To the Graduate Council:

I am submitting herewith a dissertation written by David J. Chaffin entitled "A Taylor Weak Statement Finite Element Method for Computational Fluid Dynamics." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Engineering Science and Mechanics.

A. J. Baker, Major Professor

We have read this dissertation
and recommend its acceptance:

Joe Tinsley

Jay Frankel

Terry Miller

Accepted for the Council:

Associate Vice Chancellor and
Dean of The Graduate School

A Taylor weak statement finite element method for computational fluid dynamics

A Dissertation Presented for the
Doctor of Philosophy Degree
The University of Tennessee, Knoxville

David J. Chaffin

December, 1997

Acknowledgments

The author wishes to thank his committee, Drs. Jay Frankel, Joe Iannelli, and Terry Miller, along with Drs. Kwai-Lam Wong and Subrata Roy, for many helpful suggestions. The financial and computing support of the UT Engineering Computer Center, directed by Faye Muly and Winston Holmes, is gratefully acknowledged.

Most of all, the author acknowledges the tireless efforts of major professor Dr. A. J. Baker and the constant support and encouragement of his wife, Carmen Montgomery, which were crucial to the completion of this project.

Abstract

Taylor weak statement (TWS) or Taylor-Galerkin finite element method approximations were formulated for the advection-diffusion equation, the Burgers equation, and the incompressible Navier-Stokes equations. A new TWS formulation was developed for the shallow-water equations. A Fourier analysis was performed to compare phase accuracy and stability for linear basis finite element methods; a new Fourier analysis was also performed for quadratic and cubic basis TWS one-dimensional finite element methods. A multiple-step linear basis TWS finite element method was derived and optimized for phase accuracy. A new modified equation analysis was developed in a general linear form, and the analysis was used to optimize two-dimensional TWS methods for phase accuracy and for stability. Phase-accurate TWS finite element methods were found effective in maintaining time accuracy in scalar field equations, but of limited value for the transient Navier-Stokes equation systems tested.

Stability-enhanced artificial diffusion TWS methods for the incompressible Navier-Stokes equations were systematically compared for the first time to a wide variety of modified finite element methods, including SUPG, exponential Petrov-Galerkin, SGM, hierarchical basis, least-squares, and monotone methods. The artificial diffusion methods were tested on various two-dimensional advection-diffusion and incompressible laminar Navier-Stokes benchmark problems. In contrast to the phase-accurate methods, the stability-enhanced methods were found to effectively decrease grid requirements for high Reynolds number laminar flow benchmark problems, including the backward-facing step and cylinder-in-crossflow. The best performing methods were the exponential Petrov-Galerkin and streamline-diffusion (TWS or SUPG) methods. As expected, the artificial diffusion methods were found ineffective on a high Rayleigh number natural convection benchmark problems characterized by small magnitude of velocity.

Contents

List of Tables

List of Figures

Nomenclature

$\mathcal{A}_e(\cdots)$	assembly over finite elements
A_m	Fourier expansion coefficient of analytical solution
$\hat{A}_m$	Fourier expansion coefficient of approximate solution
$\mathtt{A}[\cdots]$	1D finite element matrix without metric data
$\mathtt{B}[\cdots]$	2D finite element matrix without metric data
c	artificial diffusion function of Roy
c	artificial diffusion function of von Neumann and Richtmeyer
c	constant in error analysis
C	Courant number, $\frac{U\Delta_t}{\Delta_x}$
$\mathtt{C}[\cdots]$	3D finite element matrix without metric data
D	discrete diffusion term of Layton
D	cylinder diameter
e	subscript indicating function which varies by element
E	subscript indicating energy
f	test function of Chee and Kwon
f	coefficient of modified equation method expansion
$\mathcal{F}$	artificial diffusion function of Roy
$\mathbf{F}$	flux vector, continuous
g	prescribed flux on Γ_2, continuous
g	acceleration of gravity
G	prescribed flux on Γ_2, discrete
h	element metric
h	subscript or superscript indicating discrete
h	height variable, shallow-water equations
H^1	Hilbert space
j	subscript indicating x discretization level
j	coordinate direction subscript on u, tensor summation
J_e	finite element Jacobian with determinant $\lvert J_e \rvert$
k	subscript indicating y discretization level
k	coordinate direction subscript on u, tensor summation
k	finite element interpolation order
$\bar{k}$	SUPG artificial diffusion parameter
l	coordinate direction subscript on u, tensor summation
l	summation index on χ, upper bound M
m	summation index on ψ, upper bound M
m	mass flow variable, shallow-water equations
L	characteristic length
L^2	Sobolev space
$\mathcal{L}$	differential equation statement, continuous
$\mathcal{L}^h$	differential equation statement, discretized
$\mathcal{L}_d^h$	differential equation statement, second order part, discretized
$\hat{\mathcal{L}}^h$	differential equation statement, discretized and modified
$\mathtt{M}[\cdots]$	finite element matrix with metric data
n	superscript indicating time discretization level
n	subscript indicating normal direction
p	pressure, continuous
$\hat{p}$	pressure, continuous in space, not satisfying continuity
$\tilde{p}$	pressure, contiguous in space, approximately satisfying continuity

P	pressure, discrete
q	dependent variable, continuous
$\mathbf{q}$	dependent variable vector, continuous
q_b	prescribed variable on Γ_1
Q	dependent variable, discrete
$\mathcal{R}^n$	n-dimensional real space
s	source term, continuous
S	source term, discrete
T	discrete operator approximating $\frac{\partial}{\partial t}$
T	superscript indicating transpose
t	time
s	subscript indicating streamwise direction
$S(\cdots)$	finite difference stencil
t	tangential direction, subscript
u	velocity, continuous in space
$\hat{u}$	velocity, continuous in space, not satisfying continuity
$\tilde{u}$	velocity, continuous in space, approximately satisfying continuity
U	velocity, discrete
$\bar{U}$	velocity, discrete, average over element
$\bar{U}$	velocity, discrete, average over domain inlet, for St
w	weak statement test function
W	weak statement test function, discrete
$\mathcal{W}$	weak differential equation statement, continuous
$\mathcal{W}^h$	weak differential equation statement, discretized
$\hat{\mathcal{W}}^h$	weak differential equation statement, discretized and modified
x	coordinate direction and subscript
x_0	coordinate of beginning of element
X	discrete operator approximating $\frac{\partial}{\partial x}$
y	coordinate direction and subscript
Y	discrete operator approximating $\frac{\partial}{\partial y}$
$\Im(\cdots)$	imaginary part
$\Re(\cdots)$	real part
α	thermal diffusivity
α	Taylor weak statement correction parameter
β	Taylor weak statement correction parameter
γ	Taylor weak statement correction parameter
Γ	boundary of Ω
δ_x	finite difference centered approximation to first derivative, also subscript y
δ_x^-	finite difference one-sided approximation to first derivative, also subscript y
δ_x^2	finite difference centered approximation to second derivative, also subscript y
δ_{lm}	Kronecker delta
$\delta_{\mathtt{L}}$	deviations of elements from rectangular geometry, also subscripts $\mathtt{R}, \mathtt{B}, \mathtt{T}$
Δ_x	space discretization step, also subscript y
Δ_t	time discretization step
Δ_t^*	artificial diffusion parameter, units of time discretization step
ε	arbitrary small constant in TWS artificial diffusion method
ε	error in solution for energy norm
ε	artificial diffusion parameter of Iannelli
ζ	transformed z-coordinate

η	transformed x-coordinate
θ	implicitness
θ	angle of Layton
Θ	non-dimensional temperature
Λ	artificial diffusion function of Roy
μ	Taylor weak statement correction parameter
ν	viscosity
$\hat{\nu}$	effective viscosity
ξ	transformed y-coordinate
ρ	spectral radius
σ	phase angle in x-direction
ς	phase angle in y-direction
τ	boundary differential element
ϕ	pressure correction variable
Φ^h	relative phase velocity of an approximation
ψ	weak statement trial function
χ	weak statement test function
φ	artificial diffusion function of Iannelli
ω	Fourier frequency
Ω	space domain of weak statement
Ra	Rayleigh number $\frac{g\Delta\Theta L^3}{\nu}$
Re	Reynolds number $\frac{UL}{\nu}$
Pe	Peclet number $\frac{UL}{\alpha}$
Pr	Prandtl number $\frac{\nu}{\alpha}$
St	Strouhal number $\frac{fD}{U}$

Chapter 1

Introduction

1.1 Motivation and history

The Navier-Stokes equations appear to provide a complete differential description of fluid flow (from White [85]). However, the equation system is analytically solvable only for a very few simple flows and geometries.

Computational fluid dynamics (CFD) is the process of constructing approximate solutions to the Navier-Stokes equations. The initial development is attributed by Ames [4] to Thom in 1928 using hand calculation. Practical use awaited sufficiently powerful computers emerging in the 1960's.

Commonly used today are three methods of constructing the numerical approximation: the finite difference method (FDM), the finite volume method (FVM), and the finite element method (FEM). In the finite difference method, partial derivatives are directly replaced by Taylor expansions or divided differences expressed at points of the mesh. It is the most obvious approximation and was the first developed. The FDM has since been mostly replaced by the finite volume method, in which the differential equations are integrated over volumes. The FVM is a natural approximation for equation systems expressed as conservation laws and enables approximation of non-Cartesian geometry.

The finite element method also features a local integration of the differential equation system, but is based on the concept of error minimization or creating an optimum approximation. The FEM also allows non-Cartesian geometry and flexible handling of boundary conditions. The generality of its mathematical formulation provides error measures and encompasses finite volume methods as particular cases. Finite element methods are used in this research with comparison to finite difference and finite volume methods.

