilde{\mathbf{K}}^2\bar{\mathbf{U}}_0, \dots, \tilde{\mathbf{K}}^{k-1}\bar{\mathbf{U}}_0\}$. Therefore, when solving large systems of equations, as the value of k increases, the amount of storage required also increases. This drawback can be overcome by employing the GMRES algorithm iteratively by using a smaller value for k and restarting the algorithm after every k steps. The restart version of the GMRES algorithm [31,15] is explained in Table B.3.3.

Table B.3.2: Steps involved in using Lanczos/ORTHORES solver.

Repeat the following steps for each nonlinear iteration:

I. Initial Calculations

- (1) Form the element stiffness matrix $\tilde{\mathbf{K}}^e$ and force vector $\tilde{\mathbf{F}}^e$.
- (2) Apply the boundary conditions, and modify $\tilde{\mathbf{K}}^e$ and $\tilde{\mathbf{F}}^e$.
- (3) Store the element matrices in $\bar{\mathbf{A}}$ (whose dimensions are nem, neleq, neleq).
- (4) Store the inverse of the diagonal terms of the global system in \mathbf{W} ($W_{ii} = \sum_{e=1}^{\text{nem}} \tilde{K}_{ii}^{-1}$).
- (5) Assemble the global force vector.

II. Preconditioning

Form the preconditioned system of equations

$$\bar{\mathbf{K}}\bar{\mathbf{U}} = \bar{\mathbf{F}}, \quad \bar{\mathbf{K}} = \mathbf{W}^{-1/2}\tilde{\mathbf{K}}\mathbf{W}^{-1/2}, \quad \bar{\mathbf{U}} = \mathbf{W}^{1/2}\tilde{\mathbf{U}}, \quad \bar{\mathbf{F}} = \mathbf{W}^{-1/2}\tilde{\mathbf{F}}$$

III. Lanczos ORTHORES Algorithm

- (1) For known initial solution vector $\bar{\mathbf{U}}^0$, compute: $\mathbf{r}^0 = \bar{\mathbf{F}} - \bar{\mathbf{K}}\bar{\mathbf{U}}^0$
- (2) For each ORTHORES iteration $m = 0, 1, 2, \dots$, compute ($\tilde{\mathbf{r}} = \mathbf{r}_0, \lambda^0 = 0$):

$$\begin{aligned} \lambda^{m+1} &= \frac{(\mathbf{r}^m, \tilde{\mathbf{r}}^m)}{(\bar{\mathbf{K}}\mathbf{r}^m, \tilde{\mathbf{r}}^m)} \\ \beta^{m+1} &= \begin{cases} 1 & ; \text{ if } m = 0 \\ \left[1 - \frac{\lambda^{m+1}}{\lambda^m} \frac{(\mathbf{r}^m, \tilde{\mathbf{r}}^m)}{(\mathbf{r}^{m-1}, \tilde{\mathbf{r}}^{m-1})} \frac{1}{\beta_m} \right]^{-1} & ; \text{ if } m \geq 1 \end{cases} \\ \mathbf{r}^{m+1} &= \beta^{m+1} \left(\mathbf{r}^m - \lambda^{m+1} \bar{\mathbf{K}}\mathbf{r}^m \right) + (1 - \beta^{m+1}) \mathbf{r}^{m-1} \\ \tilde{\mathbf{r}}^{m+1} &= \beta^{m+1} \left(\tilde{\mathbf{r}}^m - \lambda^{m+1} \bar{\mathbf{K}}\tilde{\mathbf{r}}^m \right) + (1 - \beta^{m+1}) \tilde{\mathbf{r}}^{m-1} \\ \bar{\mathbf{U}}^{m+1} &= \beta^{m+1} \left(\bar{\mathbf{U}}^m + \lambda^{m+1} \mathbf{r}^m \right) + (1 - \beta^{m+1}) \bar{\mathbf{U}}^{m-1} \end{aligned}$$

- (3) Convergence criterion: $\|\bar{\mathbf{U}}^{m+1}\|/\|\mathbf{r}^0\| \leq 10^{-6}$
 - (4) If convergence criterion is satisfied: $\mathbf{U}^* = \mathbf{W}^{-1/2}\bar{\mathbf{U}}^{m+1}$
-

Table B.3.3: Steps involved in using the GMRES solver.