Early studies using the FDM such as Courant et al. [24] showed that a numerical parameter later known as the "cell Peclet number" $\frac{V\Delta_x}{\alpha}$ had to be limited to order one for accurate resolution. In that case the mesh size Δ_x, with $L = O(1)$, is limited to $O(\frac{1}{\mathrm{Re}})$ in one dimension, and that quantity cubed in three dimensions, the latter not easily achievable even on present-day computers. Practical CFD for design use (i.e. not using supercomputers) thus requires reasonable accuracy using fewer mesh points, and achieving this "coarse grid accuracy" in a finite element method is the primary goal of this research.

The early solution to the problem of a too-coarse mesh was to numerically increase the viscosity ν as in the "donor-cell" method of Courant et al. [24]. This had the effect of reducing the cell Reynolds number, thus increasing the accuracy and numerical stability. However, it also had the equivalent and undesirable effect of simulating a reduction in the the physical Reynolds number and thereby fundamentally changing the physical problem to be simulated. The method of increasing the viscosity became known as the "upwind" method or the "artificial viscosity" or "artificial dissipation" method.

Various modifications have since been made to upwind methods to try and retain the desirable stability property without the undesirable excess viscosity. Nonlinear algorithms such as the "flux-correction" method of Oran and Boris [60] are often used for the Euler equations, in which the stability problem is more acute since there is often an assumption of no physical viscosity. In incompressible Navier-Stokes computations, linear schemes are usually preferred for efficiency and so that standard linear matrix solvers may be used. Linear schemes such as the "power-law" method of Patankar [61] and the "QUICK" method of Leonard [57] are widely used for finite volume computations of the Navier-Stokes equations, the first generally regarded as still causing excess numerical viscosity [32].

Artificial viscosity methods for finite element computations were formulated as the "Petrov-Galerkin" method of Christie et al. [21, 22]. These were refined as "Streamline-upwind Petrov-Galerkin" or SUPG method of Brooks and Hughes [12] and other methods from Dick [26], Westerink and Shea [84], Kondo et al. [52], and Bouloutas and Celia [10]. Recent artificial viscosity finite element methods have been formulated by Iannelli [49] and Roy and Baker [69].

Lax and Wendroff [53] formulated an inherently transient algorithm which, depending on application, could be viewed as either an artificial viscosity method or as a method to increase the accuracy of a time evolution algorithm. The Lax-Wendroff method was extended for finite volumes by Stone and Brian [73] and Harten and Tal-Ezer [46]. The method was applied to finite elements by van Genuchten and Gray [77], by Donea [28] as the "Taylor-Galerkin" algorithm, and by Baker and Kim [9] as the "Taylor weak statement" construction.

1.2 Advection-diffusion equation and finite difference method

The linear scalar advection-diffusion equation is used in this chapter as a model problem for comparative development of finite difference and subsequent finite element methods. In later chapters, the methods will be extended to nonlinear vector equation systems such as the incompressible Navier-Stokes equations.

The advection-diffusion equation with mixed Dirichlet and Neumann boundary conditions is

$$\begin{aligned} \mathcal{L}(q) = \frac{\partial q}{\partial t} + \vec{u}(\vec{x}) \cdot \nabla q - \nu \nabla \cdot \nabla q = s(\vec{x}) \ \text{ in } \Omega \in \mathcal{R}^n \,, \\ q = q_b \text{ on } \Gamma_1 \,, \ \frac{\partial q}{\partial n} = g \text{ on } \Gamma_2 \,, \ \Gamma_1 \cup \Gamma_2 = \partial\Omega \,, \end{aligned} \tag{1.1}$$

with prescribed data for viscosity ν, velocity $\vec{u}(\vec{x})$, source $s(\vec{x})$, and boundary data q_b and g. The domain Ω lies in $\mathcal{R}^n$, $n = 1$, 2, or 3, with boundary $\partial\Omega$.

A representative finite difference construction is developed using Taylor series as in Roache [67]. A two-dimensional centered construction in regular Cartesian geometry at the grid point (x_j, y_k) results in the nodal approximation $Q_{j,k}$ of the dependent variable $q(x,y)$, with approximations of the space derivatives:

$$\begin{aligned} \frac{\partial q}{\partial x} &\approx \delta_x Q \equiv \frac{Q_{j+1,k} - Q_{j-1,k}}{2\Delta_x} + O(\Delta_x^2) \,, \\ \frac{\partial q}{\partial y} &\approx \delta_y Q \equiv \frac{Q_{j,k+1} - Q_{j,k-1}}{2\Delta_y} + O(\Delta_y^2) \,, \\ \frac{\partial^2 q}{\partial x^2} &\approx \delta_x{}^2 Q \equiv \frac{Q_{j-1,k} - 2Q_{j,k} + Q_{j+1,k}}{\Delta_x^2} + O(\Delta_x^2) \,, \\ \frac{\partial^2 q}{\partial y^2} &\approx \delta_y{}^2 Q \equiv \frac{Q_{j,k-1} - 2Q_{j,k} + Q_{j,k+1}}{\Delta_y^2} + O(\Delta_y^2) \,. \end{aligned} \tag{1.2}$$

Although the first time derivative may be similarly approximated as the first space derivative above (the "leapfrog" method), a more common form combines forward and

backward Euler method in the weighted θ-implicit approximation of Stone and Brian [73],

$$\begin{aligned} Q^n_{j,k} &= Q_{j,k}(t^n), \\ Q^{n+1}_{j,k} &= Q_{j,k}(t^n + \Delta_t), \\ \frac{\partial q}{\partial t} &\approx \frac{Q^{n+1}_{j,k} - Q^n_{j,k}}{\Delta_t} + O(\Delta_t)\,, \\ Q^{n+\theta}_{j,k} &= \theta Q^{n+1}_{j,k} + (1-\theta) Q^n_{j,k}\,. \end{aligned} \tag{1.3}$$

Continuous known data is usually approximated as the interpolated values at the nodes of the mesh, $U_{j,k} \approx u(x_j, y_k)$, $W_{j,k} \approx w(x_j, y_k)$, $S_{j,k} \approx s(x_j, y_k)$. The subscripts are omitted here as the interpolated data are all taken at the (j,k) node at (x_j, y_k). Substituting the approximations in (1.1), the "stencil" [31] or "computational molecule" [3] representation, here denoted $\mathcal{S}(\cdot)$ is shown with the (j,k) node as the center element of the matrices,

$$\begin{aligned} \mathcal{S}\Big(\mathcal{L}^h(Q)\Big) = &\left|\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}\right| \frac{Q^{n+1} - Q^n}{\Delta_t} \\ + \Bigg[&\frac{U}{2\Delta_x} \left|\begin{matrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{matrix}\right| + \frac{W}{2\Delta_y} \left|\begin{matrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{matrix}\right| + \frac{\nu}{\Delta_x^2} \left|\begin{matrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{matrix}\right| \\ &+ \frac{\nu}{\Delta_y^2} \left|\begin{matrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{matrix}\right| \Bigg] Q^{n+\theta} + \left|\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}\right| S + O(\Delta_x^2, \Delta_y^2, \Delta_t) = 0\,. \end{aligned} \tag{1.4}$$

In the stencil notation, the center element of the 3×3 stencil matrix represents the coefficient of the approximation at the (j,k) node, the upper right element represents the $(j+1, k+1)$ node, and so on. Various constructions such as one-sided difference equations may be used at Neumann boundaries, and substitutions may be used at at Dirichlet boundaries. The one-sided boundary condition has the disadvantage of either reducing the order of accuracy of the approximation or, if the order is maintained, increasing the bandwidth [3]. After boundary conditions are approximated, the presumed known Q^n and the unknown Q^{n+1} may be separated and solved for Q^{n+1}, defining a solvable time-stepping approximation.

1.3 Upwind method

Courant et al. [24] observed that spurious error waves (to be demonstrated) are generated in the approximate solution when the coefficient $\frac{U}{2\Delta_x}$ of the velocity term is at least several times larger than the coefficient $\frac{\nu}{\Delta_x^2}$ of the diffusion term. Such is the case in many practical problems unless Δ_x is made very small. Those spurious waves could be avoided by replacing the centered first derivative approximations δ_x, δ_y at the point (j,k) by one-sided approximations δ_x^-, δ_y^- biased towards the direction from which the local flow originates, called the upwind direction and shown here for positive velocity components,

$$\begin{aligned} \frac{\partial q}{\partial x} &\approx \delta_x^- Q \equiv \frac{Q_{j,k} - Q_{j-1,k}}{\Delta_x} + O(\Delta_x) , \\ \frac{\partial q}{\partial y} &\approx \delta_y^- Q \equiv \frac{Q_{j,k} - Q_{j,k-1}}{\Delta_y} + O(\Delta_y) . \end{aligned} \tag{1.5}$$

The upwind approximations reduce the order of accuracy from two to one, and are equivalent to a combination of the centered differences (1.2) for the first and second derivatives,

$$\begin{aligned} \delta_x^- Q &= (\delta_x - \frac{\Delta_x}{2}\delta_x^2)\, Q , \\ \delta_y^- Q &= (\delta_y - \frac{\Delta_y}{2}\delta_y^2)\, Q . \end{aligned} \tag{1.6}$$

This is also equivalent to increasing the viscosity ν in (1.4) to $\hat{\nu} = \nu + \frac{|U|\Delta_x}{2} + \frac{|W|\Delta_y}{2}$, hence the "artificial viscosity" construction.

A potential difficulty is that the solution obtained via the increased viscosity can not be distinguished from a solution obtained using centered differences with an arbitrarily increased physical viscosity. Since the physical viscosity determines the Reynolds number, the most significant non-dimensional parameter of fluid flow, the artificial viscosity construction may be interpreted as an arbitrary change to the problem statement rather than a method for obtaining a more accurate approximation [45].