Repeat the following steps for each nonlinear iteration:

I. *Initial Calculations*

- (1) Form the element stiffness matrix $\tilde{\mathbf{K}}^e$ and force vector $\tilde{\mathbf{F}}^e$.
- (2) Apply the boundary conditions, and modify $\tilde{\mathbf{K}}^e$ and $\tilde{\mathbf{F}}^e$.
- (3) Store the element matrices in $\bar{\mathbf{A}}$ (whose dimensions are nem, neleq, neleq).
- (4) Store the inverse of the diagonal terms of the global system in \mathbf{W}
 $(W_{ii} = \sum_{e=1}^{nem} \hat{K}_{ii}^{-1})$.
- (5) Assemble the global force vector.

II. *Preconditioning* Form the preconditioned system of equations

$$\bar{\mathbf{K}}\bar{\mathbf{U}} = \bar{\mathbf{F}}, \quad \bar{\mathbf{K}} = \mathbf{W}^{-1/2}\tilde{\mathbf{K}}\mathbf{W}^{-1/2}, \quad \bar{\mathbf{U}} = \mathbf{W}^{1/2}\tilde{\mathbf{U}}, \quad \bar{\mathbf{F}} = \mathbf{W}^{-1/2}\tilde{\mathbf{F}}$$

III. *GMRES Algorithm*

- (1) *Start* Choose $\bar{\mathbf{U}}^0$ and compute $\mathbf{r}^0 = \bar{\mathbf{F}} - \bar{\mathbf{K}}\bar{\mathbf{U}}^0$, and $\mathbf{v}^1 = \bar{\mathbf{U}}^0 / \| \bar{\mathbf{U}}^0 \|$
- (2) *Iterate* For $j = 1, 2, \dots, k$, do: $h_{i,j} = (\bar{\mathbf{K}}v_j, v_i)$, $i = 1, 2, \dots, j$, $\hat{v}_{j+1} = \bar{\mathbf{K}}v_j - \sum_{i=1}^j h_{i,j}v_i$, $h_{j+1,j} = \| \hat{v}_{j+1} \|$, and $v_{j+1} = \hat{v}_{j+1}/h_{j+1,j}$
- (3) *Form approximate solution* $\bar{\mathbf{U}}^k = \bar{\mathbf{U}}^0 + \mathbf{V}\mathbf{y}$, where \mathbf{y} minimizes $\| \mathbf{e} - \bar{\mathbf{H}}\mathbf{y} \|$, $\mathbf{y} \in \mathbf{R}^k$.
- (4) *Restart* Compute $\mathbf{r}^k = \bar{\mathbf{F}} - \bar{\mathbf{K}}\bar{\mathbf{U}}^k$; check convergence; if satisfied stop; otherwise, compute $\bar{\mathbf{U}}^0 := \bar{\mathbf{U}}^k$, $\mathbf{v}_1 := \bar{\mathbf{U}}^k / \| \bar{\mathbf{U}}^k \|$, and go to step 2.
- (5) *Convergence criterion* $\| \bar{\mathbf{U}}^{m+1} \| / \| \mathbf{r}^0 \| \leq 10^{-6}$
- (6) *If convergence criterion is satisfied* $\mathbf{U}^* = \mathbf{W}^{-1/2}\bar{\mathbf{U}}$

where \mathbf{V} is a $N \times k$ matrix whose columns are 1–2 orthonormal basis vectors $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k\}$, $\mathbf{e} = \{\| \bar{\mathbf{U}}^0 \|, 0, \dots, 0\}$, and $\bar{\mathbf{H}}$ is the upper $k \times k$ Hessenberg matrix whose entries are the scalars $h_{i,j}$. When using the restart version of the GMRES algorithm, the total number of iteration m can be computed from the number of restarts and the dimension of k .

The presence of a penalty matrix in the global coefficient matrix of the penalty model spoils the condition number. This results in slow convergence when using iterative solvers. However, the convergence of the iterative solvers can be improved by preconditioning the system. In [15], the system of equations is transformed using diagonal scaling matrix (Jacobi/diagonal preconditioning). Accordingly, the system of equations

$$\tilde{\mathbf{K}}\mathbf{U}^* = \tilde{\mathbf{F}} \tag{B.3.1}$$

becomes

$$\bar{\mathbf{K}}\bar{\mathbf{U}} = \bar{\mathbf{F}} \tag{B.3.2}$$

$$\bar{\mathbf{K}} = \mathbf{W}^{-1/2}\tilde{\mathbf{K}}\mathbf{W}^{-1/2}; \quad \bar{\mathbf{U}} = \mathbf{W}^{1/2}\mathbf{U}^*; \quad \bar{\mathbf{F}} = \mathbf{W}^{-1/2}\tilde{\mathbf{F}} \tag{B.3.3}$$

where $\mathbf{W}_{ii} = \tilde{\mathbf{K}}_{ii}^{-1}$ is a diagonal matrix. During the matrix multiplication, the element-by-element data structure is exploited and the multiplications are carried out at the element level and the residuals are then assembled to form the global vector. The diagonal terms are always positive because of the viscous and penalty terms.

References for Additional Reading

1. G. F. Carey and J. T. Oden, *Finite Elements: Computational Aspects*, Prentice Hall, Englewood Cliffs, New Jersey (1984).
2. K. E. Atkinson, *An Introduction to Numerical Analysis*, Wiley, New York (1978).
3. B. Carnahan, H. A. Luther, and J. O. Wilkes, *Applied Numerical Methods*, Wiley, New York (1969).
4. D. K. Fadeev and V. N. Fadeeva, *Computational Methods of Linear Algebra*, Freeman, San Francisco, California (1963).
5. P. Hood, “Frontal Solution Program for Unsymmetric Matrices,” *International Journal for Numerical Methods in Engineering*, **10**, 379–399 (1976); also see **10**, 1055 (1976) for a correction.
6. K. J. Bathe, *Finite Element Procedures*, Prentice Hall, Englewood Cliffs, New Jersey (1998).
7. M. R. Hestenes and E. L. Stiefel, “Methods of Conjugate Gradients for Solving Linear Systems,” *National Bureau of Standards Journal of Research*, **49**, 409–436 (1952).
8. G. H. Golub and C. F. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, Maryland (1989).
9. K. C. Jea and D. M. Young, “On the Simplification of Generalized Conjugate-Gradient Methods for Nonsymmetrizable Linear Systems,” *Linear Algebra Applications*, **52**, 399–417 (1983).
10. D. J. Evans, “Use of Preconditioning in Iterative Methods for Solving Linear Equations with Symmetric Positive Definite Matrices,” *Computer Journal*, **4**, 73–78 (1961).
11. R. L. Fox and E. L. Stanton, “Developments in Structural Analysis by Direct Energy Minimization,” *AIAA Journal*, **6**, 1036–1042 (1968).
12. I. Fried, “Gradient Methods for Finite Element Eigenproblems,” *AIAA Journal*, **7**, 739–741 (1969).
13. I. Fried, “More on Generalized Iterative Methods in Finite Element Analysis,” *AIAA Journal*, **7**, 565–567 (1969).
14. I. Fried, “A Gradient Computational Procedure for the Solution of Large Problems Arising from the Finite Element Discretization Method,” *International Journal for Numerical Methods in Engineering*, **2**, 477–494 (1970).
15. M. P. Reddy, J. N. Reddy and H. U. Akay, “Penalty Finite Element Analysis of Incompressible Flows Using Element by Element Solution Algorithms,” *Computer Methods in Applied Mechanics and Engineering*, **100**, 169–205 (1992).
16. A. J. Wathen, “An Analysis of Some Element-By-Element Techniques,” *Computer Methods in Applied Mechanics and Engineering*, **74**, 271–287 (1989).