1.4 Lax-Wendroff method

Lax and Wendroff [53] in 1960 proposed a different approach for the one-dimen-

sional advection-diffusion equation,

$$\mathcal{L}(q) = \frac{\partial q}{\partial t} + u\frac{\partial q}{\partial x} - \nu\frac{\partial^2 q}{\partial x^2} = 0\,. \tag{1.7}$$

This equation has an explicit centered FDM approximation of the form

$$\begin{aligned}\mathcal{L}^h(Q) =& \frac{Q^{n+1} - Q^n}{\Delta_t} + \frac{U}{2\Delta_x}(Q^n_{j+1} - Q^n_{j-1}) \\ &- \frac{\nu}{\Delta_x^2}(Q^n_{j+1} - 2Q^n_j + Q^n_{j-1}) + O(\Delta_x, \Delta_t) = 0\,.\end{aligned} \tag{1.8}$$

Assuming sufficient continuity exists, a forward Taylor series in time is defined,

$$q^{n+1} = q^n + \Delta_t \frac{\partial q}{\partial t}^n + \frac{\Delta_t^2}{2}\frac{\partial^2 q}{\partial t^2}^n + O(\Delta_t^3)\,, \tag{1.9}$$

and rearranged to form a higher-order accurate approximation of the time derivative as

$$\frac{\partial q}{\partial t}^n = \frac{q^{n+1} - q^n}{\Delta_t} - \frac{\Delta_t}{2}\frac{\partial^2 q}{\partial t^2}^n + O(\Delta_t^2)\,. \tag{1.10}$$

The time derivative approximation may be equated to a higher-order space derivative using (1.7) and assuming zero physical diffusion, yielding

$$\begin{aligned}&\frac{\partial q}{\partial t} = -u\frac{\partial q}{\partial x}\,, \\ &\frac{\partial^2 q}{\partial t^2} = -\frac{\partial}{\partial t}u\frac{\partial q}{\partial x} = -u\frac{\partial}{\partial t}\frac{\partial q}{\partial x} = -u\frac{\partial}{\partial x}\frac{\partial q}{\partial t} = u^2\frac{\partial^2 q}{\partial x^2}\,.\end{aligned} \tag{1.11}$$

Substituting (1.11) in (1.10) yields a modified semi-discrete equation in which the higher-order time derivative errors are canceled by the modified space discretization terms, yielding

$$\begin{aligned}&\frac{\partial q}{\partial t}^n = \frac{q^{n+1} - q^n}{\Delta_t} - \frac{u^2\Delta_t}{2}\frac{\partial^2 q}{\partial x^2}^n + O(\Delta_t^2)\,, \\ &\mathcal{L}^h(q) = \frac{q^{n+1} - q^n}{\Delta_t} + u\frac{\partial q}{\partial x} - \frac{u^2\Delta_t}{2}\frac{\partial^2 q}{\partial x^2}^n + O(\Delta_t^2) = 0\,.\end{aligned} \tag{1.12}$$

The resulting second-order approximation to the time derivative (the first and third terms on the right-hand side) is substituted for the first-order approximation in (1.8). In the fully discrete form, the space derivatives are also replaced by their centered discrete approxi-

mations (1.2), yielding the Lax-Wendroff finite difference form

$$\hat{\mathcal{L}}^h(Q) = \frac{Q^{n+1} - Q^n}{\Delta_t} + \frac{U}{2\Delta_x}(Q^n_{j+1} - Q^n_{j-1}) - \frac{U^2\Delta_t}{2\Delta_x^2}(Q^n_{j+1} - 2Q^n_j + Q^n_{j-1}) + O(\Delta_x, \Delta_t^2) = 0\,. \tag{1.13}$$

Comparing to (1.6) and (1.9) shows that the Lax-Wendroff construction can be interpreted as adding an artificial viscosity of magnitude $\frac{U^2\Delta_t}{2}$, compared to the $\frac{U}{2\Delta_x}$ of the upwind method. The Lax-Wendroff artificial viscosity depends on the timestep Δ_t rather than the discretization Δ_x and has mostly been used for inherently transient problems. Multiplying the equation through by Δ_t makes the coefficients nondimensional as $\frac{U\Delta_t}{2\Delta_x} = \frac{C}{2}$ on the convection term and $\frac{U^2\Delta_t^2}{2\Delta_x^2} = \frac{C^2}{2}$ on the diffusion term, where the parameter C, called the Courant number, is a significant non-dimensional parameter in transient approximations [67].

1.5 This dissertation

Practical computational fluid dynamics requires accurate and efficient approximation of three-dimensional turbulent flow in non-Cartesian geometry; the requirements are not yet fully achieved. Several components of CFD models require improvement to meet these requirements, including topics such as automatic meshing and turbulence modeling. Accurate approximation presently requires fine meshes, which are generally impractical in three dimensions because of the resulting heavy computing demand. Accuracy with efficiency on a relatively coarse mesh will be a primary objective of this research.

In this dissertation research project, multi-dimensional Lax-Wendroff Taylor weak statement constructions and similar competing constructions are investigated for coarse-mesh accuracy. Finite element methods are used to directly facilitate non-Cartesian geometry and flexibility of approximation. Testing of algorithms is performed using two-dimensional laminar flow to allow comprehensive evaluation in a reasonable time while still accounting for directional effects not found in one dimension. Other reasons to simulate only laminar flow are that turbulence models create diffusion which complicates the evaluation of artificial diffusion algorithms, and that accurate comparative solutions to benchmark problems are generally available only for laminar flow.

Analysis procedures such as the Fourier modal analysis and the modified equation method are obtained from the literature and extended as needed. The procedures are used to analyze, and optimize for time accuracy and stability, TWS and competitive algorithms. Linear basis algorithms in one to three dimensions and quadratic and cubic basis algorithms in one dimension are fully analyzed.

Various artificial diffusion algorithms (SUPG, exponential-Petrov-Galerkin, SGM, bubble function,least-squares) are reformulated, if possible, as TWS algorithms and compared. Appropriate dissipation time/length scaling is compared and assessed.

The resulting algorithms are tested on verification problems (simplified analytical solutions) in one to three dimensions. Select algorithms are also tested on two-dimensional benchmark problems for the Navier-Stokes equations. The better-performing algorithms are identified and their performance quantized in integral and discrete norms.

Chapter 2

Galerkin weak statement finite element method

2.1 Weak statement form

The scalar linear advection-diffusion equation (1.1) remains as as a model problem, with extensions in later chapters to nonlinear vector systems. The classical or "strong" formulation $\mathcal{L}(q(\vec{x},t)) = s(\vec{x})$ is not solvable in general [85], thus a good approximation is sought to the unknown solution. One approach is facilitated by recasting the model problem into the "weak" formulation [14],

$$\mathcal{W}\Big(\mathcal{L}(q)-s\Big)=\int_{\Omega} w(\vec{x})\left[\mathcal{L}\Big(q(\vec{x},t)\Big)-s(\vec{x},t)\right]d\Omega=0\,. \tag{2.1}$$

The terminology refers to, in some applications, a weakening of the necessary continuity of the solution $q(\vec{x},t)$. The function $w(\vec{x})$ is an arbitrary non-zero "test" function, the same boundary conditions apply as to the classical formulation (1.1), and solutions of (1.1) will also be solutions of (2.1) though not necessarily the converse.

Since, in CFD applications, $\mathcal{L}(q(\vec{x},t)) = s(\vec{x})$ is often a second-order differential equation represented in divergence flux vector form, one common weak formulation is to define $w(\vec{x})$ to be a constant while subdividing Ω into local volumes with artificial boundaries. The divergence theorem is then used to project the flux vector efflux onto the local boundaries, resulting in a "finite volume" method. On regular Cartesian subdomains, the resulting method is generally similar to a finite difference method, although the implementation of boundary conditions usually differs.

Another weak form is often termed the "weighted residual formulation" [33], which requires a differentiable test function. The test function and the solution are generally assumed to both be members of the infinite-dimensional linear space $\mathrm{L}^2(\Omega)$, where $w(\vec{x}) \in \mathrm{L}^2(\Omega)$ if $w(\vec{x})$ is piecewise continuous and $\int_\Omega w(\vec{x})^2 d\Omega < \infty$. Usually the functions are further restricted to the Hilbert space $\mathrm{H}^1(\Omega)$, in which the functions and their first derivatives both belong to $\mathrm{L}^2(\Omega)$. This restriction allows the application of "integration by parts" or the Green-Gauss theorem to the second-order derivatives in $\mathcal{L}(q(\vec{x},t))$. Substituting the advection-diffusion equation (1.1) in the weak formulation (2.1) and applying

the Green-Gauss theorem yields

$$\mathcal{W}(\mathcal{L}(q)-s)) = \int_\Omega \left[w(\vec{x}) \left(\frac{\partial q(\vec{x},t)}{\partial t} + \vec{u} \cdot \nabla q(\vec{x},t) - s(\vec{x}) \right) + \nu \nabla w(\vec{x}) \cdot \nabla q(\vec{x},t) \right] d\Omega$$
$$- \int_{\Gamma_2} \nu w(\vec{x}) \frac{\partial q(\vec{x},t)}{\partial n} d\Gamma = 0 \, , \, \forall w(\vec{x}) \in \mathrm{H}^1(\Omega) \, . \tag{2.2}$$

The weak form (2.2) highlights some important distinctions from the FVM. The diffusive term $-\nu \nabla \cdot \nabla q$ is integrated by parts rather than projected to a boundary via the divergence theorem. Additionally, first order flux vector terms may optionally be integrated by parts, not shown here. After eventual subdivision of the domain Ω, interior boundary integrals do not require evaluation since the piecewise continuous test function allows them to cancel [3]. In practice, the domain boundary integral with Neumann boundary condition, $\frac{\partial q(\vec{x},t)}{\partial n} = g$ on Γ_2, is substituted in the boundary integral term and the normal derivative term may not require approximation, depending on the form of g. This may allow significant simplification of (2.2), particularly in non-Cartesian geometry.

The weighted residual method, discretized as a finite element method, also facilitates a more rigorous development of an approximation to (2.2). Since Hilbert spaces are complete [81], the solution q and test function w may be represented by convergent infinite series, i.e. Fourier or power series, hence approximated by truncated series $q_M(\vec{x},t)$ and $w_M(\vec{x})$,

$$q(\vec{x},t) \equiv \sum_{m=1}^{\infty} \psi_m(\vec{x}) Q_m(t) \approx q_M(\vec{x},t) \equiv \sum_{m=1}^{M} \psi_m(\vec{x}) Q_m(t) \, ,$$
$$w(\vec{x}) \equiv \sum_{l=1}^{\infty} \chi_l(\vec{x}) W_l \approx w_M(\vec{x}) \equiv \sum_{l=1}^{M} \chi_l(\vec{x}) W_l \, . \tag{2.3}$$

where the coefficients $Q_m(t)$ are the nodal values of the approximation and the coefficients W_l must be arbitrary. In theory, the expansion for $q(\vec{x},t)$ must meet any Dirichlet boundary conditions on Γ_1, but in practice, those boundary conditions are met by matrix substitution in a completed discrete approximation. Substituting the approximations of (2.3), and similar expansions for $s(\vec{x},t)$ and g on Γ_2, in the continuum weak form (2.2) yields the

approximate weak form $\mathcal{W}^h$,

$$\mathcal{W}^h(\mathcal{L}q_M - s_M) = \sum_{l=1}^{M} W_l \sum_{m=1}^{M} \left\{ \int_{\Omega^h} \left[\chi_l \left(\psi_m \frac{dQ_m}{dt} + \vec{U} \cdot \nabla \psi_m Q_m(t) - \psi_m S_m \right) + \nu \nabla \chi_l \nabla \psi_m Q_m(t) \right] d\Omega^h - \int_{\Gamma_2} \nu \chi_l \psi_m G_m(t) d\Gamma \right\} = 0 . \tag{2.4}$$

Condensing the notation for brevity in the portion of (2.4) inside braces, and assuming a linear relationship (to be defined) between $\frac{dQ_m}{dt}$ and $Q_m(t)$, (2.4) is the scalar

$$\mathcal{W}^h(\mathcal{L}q_M - s_M) = \sum_{l=1}^{M} W_l \sum_{m=1}^{M} \left\{ \mathrm{K}_{lm} Q_m(t) - \mathrm{F}_l \right\} = 0 . \tag{2.5}$$

Since the W_l are arbitrary, choose, for example, $W = \{\mathrm{C}_1, 0, 0, \ldots\}$, resulting in

$$\mathrm{C}_1 \sum_{m=1}^{M} \left\{ \mathrm{K}_{1m} Q_m(t) - \mathrm{F}_1 \right\} = 0 \text{ for } l = 1 , \tag{2.6}$$

and $W = \{0, \mathrm{C}_2, 0, \ldots\}$, resulting in

$$\mathrm{C}_2 \sum_{m=1}^{M} \left\{ \mathrm{K}_{2m} Q_m(t) - \mathrm{F}_2 \right\} = 0 \text{ for } l = 2 , \tag{2.7}$$

and so on. Then, dividing each equation by the constants C_l, (2.5) is equivalent to

$$\mathcal{W}^h(\mathcal{L}q_M - s_M) = \sum_{m=1}^{M} \left\{ \mathrm{K}_{lm} Q_m(t) - \mathrm{F}_l \right\} = 0 , \ (l = 1, \ldots, M) , \tag{2.8}$$

which has the form of an $M \times M$ matrix multiplication which, if invertible, is solvable for the coefficients $Q_m(t)$.

Equivalently, assuming that $\mathcal{L}(q(\vec{x}, t))$ is elliptic and taking the stationary point of $\mathcal{W}^h(\mathcal{L}q_M - s_M)$ with respect to W_l results in a "variational" method with an identical

result, expanded for clarity as

$$\mathcal{W}^h\Big(\mathcal{L}(q_M)-s_M\Big)=\sum_{m=1}^{M}\left\{\int_{\Omega^h}\left[\chi_l\left(\psi_m\frac{dQ_m}{dt}+\vec{U}\cdot\nabla\psi_m Q_m(t)-\psi_m S_m\right)\right.\right.$$
$$\left.\left.+\nu\nabla\chi_l\nabla\psi_m Q_m(t)\right]d\Omega^h-\int_{\Gamma_2}\nu\chi_l\psi_m G_m(t)d\Gamma\right\}=0\,,\ (l=1,\ldots,M)\,. \tag{2.9}$$

Since (2.9) is required to hold for all $w^h\in \mathrm{H}^1$, the test interpolation function χ and the corresponding trial or basis function ψ may be defined to be equal, resulting in a "Galerkin" method. For elliptic model problems the Galerkin method (in that case also the variational method) can be shown by Cea's lemma [11] to be the optimal approximation in H^1, i.e. the approximation with the smallest error in H^1. The question of the optimal approximation for the advection-diffusion equation is not settled, and that equation is often approximated by a "Petrov-Galerkin" method (to be examined) in which the test function interpolation is modified to produce differing performance properties.

The $M\times M$ integral matrices in (2.9) are functions of Ω^h and are independent of the data (except possibly $\vec{u}$); here the matrices are replaced by symbols in a modification of the notation of Baker [8]. The notation is expanded to include an n-dimensional matrix form $\mathrm{M}[\cdots]$, assumed to include any necessary geometric transformation data. The form $\mathrm{M}[\cdots]$ is a non-specific reference to the one-dimensional matrices $\mathrm{A}[\cdots]$, two-dimensional $\mathrm{B}[\cdots]$, and three-dimensional $\mathrm{C}[\cdots]$ and their further refinements per various basis functions. Letters J, K following the prefix indicate a tensor summation over the range of space derivatives; numbers indicate individual space derivatives in transform space (η,ξ), and letters X, Y, Z indicate individual space derivatives in Cartesian space (x,y,z), In addition, since each matrix incorporates the double summation over (l,m), the summation is indicated by an outer vector product or column-row matrix product such as $\{\chi\}\{\psi\}^T$, with

examples

$$\begin{aligned}
\texttt{M200} &= \sum_{l=1}^{M} \int_{\Omega^h} \chi_l \psi_m \, d\Omega^h \; (m = 1,\ldots,M) \; = \int_{\Omega^h} \{\chi\}\{\psi\}^T \, d\Omega^h \,, \\
\bar{U}_k \, \texttt{M20K} &= \sum_{l=1}^{M} \int_{\Omega^h} \chi_l \vec{U} \cdot \nabla \psi_m \, d\Omega^h \; (m = 1,\ldots,M) \; = \int_{\Omega^h} \{\chi\}\vec{U} \cdot \nabla\{\psi\}^T \, d\Omega^h \,, \\
\texttt{M2KK} &= \sum_{l=1}^{M} \int_{\Omega^h} \nabla\chi_l \cdot \nabla \psi_m \, d\Omega^h \; (m = 1,\ldots,M) \; = \int_{\Omega^h} \nabla\{\chi\}\vec{U} \cdot \nabla\{\psi\}^T \, d\Omega^h \,, \\
\texttt{M200N} &= \sum_{l=1}^{M} \int_{\Gamma_2} \chi_l \psi_m \, d\Gamma^h \; (m = 1,\ldots,M) \; = \int_{\Gamma_2} \{\chi\}\{\psi\}^T \, d\Gamma^h \,.
\end{aligned} \tag{2.10}$$

The boundary integral matrix M200N is similar to the interpolation matrix M200, but of dimension $n-1$ rather than n. If there is significant variation in $\vec{u}$, the velocity may also be interpolated similarly to q for an alternate form of the convection matrix M20K (cf. Baker [8]),

$$\begin{aligned}
\{U_k\}^T \, \texttt{M300K} &= \int_{\Omega^h} \chi_l \psi_n \{\vec{U}\} \cdot \nabla \psi_m \, d\Omega^h \; (m = 1,\ldots,M) \\
&= \int_{\Omega^h} \{\chi\}\{\psi\}^T \{\vec{U}\} \cdot \nabla\{\psi\}^T \, d\Omega^h \,.
\end{aligned} \tag{2.11}$$

2.2 Galerkin finite element form

The generalized Galerkin matrix form (2.9) remains abstract in that the test function set $\{\chi\}$ and interpolation function set $\{\psi\}$ are as yet unspecified members of the Hilbert space $H_h^1(\Omega)$. Functions used in practice are generally Lagrange or Hermite polynomials of degree one to three, defined locally on small subdivisions Ω_e of Ω and zero elsewhere. The subdivisions are called "elements", hence a "finite element" method or FEM.

The form of (2.9), as a matrix sum of integrals, allows the domain Ω^h of the integrals (2.11) to be subdivided into E finite elements Ω_e. The matrices may then be computed individually and summed or "assembled" over the domain. Because the compact interpolation is continuous at the nodes, only the elements adjacent to a node are assembled into the equation for that node. Similarly, only elements adjacent to a boundary are assembled for a boundary integral, a significant distinction from the FVM in which all

integrals of flux-vector derivatives are treated as interior boundary integrals. A numerical example of the assembly process is shown for two-dimensional bilinear elements in (2.48).