17. T. J. R. Hughes, I. Levit, and J. Winget, "An Element-By-Element Implicit Algorithm for Heat Conduction," *Journal of the Engineering Mechanics Division, ASCE*, **109**(2), 576–585 (1983).
18. J. M. Winget and T. J. R. Hughes, "Solution Algorithms for Nonlinear Transient Heat Conduction Analysis Employing Element-By-Element Iterative Strategies," *Computer Methods in Applied Mechanics and Engineering*, **52**, 711–815 (1985).
19. G. F. Carey and B. Jiang, "Element-By-Element Linear and Nonlinear Solution Schemes," *Communications in Applied Numerical Methods*, **2**, 145–153 (1986).
20. J. L. Hayes and P. Devloo, "An Element-By-Element Block Iterative Method for Large Non-Linear Problems," in *Innovative Methods for Nonlinear Behavior*, W. K. Liu et al. (eds.), Pineridge Press, Swansea, U.K., 51–62 (1985).
21. E. K. Buratynski, "An Element-By-Element Method for Heat Conduction CAE Including Composite Problems," *International Journal for Numerical Methods in Engineering*, **26**, 199–215 (1988).
22. I. Levit, "Element By Element Solvers of Order N," *Computers & Structures*, **27**(3), 357–360 (1987).
23. A. de La Bourdinaye, "The Element By Element Method as a Preconditioner for Linear Systems Coming from Finite Element Models," *International Journal of Supercomputer Applications*, **3**, 60–68 (1989).
24. K. V. G. Prakhya, "Some Conjugate Gradient Methods for Symmetric and Nonsymmetric Systems," *Communications in Applied Numerical Methods*, **4**, 531–539 (1988).
25. G. F. Carey, E. Barragy, R. McLay, and M. Sharma, "Element-by-Element Vector and Parallel Computations," *Communications in Applied Numerical Methods*, **4**, 299–307 (1988).
26. L. J. Hayes and P. Devloo, "A Vectorized Version of a Sparse Matrix-Vector Multiplication," *International Journal for Numerical Methods in Engineering*, **23**, 1043–1056 (1986).
27. F. Shakib, T. J. R. Hughes, and Z. Johan, "A Multi-Element Group Preconditioned GMRES Algorithm for Nonsymmetric Systems Arising in Finite Element Analysis," *Computer Methods in Applied Mechanics and Engineering*, **75**, 415–456 (1989).
28. T. E. Tezduyar and J. Liou, "Element-by-Element and Implicit-Explicit Finite Element Formulations for Computational Fluid Dynamics," *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, R. Glowinski, G. H. Golub, G. A. Meurant, and J. Periaux (eds.), SIAM, Philadelphia, Pennsylvania, 281–300 (1988).
29. T. E. Tezduyar and J. Liou, "Grouped Element-By-Element Iteration Schemes for Incompressible Flow Computations," *Computational Physics Communications*, **53**, 441–453 (1989).
30. J. Liou and T. E. Tezduyar, "A Clustered Element-By-Element Iteration Method for Finite Element Computations," University of Minnesota Supercomputer Institute Research Report, 90/116 (1990).
31. Y. Saad and M. H. Schultz, "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal of Scientific and Statistical Computations*, **7**(3), 856–869 (1986).
32. J. A. Mitchell and J. N. Reddy, "A Multilevel Hierarchical Preconditioner for Thin Elastic Solids," *International Journal for Numerical Methods in Engineering*, **43**, 1383–1400 (1998).
33. J. A. Mitchell and J. N. Reddy, "A High Performance Iterative Solution Procedure for the Analysis of Structural Problems," *Journal of High Performance Computing*, **5**(1), 3–13 (1999).
34. J. A. Mitchell and J. N. Reddy, "A Hierarchical Iterative Procedure for the Analysis of Composite Laminates," *Computer Methods in Applied Mechanics and Engineering*, **181**, 237–260 (2000).

Appendix C

Fixed Point Methods and Contraction Mappings

C.1 Fixed Point Theorem

The discussion in this section will focus on the use of fixed point iteration schemes for the solution of the system of nonlinear equations of the form

$$\mathcal{R}(\mathbf{x}) = \mathbf{0} \quad (\text{C.1.1})$$

or its fixed point form

$$\mathbf{x} = \mathcal{G}(\mathbf{x}) \equiv \mathbf{x} - \mathcal{R}(\mathbf{x}) \quad (\text{C.1.2})$$

where \mathbf{x} represents a vector of unknowns. Solutions of Eq. (C.1.2) are called *fixed points* of the mapping \mathcal{G} , because they are unchanged by the operation of \mathcal{G} . The form of (C.1.2) immediately suggests the following iteration scheme [1]:

$$\mathbf{x}^{n+1} = \mathcal{G}(\mathbf{x}^n) \quad (\text{C.1.3})$$

where $n = 0, 1, \dots$. Equation (C.1.3) is the method of successive substitutions known as the Picard iteration method. The convergence of the method in (C.1.3) is guaranteed by the *Banach Fixed Point Theorem*, which gives the sufficient conditions for the existence and uniqueness of solutions (see Reddy [1], pp. 202–208). Before we present the theorem, it is necessary to define a contraction mapping.