Symbolically, the matrices $\mathtt{M2}[\cdot]$ from (2.10) are assembled ($\mathcal{A}_e$) from "element matrices" $\mathtt{M2}[\cdot]_e$ as

$$\begin{aligned}
\mathtt{M200} &= \mathcal{A}_e(\mathtt{M200}_e) = \mathcal{A}_e\left(\int_{\Omega^e}\{\chi\}\{\psi\}^T d\Omega_e\right), \\
\{U_k\}^T\,\mathtt{M300K} &= \mathcal{A}_e(\{U_k\}^T\,\mathtt{M300K}_e) = \mathcal{A}_e\left(\int_{\Omega^e}\{\chi\}\{\psi\}^T\{\bar{U}\}\cdot\nabla\{\psi\}^T d\Omega_e\right), \\
\bar{U}_k\,\mathtt{M20K} &= \mathcal{A}_e(\bar{U}_k\,\mathtt{M20K}_e) = \mathcal{A}_e\left(\int_{\Omega^e}\{\chi\}\vec{U}\cdot\nabla\{\psi\}^T d\Omega_e\right), \\
\mathtt{M2KK} &= \mathcal{A}_e(\mathtt{M2KK}_e) = \mathcal{A}_e\left(\int_{\Omega^e}\nabla\{\chi\}\cdot\nabla\{\psi\}^T d\Omega_e\right), \\
\mathtt{M200N} &= \mathcal{A}_e(\mathtt{M200N}_e) = \mathcal{A}_e\left(\int_{\Gamma_2}\{\chi\}\{\psi\}^T d\Gamma_e\right).
\end{aligned} \tag{2.12}$$

Combining (2.12) with (2.9) produces the matrix statement,

$$\begin{aligned}
\mathcal{W}^h\Big(\mathcal{L}(\{Q(t)\}) - \{S(T)\}\Big) = +\big(\{U_k\}\,\mathtt{M300K} + \nu(\,\mathtt{M2KK} - \mathtt{M200N})\big)\{Q(t)\} \\
- \mathtt{M200}\{S(t)\}\Big] = 0,
\end{aligned} \tag{2.13}$$

which on evaluation of the matrix integrals yields a system of ordinary differential equations for $Q(t)$. In this research the ODE system is approximated by a two-step discrete finite difference in time, usually the θ-implicit method (1.6). Substituting in (2.13),

$$\begin{aligned}
\mathcal{W}^h\Big(\mathcal{L}(\{Q(t)\}) - \{S(T)\}\Big) = \Big[\mathtt{M200}\frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} + \Big(\{U_k\}\,\mathtt{M300K} \\
+ \nu(\,\mathtt{M2KK} - \mathtt{M200N})\Big)\{Q\}^{n+\theta} - \mathtt{M200}\{S\}^{n+\theta}\Big] = 0.
\end{aligned} \tag{2.14}$$

Assuming known values of $\{Q\}^n$ and $\{S\}^n$, and assuming necessary Dirichlet boundary conditions are applied, (2.14) may be separated and rearranged to solve for $\{Q\}^{n+1}$ as

$$\begin{aligned}
&\left[\mathtt{M200} + \theta\Delta_t\left(\{U_k\}\,\mathtt{M300K} + \nu(\,\mathtt{M2KK} - \mathtt{M200N})\right)\right]\{Q\}^{n+1} \\
&= \left[\mathtt{M200} + (\theta-1)\Delta_t\left(\{U_k\}\,\mathtt{M300K} + \nu(\,\mathtt{M2KK} - \mathtt{M200N})\right)\right]\{Q\}^n - \mathtt{M200}\,S^{n+\theta},
\end{aligned} \tag{2.15}$$

usually rearranged to solve for the difference over the timestep Δ_t as the the delta-form,

$$\begin{aligned}&\left[\texttt{M200}+\theta\Delta_t\left(\{U_k\}\,\texttt{M300K}+\nu(\,\texttt{M2KK}-\,\texttt{M200N})\right)\right](\{Q\}^{n+1}-\{Q\}^n)\\&=-\Delta_t\left[\left(\{U_k\}\,\texttt{M300K}+\nu(\,\texttt{M2KK}-\,\texttt{M200N})\right)\{Q\}^n-\texttt{M200}\,\{S\}^{n+\theta}\right].\end{aligned}\tag{2.16}$$

2.2.1 One-dimensional linear basis examples and matrix forms

The linear basis element is obtained using the Lagrange polynomial interpolation function (cf. Zienkiewicz [88]). For convenience in performing eventual integration, the non-dimensional form is used, with the x-coordinate of the left side of the element defined as x_0 and the element length as Δ_x as

$$\{\psi\}=\begin{bmatrix}1-\zeta\\ \zeta\end{bmatrix},\ \zeta=\frac{x-x_0}{\Delta_x},\tag{2.17}$$

which in turn defines $x=\Delta_x\zeta+x_0$ and $0<zeta<1$. The Galerkin selection $\{\chi\}=\{\psi\}$ yields element matrices $\texttt{M2}[\cdot]$, $\texttt{M3}[\cdot]$, and each element matrix in (2.12) may be expressed as the element-dependent dimensional data, Δ_x in one dimension, multiplying element-independent matrices of constants here designated $\texttt{A2}[\cdot]$, $\texttt{A3}[\cdot]$. The integrations are elementary in the transformed ζ space on the reference element, yielding

$$\begin{aligned}\texttt{M200}_e&=\int_{\Omega^e}\{\psi\}\{\psi\}^T d\Omega_e=\Delta_x\int_0^1\{\psi\}\{\psi\}^T d\zeta=\Delta_x\,\texttt{A200L}\,,\\ \{U_k\}^T\,\texttt{M300K}_e&=\{U\}^T\,\texttt{M300X}_e=\{U\}^T\int_{\Omega^e}\{\psi\}\{\psi\}\frac{\partial\{\psi\}^T}{\partial x}d\Omega_e\\&=\{U\}^T\int_0^1\{\psi\}\{\psi\}\frac{\partial\{\psi\}^T}{\partial\zeta}d\zeta=\{U\}^T\,\texttt{A3001L}\,,\\ \bar{U}_k\,\texttt{M20K}_e&=\bar{U}\,\texttt{M20X}_e=\bar{U}\int_{\Omega^e}\{\psi\}\frac{\partial\{\psi\}^T}{\partial x}d\Omega_e=\bar{U}\int_0^1\{\psi\}\frac{\partial\{\psi\}^T}{\partial\zeta}d\zeta=\bar{U}\,\texttt{A201L}\,,\\ \texttt{M2KK}_e&=\,\texttt{M2XX}_e=\int_{\Omega^e}\frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}d\Omega_e=\frac{1}{\Delta_x}\int_0^1\frac{\partial\{\psi\}}{\partial\zeta}\frac{\partial\{\psi\}^T}{\partial\zeta}d\zeta\\&=\frac{1}{\Delta_x}\,\texttt{A211L}\,.\end{aligned}\tag{2.18}$$

The linear basis arrays of constants A2[·], A3[·] evaluate as

$$\begin{aligned} \texttt{A200L} &= \frac{1}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}, \quad \texttt{A211L} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \\ \texttt{A201L} &= \frac{1}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}, \quad \texttt{A3001L} = \frac{1}{6}\begin{bmatrix} -(2,1) & (2,1) \\ -(1,2) & (1,2) \end{bmatrix}. \end{aligned} \tag{2.19}$$

2.2.2 One-dimensional quadratic basis form and matrices

The quadratic basis element is similarly obtained using the quadratic Lagrange polynomial interpolation function (cf. Zienkiewicz [88]),

$$\{\psi\} = \begin{bmatrix} (1-\zeta)(1-2\zeta) \\ 4\zeta(1-\zeta) \\ \zeta(2\zeta-1) \end{bmatrix}. \tag{2.20}$$

After integration, the resulting element matrices are similarly expressed as products of element-dependent data Δ_x onto element-independent matrices of constants, replacing those in (2.19) with

$$\texttt{A200Q} = \frac{1}{30}\begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix}, \quad \texttt{A211Q} = \frac{1}{6}\begin{bmatrix} 7 & -8 & 1 \\ -8 & 16 & -8 \\ 1 & -8 & 7 \end{bmatrix}, \tag{2.21}$$

$$\texttt{A201Q} = \frac{1}{6}\begin{bmatrix} -3 & 4 & -1 \\ -4 & 0 & -4 \\ 1 & -4 & 3 \end{bmatrix}, \tag{2.22}$$

$$\texttt{A3001Q} = \frac{1}{30}\begin{bmatrix} (-10,-6,1) & (12,8,0) & (-2,-2,-1) \\ (-6,-16,2) & (8,0,-8) & (-2,16,6) \\ (1,2,2) & (0,-8,-12) & (-1,6,10) \end{bmatrix}. \tag{2.23}$$

2.2.3 One-dimensional cubic basis form and matrices

The cubic basis element is obtained using the cubic Lagrange polynomial interpo-

lation function (cf. Zienkiewicz [88]),

$$\{\psi\} = \frac{1}{2}\begin{bmatrix} -(\zeta-1)(3\zeta-2)(3\zeta-1) \\ 9\zeta(\zeta-1)(3\zeta-2) \\ -9\zeta(\zeta-1)(3\zeta-1) \\ \zeta(3\zeta-1)(3\zeta-2) \end{bmatrix}. \tag{2.24}$$

For the Galerkin form, the element-independent constant matrices replacing (2.19) are

$$\texttt{A200C} = \frac{1}{1680}\begin{bmatrix} 128 & 99 & -36 & 19 \\ 99 & 648 & -81 & -36 \\ -36 & -81 & 648 & 99 \\ 19 & -36 & 99 & 128 \end{bmatrix}, \tag{2.25}$$

$$\texttt{A211C} = \frac{1}{40}\begin{bmatrix} 148 & -189 & 54 & -13 \\ -189 & 432 & -297 & 54 \\ 54 & -297 & 432 & -189 \\ -13 & 54 & -189 & 148 \end{bmatrix}, \tag{2.26}$$

$$\texttt{A201C} = \frac{1}{80}\begin{bmatrix} -40 & 56 & -24 & 7 \\ -56 & 0 & 81 & -24 \\ 24 & -81 & 0 & 56 \\ -7 & 24 & -56 & 40 \end{bmatrix}, \tag{2.27}$$

$$\texttt{A3001C} = \frac{1}{84}\begin{bmatrix} (-2240,-1611,630,-139) & (3222,2025,-648,189) \\ (-1611,-4050,1053,-180) & (2025,0,-2187,162) \\ (630,1053,324,9) & (-648,-2187,-4374,405) \\ (-139,-180,9,-278) & (189,162,405,1260) \\ (-1260,-405,-162,-189) & (278,-9,180,139) \\ (-405,4374,2187,648) & (-9,-324,-1053,-630) \\ (-162,2187,0,-2025) & (180,-1053,4050,1611) \\ (-189,648,-2025,-3222) & (139,-630,1611,2240) \end{bmatrix}. \tag{2.28}$$

A Macsyma ® program producing the one-dimensional matrices for linear, quadratic, and cubic bases is shown in Appendix I.