An operator \mathcal{G} is termed a *contraction mapping* if there exists a positive number λ , $0 < \lambda < 1$, such that

$$\| \mathcal{G}(\mathbf{x}) - \mathcal{G}(\mathbf{y}) \| \leq \lambda \| \mathbf{x} - \mathbf{y} \| \quad (\text{C.1.4})$$

holds for all \mathbf{x}, \mathbf{y} in the closed ball $\mathcal{N} = \{ \mathbf{x} : \| \mathbf{x} - \mathbf{x}_0 \| \leq r \}$ of a normed space \mathcal{V} , with center at \mathbf{x}_0 and radius r . Here λ denotes the contraction factor. The notation $\| \cdot \|$ is an appropriate norm for the operator in \mathcal{N} (see [1]). It is clear from Eq. (C.1.4) that a contraction mapping “contracts distances”: the distance between the images $\mathcal{G}(\mathbf{x})$ and $\mathcal{G}(\mathbf{y})$ is smaller by a scale factor of λ than the distance between the elements \mathbf{x} and \mathbf{y} .

Theorem C.1: Let $\mathcal{G}(\mathbf{x})$ be a contraction mapping on a closed subset \mathcal{N} of a closed normed space (or a Banach space) \mathcal{V} , with a contraction factor λ and let \mathbf{x}_0 be such that

$$\frac{1}{1 - \lambda} \| \mathcal{G}(\mathbf{x}_0) - \mathbf{x}_0 \| = r_0 \leq r$$

Then

1. The sequence defined by $\mathbf{x}^{n+1} = \mathcal{G}(\mathbf{x}^n)$ converges to a point \mathbf{x}^* in \mathcal{N} .
2. \mathbf{x}^* is a fixed point of the operator \mathcal{G} .
3. \mathbf{x}^* is the unique fixed point of \mathcal{G} in \mathcal{N} .

The proof of this theorem and a detailed background on nonlinear solution methods are available in [1,2].

Based on the above theorem, a further result can be obtained if a few restrictions are placed on \mathcal{G} . Assume that (C.1.3) has a solution \mathbf{x}^* and that \mathcal{G} has a continuous Jacobian that satisfies

$$\| \mathcal{G}'(\mathbf{x}) \|_{\infty} \leq \lambda < 1 \quad (\text{C.1.5})$$

where $\| \cdot \|_{\infty}$ denotes the *sup-norm* (“sup” stands for supremum). Then it can be shown that for any initial estimate \mathbf{x}^0 satisfying

$$\| \mathbf{x}^0 - \mathbf{x}^* \|_{\infty} \leq r \quad (\text{C.1.6})$$

the iterative scheme in Eq. (C.1.3) will converge to the unique solution \mathbf{x}^* . The conditions (C.1.5) and (C.1.6) basically state that the norm of the Jacobian of the iteration operator must be less than unity and the initial estimate for the solution must be within the ball of contraction, if the fixed point scheme is to converge. The quantity r is also referred to as the radius of convergence of the iteration scheme.

Generally, it is quite difficult to prove that a given operator \mathcal{G} is a contraction mapping, especially for the systems generated by finite element applications. It is somewhat easier to use (C.1.5) and estimate the norm of \mathcal{G}' to test for a convergent process. However, the real benefit from the ideas of fixed point operators is derived from the iteration scheme in (C.1.3) and its generalization,

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \alpha \mathbf{A}(\mathbf{x}^n) \mathcal{F}(\mathbf{x}^n) \quad (\text{C.1.7})$$

where α is a constant and \mathbf{A} is a nonsingular matrix function of \mathbf{x} . In the following, a few of the basic iteration schemes will be defined based on the definition of \mathbf{A} .

C.2 Chord Method

The simplest choice for \mathbf{A} is a constant which defines the parallel chord method. For a one degree-of-freedom problem, this becomes

$$x^{n+1} = x^n - af(x^n) \quad (\text{C.2.1})$$

which is shown graphically in Figure C.2.1. In the multi-dimensional case, the line with slope $\frac{1}{a}$ is replaced by a hyperplane. The behavior of this method depends strongly on the form of \mathbf{A} . One possibility is to set $\mathbf{A} = \alpha \mathbf{I}$ (\mathbf{I} is the identity matrix) in which case each component of \mathbf{x} is corrected in proportion to α . If $\mathbf{A} = \alpha^T \mathbf{I}$ where α is a vector, then each component of \mathbf{x} can be corrected individually. The problem remains, however, as to how to select α .