2.2.4 Factorized form and matrices

2.2.5 Two-dimensional form and matrices

The element derivative matrices in two or three dimensions may be calculated by Cartesian components for illustration. The directional derivatives in element matrices defined on $\mathcal{R}^2$ are denoted here by X and Y, e.g. M2XX,

$$
\begin{aligned}
\{U_k\}\,\mathtt{M300K}_e &= \int_{\Omega^e} \{\chi\}\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\,d\Omega_e + \int_{\Omega^e} \{\chi\}\{\psi\}^T\{U_y\}\frac{\partial\{\psi\}^T}{\partial y}\,d\Omega_e \\
&= \{U_x\}^T \int_{\Omega^e} \{\psi\}\{\chi\}\frac{\partial\{\psi\}^T}{\partial x}\,d\Omega_e + \{U_y\}^T \int_{\Omega^e} \{\psi\}\{\chi\}\frac{\partial\{\psi\}^T}{\partial y}\,d\Omega_e \\
&= \{U_x\}^T\,\mathtt{M300X}_e + \{U_y\}^T\,\mathtt{M300Y}_e\,, \\
\bar{U}_k\,\mathtt{M20K}_e &= \int_{\Omega^e} \{\chi\}\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\,d\Omega_e + \int_{\Omega^e} \{\chi\}\{U_y\}\frac{\partial\{\psi\}^T}{\partial y}\,d\Omega_e \\
&= \bar{U}_x \int_{\Omega^e} \{\chi\}\frac{\partial\{\psi\}^T}{\partial x}\,d\Omega_e + \bar{U}_y \int_{\Omega^e} \{\chi\}\frac{\partial\{\psi\}^T}{\partial y}\,d\Omega_e \\
&= \bar{U}_x\,\mathtt{M20X}_e + \bar{U}_y\,\mathtt{M20Y}_e\,, \\
\mathtt{M2KK}_e &= \int_{\Omega^e} \frac{\partial\{\chi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}\,d\Omega_e + \int_{\Omega^e} \frac{\partial\{\chi\}}{\partial y}\frac{\partial\{\psi\}^T}{\partial y}\,d\Omega_e \\
&= \mathtt{M2XX}_e + \mathtt{M2YY}_e\,.
\end{aligned}
\tag{2.29}
$$

In this research, only bilinear basis functions are used in two dimensions, thus the element matrix designations do not include suffixes L, Q, or C indicating interpolation degree as in one dimension. Also, in this chapter only Galerkin methods are considered, thus $\{\chi\} = \{\psi\}$. In later chapters methods will be considered in which $\{\chi\}$ is modified.

The two-dimensional integrations are simplified on non-Cartesian geometries by transforming the space from the element Ω_e with coordinates (x, y) to a reference Cartesian element $\hat{\Omega}$ spanned by coordinates (η, ξ) in the same manner as ζ in one dimension. The non-dimensional coordinates in two or three dimensions customarily have a range of -1 to 1 rather than 0 to 1.

Assume a transformation exists which is invertible, thus

$$\mathtt{J}_e = \left[\frac{\partial x_i}{\partial \eta_j}\right]_e = \begin{bmatrix} \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} \\ \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \xi} \end{bmatrix}_e , \tag{2.30}$$

with inverse

$$\mathtt{J}_e^{-1} = \begin{bmatrix} \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \\ \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \end{bmatrix}_e = \frac{1}{|\mathtt{J}|_e} \begin{bmatrix} \frac{\partial y}{\partial \xi} & -\frac{\partial x}{\partial \xi} \\ -\frac{\partial y}{\partial \eta} & \frac{\partial x}{\partial \eta} \end{bmatrix}_e , \tag{2.31}$$

where $|\mathtt{J}|_e$ is the determinant of (2.30). This invertible transformation can be shown to exist provided the mesh is sufficiently regular. Then by the chain rule,

$$\begin{aligned} \frac{\partial \{\psi\}^T}{\partial x} &= \frac{\partial \{\psi\}^T}{\partial \eta}\frac{\partial \eta}{\partial x} + \frac{\partial \{\psi\}^T}{\partial \xi}\frac{\partial \xi}{\partial x} , \\ \frac{\partial \{\psi\}^T}{\partial y} &= \frac{\partial \{\psi\}^T}{\partial \eta}\frac{\partial \eta}{\partial y} + \frac{\partial \{\psi\}^T}{\partial \xi}\frac{\partial \xi}{\partial y} . \end{aligned} \tag{2.32}$$

The domain of integration is transformed to the reference element, cf. Zienkiewicz [88],

$$d\Omega_e = dx\,dy = |\mathtt{J}|_e d\hat{\Omega} = |\mathtt{J}|_e d\eta\, d\xi , \tag{2.33}$$

hence

$$\mathtt{M200}_e = \int_{\Omega^e} \{\chi\}\{\psi\}^T |\mathtt{J}|_e d\eta\, d\xi , \tag{2.34}$$

$$\begin{aligned} \{U_x\}^T \mathtt{M300X}_e &= \{U_x\}^T \int_{\Omega^e} \{\psi\}\{\chi\}\left(\frac{\partial \{\psi\}^T}{\partial \eta}\frac{\partial \eta}{\partial x} + \frac{\partial \{\psi\}^T}{\partial \xi}\frac{\partial \xi}{\partial x}\right) d\Omega_e \\ &= \{U_x\}^T \int_{\Omega^e} \{\psi\}\{\chi\}\left(\frac{\partial \{\psi\}^T}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial \{\psi\}^T}{\partial \xi}\frac{\partial y}{\partial \eta}\right) d\eta\, d\xi , \\ \{U_y\}^T \mathtt{M300Y}_e &= \{U_y\}^T \int_{\Omega^e} \{\psi\}\{\chi\}\left(\frac{\partial \{\psi\}^T}{\partial \eta}\frac{\partial \eta}{\partial y} + \frac{\partial \{\psi\}^T}{\partial \xi}\frac{\partial \xi}{\partial y}\right) d\Omega_e \\ &= \{U_y\}^T \int_{\Omega^e} \{\psi\}\{\chi\}\left(-\frac{\partial \{\psi\}^T}{\partial \eta}\frac{\partial x}{\partial \xi} + \frac{\partial \{\psi\}^T}{\partial \xi}\frac{\partial x}{\partial \eta}\right) d\eta\, d\xi , \end{aligned} \tag{2.35}$$

$$
\begin{aligned}
\bar{U}_x \,\mathtt{M20X}_e &= \bar{U}_x \int_{\Omega^e} \{\chi\} \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial \eta}{\partial x} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial \xi}{\partial x} \right) d\Omega_e \\
&= \bar{U}_x \int_{\Omega^e} \{\chi\} \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial y}{\partial \xi} - \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial y}{\partial \eta} \right) d\eta \, d\xi \,, \\
\bar{U}_y \,\mathtt{M20Y}_e &= \bar{U}_y \int_{\Omega^e} \{\chi\} \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial \eta}{\partial y} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial \xi}{\partial y} \right) d\Omega_e \\
&= \bar{U}_y \int_{\Omega^e} \{\chi\} \left(-\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial x}{\partial \xi} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial x}{\partial \eta} \right) d\eta \, d\xi \,,
\end{aligned}
\tag{2.36}
$$