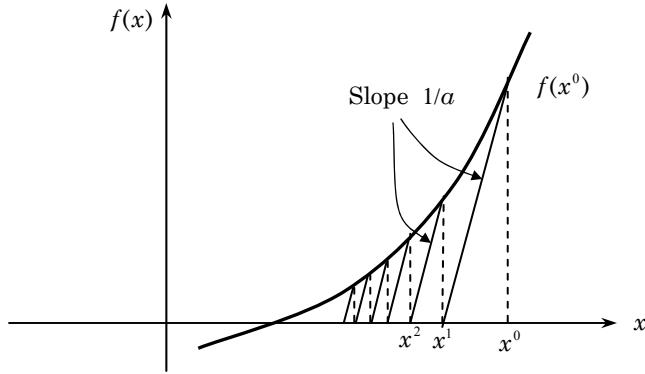


Figure C.2.1: Illustration of the chord method for one degree-of-freedom problem.

C.3 Newton's Method

Extending the idea that \mathbf{A} is in some sense an inverse “slope,” it is appropriate to relate \mathbf{A} to the derivative of \mathcal{F} . Therefore, let

$$\mathbf{A} = \left[\frac{\partial \mathcal{F}}{\partial \mathbf{x}} \right]^{-1} = \mathbf{J}^{-1} \quad (\text{C.3.1})$$

in which case Eq. (C.1.7) becomes

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \alpha \mathbf{J}^{-1}(\mathbf{x}^0) \mathcal{F}(\mathbf{x}^n) \quad (\text{C.3.2})$$

where the Jacobian is evaluated at \mathbf{x}^0 . For the one degree-of-freedom case, this becomes ($\alpha = 1$)

$$x^{n+1} = x^n - \frac{1}{f'(x^0)} f(x^n) \quad (\text{C.3.3})$$

which is illustrated in Figure C.3.1. This is recognized as a modified Newton's method since the Jacobian is not updated at each iteration.

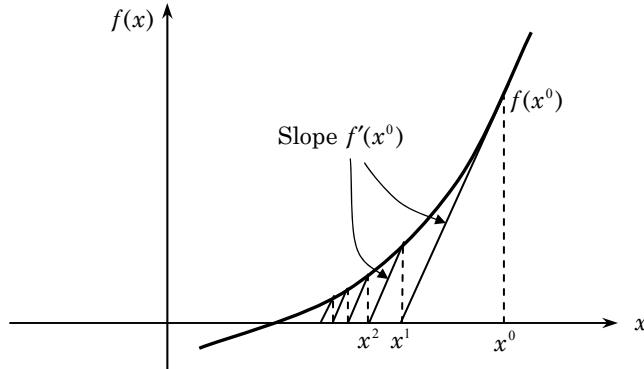


Figure C.3.1: Illustration of modified Newton's method.

C.4 The Newton–Raphson Method

The modified Newton's scheme suggests a rational manner in which to select \mathbf{A} . However, its convergence is little better than the chord method due to the use of a constant Jacobian. Allowing the Jacobian to change at each iteration produces the standard Newton (or Newton–Raphson) procedure

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \alpha \mathbf{J}^{-1}(\mathbf{x}^n) \mathcal{F}(\mathbf{x}^n) \quad (\text{C.4.1})$$

This is shown graphically in Figure C.4.1 for a one-component system. Note that if $\alpha \neq 1$ a damped Newton scheme is produced.

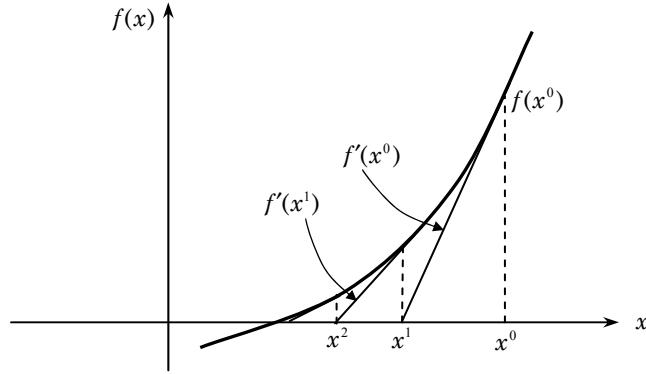


Figure C.4.1: Illustration of Newton's method.

C.5 Descent Methods

Descent methods reinterpret the general algorithm in (C.1.7) as

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \alpha \mathbf{D}^n \quad (\text{C.5.1})$$

where \mathbf{D} is a vector that determines the direction of the correction and α controls the magnitude of the correction. In addition, rather than directly finding the solution to (C.1.1), descent methods work by trying to minimize the function

$$R(\mathbf{x}) = \mathcal{F}^T(\mathbf{x}) \mathcal{F}(\mathbf{x}) \quad (\text{C.5.2})$$

The simplest descent methods are the univariant or relaxation schemes in which \mathbf{D} is selected so that only one component of \mathbf{x} is corrected at each iteration. A better approach involves the gradients of R , such as the method of steepest descent. In this case

$$\mathbf{D}^n = \mathbf{g}^n = \nabla R(\mathbf{x}^n) \quad (\text{C.5.3})$$

The method of steepest descent has a linear rate of convergence.

Finally, the conjugate gradient method is a sequential method in which the correction vectors are computed from the recursion relation

$$\mathbf{D}^n = \nabla R(\mathbf{x}^n) + \beta^{n-1} \mathbf{D}^{n-1} \quad (\text{C.5.4})$$

Here the new correction vector is a combination of the gradient at \mathbf{x}^n plus the old correction direction. The parameter β is defined to insure that the new and old correction vectors are properly orthogonal (conjugate).

Full details for all of the above algorithms and the general concepts behind nonlinear solution methods can be found in [3–6].

References for Additional Reading

1. J. N. Reddy, *Applied Functional Analysis and Variational Methods in Engineering*, McGraw-Hill, New York (1986); reprinted by Krieger Publishers, Melbourne, Florida (1991).
2. M. R. Hestenes, *Optimization Theory: The Finite Dimensional Case*, John Wiley & Sons, New York (1975).
3. E. B. Becker, G. F. Carey, and J. T. Oden, *Finite Elements, An Introduction*, Vol. I, Prentice Hall, Englewood Cliffs, New Jersey (1981).
4. J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York (1970).
5. C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, Cambridge, UK (1987).
6. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis*, Oxford University Press, Oxford, UK (2004).

The Finite Element Method in Heat Transfer and Fluid Dynamics

As computational fluid dynamics (CFD) and computational heat transfer (CHT) evolve and become increasingly important in standard engineering design and analysis practice, users require a solid understanding of mechanics and numerical methods to make optimal use of available software. **The Finite Element Method in Heat Transfer and Fluid Dynamics, Third Edition** illustrates what a user must know to ensure the optimal application of computational procedures—particularly the finite element method (FEM)—to important problems associated with heat conduction, incompressible viscous flows, and CHT.

This book follows the tradition of the bestselling previous editions, noted for their concise explanation and powerful presentation of useful methodology tailored for use in simulating CFD and CHT. The authors update research developments while retaining the previous editions' key material and popular style in regard to text organization, equation numbering, references, and symbols.

This updated third edition features new or extended coverage of

- Coupled problems and parallel processing
- Mathematical preliminaries and low-speed compressible flows
- Mode superposition methods and a more detailed account of radiation solution methods
- Variational multi-scale methods (VMM) and least-squares finite element models (LSFEM)
- Application of the finite element method to non-isothermal flows
- Formulation of low-speed, compressible flows

With its presentation of realistic, applied examples of FEM in thermal and fluid design analysis, this proven masterwork is an invaluable tool for mastering basic methodology, competently using existing simulation software, and developing simpler special-purpose computer codes. It remains one of the very best resources for understanding numerical methods used in the study of fluid mechanics and heat transfer phenomena.



CRC Press
Taylor & Francis Group
an informa business
www.crcpress.com

6000 Broken Sound Parkway, NW
Suite 300, Boca Raton, FL 33487
270 Madison Avenue
New York, NY 10016
2 Park Square, Milton Park
Abingdon, Oxon OX14 4RN, UK

85980

ISBN: 978-1-4200-8598-3
90000



9 781420 085983