$$
\begin{aligned}
\mathtt{M2XX}_e &= \int_{\Omega^e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial \eta}{\partial x} + \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial \xi}{\partial x} \right) \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial \eta}{\partial x} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial \xi}{\partial x} \right) d\Omega_e \\
&= \int_{\Omega^e} \frac{1}{|\mathtt{J}|_e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial y}{\partial \xi} - \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial y}{\partial \eta} \right) \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial y}{\partial \xi} - \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial y}{\partial \eta} \right) d\eta \, d\xi \,, \\
\mathtt{M2XY}_e &= \int_{\Omega^e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial \eta}{\partial x} + \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial \xi}{\partial x} \right) \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial \eta}{\partial y} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial \xi}{\partial y} \right) d\Omega_e \\
&= \int_{\Omega^e} \frac{1}{|\mathtt{J}|_e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial y}{\partial \xi} - \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial y}{\partial \eta} \right) \left(-\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial x}{\partial \xi} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial x}{\partial \eta} \right) d\eta \, d\xi \,, \\
\mathtt{M2YX}_e &= \int_{\Omega^e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial \eta}{\partial y} + \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial \xi}{\partial y} \right) \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial \eta}{\partial x} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial \xi}{\partial x} \right) d\Omega_e \\
&= \int_{\Omega^e} \frac{1}{|\mathtt{J}|_e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial x}{\partial \xi} - \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial x}{\partial \eta} \right) \left(-\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial y}{\partial \xi} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial y}{\partial \eta} \right) d\eta \, d\xi \,, \\
\mathtt{M2YY}_e &= \int_{\Omega^e} \left(\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial \eta}{\partial y} + \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial \xi}{\partial y} \right) \left(\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial \eta}{\partial y} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial \xi}{\partial y} \right) d\Omega_e \\
&= \int_{\Omega^e} \frac{1}{|\mathtt{J}|_e} \left(-\frac{\partial \{\chi\}}{\partial \eta} \frac{\partial x}{\partial \xi} + \frac{\partial \{\chi\}}{\partial \xi} \frac{\partial x}{\partial \eta} \right) \left(-\frac{\partial \{\psi\}^T}{\partial \eta} \frac{\partial x}{\partial \xi} + \frac{\partial \{\psi\}^T}{\partial \xi} \frac{\partial x}{\partial \eta} \right) d\eta \, d\xi \,, \\
\mathtt{M200N}_e &= \int_{\Gamma_2} \{\chi\} \{\psi\}^T d\Gamma = \Delta_s \, \mathtt{A200} \,.
\end{aligned}
\tag{2.37}
$$

An analytical form is not known for the diffusion integrals M2XX, M2YY, etc., since the integrand contains rational polynomials from the division by |J| (or multiplication by |J| and division by $|\mathtt{J}|^2$). In this research relatively regular meshes are used and it is assumed that coordinate transformation terms $\frac{\partial \eta}{\partial x}$, |J|, etc., may be evaluated at the element centroid ($\eta = 0, \xi = 0$), hence taken outside of the integration. The order-of-magnitude analysis detailing this assumption is given in Appendix II. If the assumption is not justified, then the element integrals may be approximated by quadrature to a desired order of accuracy

[88].

With element transfor nation terms removed from the integrals, the element-dependent matrix integrals M2[·], M3[·] in (x,y) space may be expressed as element data $\frac{\partial x}{\partial \eta}$, etc., which may be used to multiply element-independent matrices of constants denoted in two dimensions as B2[·]. In the matrices of constants, derivatives are taken in transform space (η,ξ) and represented by 1 or 2 respectively, e.g. B211. Rearranging (2.37) in that form,

$$\mathtt{M200}_e = |\mathtt{J}|_e\,\mathtt{B200}\,, \tag{2.38}$$

$$\begin{aligned}
\mathtt{M300X}_e &= \frac{\partial y}{\partial \xi}\,\mathtt{B3001} - \frac{\partial y}{\partial \eta}\,\mathtt{B3002}\,,\\
\mathtt{M300Y}_e &= -\frac{\partial x}{\partial \xi}\,\mathtt{B3002} + \frac{\partial x}{\partial \eta}\,\mathtt{B3001}\,,\\
\mathtt{M20X}_e &= \frac{\partial y}{\partial \xi}\,\mathtt{B201} - \frac{\partial y}{\partial \eta}\,\mathtt{B202}\,,\\
\mathtt{M20Y}_e &= -\frac{\partial x}{\partial \xi}\,\mathtt{B202} + \frac{\partial x}{\partial \eta}\,\mathtt{B202}\,,
\end{aligned} \tag{2.39}$$

$$\begin{aligned}
\mathtt{M2XX}_e &= \frac{1}{|\mathtt{J}|_e}\Big(\frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \xi}\,\mathtt{B211} + \frac{\partial y}{\partial \eta}\frac{\partial y}{\partial \eta}\,\mathtt{B222} - \frac{\partial y}{\partial \eta}\frac{\partial y}{\partial \xi}(\mathtt{B212}+\mathtt{B221})\Big)\,,\\
\mathtt{M2XY}_e &= \frac{1}{|\mathtt{J}|_e}\Big(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \xi}\,\mathtt{B211} + \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \eta}\,\mathtt{B222} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}(\mathtt{B212}+\mathtt{B221})\Big)\,,\\
\mathtt{M2YX}_e &= \frac{1}{|\mathtt{J}|_e}\Big(\frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \xi}\,\mathtt{B211} + \frac{\partial y}{\partial \eta}\frac{\partial x}{\partial \eta}\,\mathtt{B222} - \frac{\partial y}{\partial \eta}\frac{\partial x}{\partial \xi}(\mathtt{B212}+\mathtt{B221})\Big)\,,\\
\mathtt{M2YY}_e &= \frac{1}{|\mathtt{J}|_e}\Big(\frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \xi}\,\mathtt{B211} + \frac{\partial x}{\partial \eta}\frac{\partial x}{\partial \eta}\,\mathtt{B222} - \frac{\partial x}{\partial \eta}\frac{\partial x}{\partial \xi}(\mathtt{B212}+\mathtt{B221})\Big)\,.
\end{aligned} \tag{2.40}$$

The element-independent matrix definitions are

$$\begin{aligned}
\texttt{B200} &= \int_{-1}^{-1}\int_{-1}^{-1} \{\chi\}\{\psi\}^T d\eta\, d\xi\,, \\
\texttt{B3001} &= \int_{-1}^{-1}\int_{-1}^{-1} \{\psi\}\{\chi\}\frac{\partial\{\psi\}^T}{\partial\eta} d\eta\, d\xi\,, \quad \texttt{B3002} = \int_{-1}^{-1}\int_{-1}^{-1} \{\psi\}\{\chi\}\frac{\partial\{\psi\}^T}{\partial\xi} d\eta\, d\xi\,, \\
\texttt{B201} &= \int_{-1}^{-1}\int_{-1}^{-1} \{\chi\}\frac{\partial\{\psi\}^T}{\partial\eta} d\eta\, d\xi\,, \quad \texttt{B202} = \int_{-1}^{-1}\int_{-1}^{-1} \{\chi\}\frac{\partial\{\psi\}^T}{\partial\xi} d\eta\, d\xi\,, \\
\texttt{B211} &= \int_{-1}^{-1}\int_{-1}^{-1} \frac{\partial\{\chi\}}{\partial\eta}\frac{\partial\{\psi\}^T}{\partial\eta} d\eta\, d\xi\,, \quad \texttt{B212} = \int_{-1}^{-1}\int_{-1}^{-1} \frac{\partial\{\chi\}}{\partial\eta}\frac{\partial\{\psi\}^T}{\partial\xi} d\eta\, d\xi\,, \\
\texttt{B221} &= \int_{-1}^{-1}\int_{-1}^{-1} \frac{\partial\{\chi\}}{\partial\xi}\frac{\partial\{\psi\}^T}{\partial\eta} d\eta\, d\xi\,, \quad \texttt{B222} = \int_{-1}^{-1}\int_{-1}^{-1} \frac{\partial\{\chi\}}{\partial\xi}\frac{\partial\{\psi\}^T}{\partial\xi} d\eta\, d\xi\,.
\end{aligned} \tag{2.41}$$

The bilinear element employs products of the linear Lagrange interpolation polynomial function (cf. Zienkiewicz [88]),

$$\{\psi\} = \frac{1}{4}\begin{bmatrix} (1-\eta)(1-\xi) \\ (1+\eta)(1-\xi) \\ (1+\eta)(1+\xi) \\ (1-\eta)(1+\xi) \end{bmatrix}. \tag{2.42}$$

Assuming a Galerkin algorithm with $\{\chi\} = \{\psi\}$, the constant matrices on the reference element evaluate to

$$\texttt{B200} = \frac{1}{9}\begin{bmatrix} 4 & 2 & 1 & 2 \\ 2 & 4 & 2 & 1 \\ 1 & 2 & 4 & 2 \\ 2 & 1 & 2 & 4 \end{bmatrix}, \tag{2.43}$$

$$\mathtt{B3001} = \frac{1}{36}\begin{bmatrix} -(6,3,1,2) & (6,3,1,2) & (2,1,1,2) & -(2,1,1,2) \\ -(3,6,2,1) & (3,6,2,1) & (1,2,2,1) & -(1,2,2,1) \\ -(1,2,2,1) & (1,2,2,1) & (1,2,6,3) & -(1,2,6,3) \\ -(2,1,1,2) & (2,1,1,2) & (2,1,3,6) & -(2,1,3,6) \end{bmatrix},$$

$$\mathtt{B3002} = \frac{1}{36}\begin{bmatrix} -(6,2,1,3) & -(2,2,1,1) & (2,2,1,1) & (6,2,1,3) \\ -(2,2,1,1) & -(2,6,3,1) & (2,6,3,1) & (2,2,1,1) \\ -(1,1,2,2) & -(1,3,6,2) & (1,3,6,2) & (1,1,2,2) \\ -(3,1,2,6) & -(1,1,2,2) & (1,1,2,2) & (3,1,2,6) \end{bmatrix}, \tag{2.44}$$

$$\mathtt{B201} = \frac{1}{6}\begin{bmatrix} -2 & 2 & 1 & -1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ -1 & 1 & 2 & -2 \end{bmatrix}, \quad \mathtt{B202} = \frac{1}{6}\begin{bmatrix} 2 & -1 & 1 & 2 \\ -1 & -2 & 2 & 1 \\ -1 & -2 & 2 & 1 \\ 2 & -1 & 1 & 2 \end{bmatrix},$$

$$\mathtt{B211} = \frac{1}{6}\begin{bmatrix} 2 & -2 & -1 & 1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ 1 & -1 & -2 & -2 \end{bmatrix}, \quad \mathtt{B212} = \frac{1}{4}\begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \end{bmatrix}, \tag{2.45}$$

$$\mathtt{B221} = \frac{1}{4}\begin{bmatrix} 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \end{bmatrix}, \quad \mathtt{B222} = \frac{1}{6}\begin{bmatrix} 2 & 1 & -1 & -2 \\ 1 & 2 & -2 & -1 \\ -1 & -2 & 2 & 1 \\ -2 & -1 & 1 & 2 \end{bmatrix}.$$

In case of a rectangular Cartesian grid, metric data evaluation is simplified (and exact), since

$$\mathtt{J} = \begin{bmatrix} \frac{\Delta_x}{2} & 0 \\ 0 & \frac{\Delta_y}{2} \end{bmatrix}, \ \mathtt{J}^{-1} = \frac{1}{|\mathtt{J}|}\begin{bmatrix} \frac{\Delta_y}{2} & 0 \\ 0 & \frac{\Delta_x}{2} \end{bmatrix}, \ |\mathtt{J}| = \frac{\Delta_x \Delta_y}{4}, \text{ hence}$$

$$\mathtt{M200} = \frac{\Delta_x \Delta_y}{4}\,\mathtt{B200},$$

$$\{U_x\}^T\,\mathtt{M300X} = \frac{\Delta_y}{2}\{U_x\}^T\,\mathtt{B3001}, \ \{U_y\}^T\,\mathtt{M300Y} = \frac{\Delta_x}{2}\{U_y\}^T\,\mathtt{B3002}, \tag{2.46}$$

$$\mathtt{M2XX} = \frac{\Delta_y}{\Delta_x}\,\mathtt{B211}, \ \mathtt{M2YY} = \frac{\Delta_x}{\Delta_y}\,\mathtt{B222},$$

$$\mathtt{M2XY} = \mathtt{M2YX} = \mathtt{B212} + \mathtt{B221}.$$

Using the Cartesian metric data shown in (2.46), a finite element equivalent to the finite difference stencil $\mathcal{S}(\cdot)$ may be formed for comparison (although the finite element form is not restricted to Cartesian data). The element matrices are assembled in two dimensions over a patch of four adjacent quadrilateral elements, with the node at the center of the patch representing $Q_{i,j}$. Assuming a 4×4 element matrix with the usual notation of matrix members, its assembled stencil is

$$\mathcal{S}\left(\mathcal{A}_e\left(\begin{bmatrix} (1,1) & (1,2) & (1,3) & (1,4) \\ (2,1) & (2,2) & (2,3) & (2,4) \\ (3,1) & (3,2) & (3,3) & (3,4) \\ (4,1) & (4,2) & (4,3) & (4,4) \end{bmatrix}\right)\right) = \tag{2.47}$$

$$\begin{vmatrix} (2,4) & (2,3)+(1,4) & (1,3) \\ (2,1)+(3,4) & (1,1)+(2,2)+(3,3)+(4,4) & (1,2)+(4,3) \\ (4,2) & (3,2)+(4,1) & (3,1) \end{vmatrix}.$$

In this manner, the stencil of the interpolation matrix M200 in regular Cartesian geometry, including metric data $|\mathtt{J}|_e = \frac{\Delta_x \Delta_y}{4}$, is

$$\mathcal{S}(\,\mathtt{M200}\,Q) = \mathcal{S}(\mathcal{A}_e(|\mathtt{J}|_e\,\mathtt{B200}\,Q)) = \frac{\Delta_x\Delta_y}{36}\begin{vmatrix} Q_{i-1,j+1} & 4Q_{i,j+1} & Q_{i+1,j+1} \\ 4Q_{i-1,j} & 16Q_{i,j} & 4Q_{i+1,j} \\ Q_{i-1,j-1} & 4Q_{i,j-1} & Q_{i+1,j-1} \end{vmatrix}. \tag{2.48}$$

The nodal positions of Q from (2.48) are used hereafter in two-dimensional stencils, with the $Q_{i,j}$ omitted for brevity. Using the 4×4 matrices given in (2.28), the stencils may be formed on a rectangular grid as

$$\mathcal{S}(\,\mathtt{M20X}) = \mathcal{S}(\mathcal{A}_e(\,\mathtt{M20X}_e)) = \mathcal{S}\left(\mathcal{A}_e\left(\frac{\partial y}{\partial \xi_e}\,\mathtt{B201}\right)\right) = \frac{\Delta_y}{12}\begin{vmatrix} -1 & 0 & 1 \\ -4 & 0 & 4 \\ -1 & 0 & 1 \end{vmatrix},$$

$$\mathcal{S}(\,\mathtt{M20Y}) = \mathcal{S}(\mathcal{A}_e(\,\mathtt{M20Y}_e)) = \mathcal{S}\left(\mathcal{A}_e\left(\frac{\partial y}{\partial \xi_e}\,\mathtt{B202}\right)\right) = \frac{\Delta_x}{12}\begin{vmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{vmatrix}, \tag{2.49}$$

$$S(\texttt{M2XX}) = S(\mathcal{A}_e(\texttt{M2XX}_e)) = S\left(\mathcal{A}_e\left(\frac{1}{|J|_e}\frac{\partial y}{\partial \xi_e}\frac{\partial y}{\partial \xi_e}\,\texttt{B211}\right)\right) = \frac{\Delta_y}{6\Delta_x}\begin{vmatrix} -1 & 2 & -1 \\ -4 & 8 & -4 \\ -1 & 2 & -1 \end{vmatrix},$$

$$S(\texttt{M2YY}) = S(\mathcal{A}_e(\texttt{M2YY}_e)) = S\left(\mathcal{A}_e\left(\frac{1}{|J|_e}\frac{\partial x}{\partial \eta_e}\frac{\partial x}{\partial \eta_e}\,\texttt{B222}\right)\right) = \frac{\Delta_x}{6\Delta_y}\begin{vmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{vmatrix},$$

$$S(\texttt{M2XY}) = S(\mathcal{A}_e(\texttt{M2XY}_e)) = S\left(\mathcal{A}_e\left(\frac{1}{|J|_e}\frac{\partial y}{\partial \xi_e}\frac{\partial x}{\partial \eta_e}\,\texttt{B212}\right)\right) = \frac{1}{4}\begin{vmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{vmatrix},$$

$$S(\texttt{M2YX}) = S(\mathcal{A}_e(\texttt{M2YX}_e)) = S\left(\mathcal{A}_e\left(\frac{1}{|J|_e}\frac{\partial y}{\partial \xi_e}\frac{\partial x}{\partial \eta_e}\,\texttt{B221}\right)\right) = \frac{1}{4}\begin{vmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{vmatrix}. \tag{2.50}$$

Substituting these stencils into the matrix weak statement (2.14) yields the recursion relation form of the bilinear Galerkin finite element algorithm for the advection-diffusion equation, (2.1). For comparison with the finite difference form (1.4), the complete rectangular-grid Galerkin stencil is

$$\begin{aligned} S\left(\mathcal{W}^h(\{Q\}^h)\right) = {} & \frac{\Delta_x\Delta_y}{36}\begin{vmatrix} 1 & 4 & 1 \\ 4 & 16 & 4 \\ 1 & 4 & 1 \end{vmatrix}\frac{\{Q\}^{n+1}-\{Q\}^n}{\Delta_t} \\ & + \left[\frac{\bar{U}_x\Delta_y}{12}\begin{vmatrix} -1 & 0 & 1 \\ -4 & 0 & 4 \\ -1 & 0 & 1 \end{vmatrix} + \frac{\bar{U}_y\Delta_x}{12}\begin{vmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{vmatrix}\right. \\ & \left. + \frac{\nu}{6}\frac{\Delta_y}{\Delta_x}\begin{vmatrix} -1 & 2 & -1 \\ -4 & 8 & -4 \\ -1 & 2 & -1 \end{vmatrix} + \frac{\nu}{6}\frac{\Delta_x}{\Delta_y}\begin{vmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{vmatrix}\right]\{Q\}^{n+\theta} \\ & + \frac{\Delta_x\Delta_y}{36}\begin{vmatrix} 1 & 4 & 1 \\ 4 & 16 & 4 \\ 1 & 4 & 1 \end{vmatrix}\{S\}^{n+\theta} = \{0\}. \end{aligned} \tag{2.51}$$

2.2.6 Three-dimensional finite element method

Components of the three-dimensional algorithm mostly follow by inspection from the two-dimensional form.

The trilinear element employs tensor products of the Lagrange polynomial interpolation function set (cf. Zienkiewicz [88]),

$$\{\psi\} = \frac{1}{8}\begin{bmatrix} (1-\eta)(1-\xi)(1-\zeta) \\ (1+\eta)(1-\xi)(1-\zeta) \\ (1+\eta)(1+\xi)(1-\zeta) \\ (1-\eta)(1+\xi)(1-\zeta) \\ (1-\eta)(1-\xi)(1+\zeta) \\ (1+\eta)(1-\xi)(1+\zeta) \\ (1+\eta)(1+\xi)(1+\zeta) \\ (1-\eta)(1+\xi)(1+\zeta) \end{bmatrix}. \tag{2.52}$$

The Jacobian is

$$\mathrm{J} = \begin{bmatrix} \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \zeta} \\ \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \zeta} \end{bmatrix}, \tag{2.53}$$

with inverse

$$\mathrm{J}^{-1} = \frac{1}{|\mathrm{J}|}\begin{bmatrix} \frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \zeta} - \frac{\partial y}{\partial \zeta}\frac{\partial z}{\partial \xi} & -\frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \zeta} + \frac{\partial x}{\partial \zeta}\frac{\partial z}{\partial \xi} & \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \zeta} - \frac{\partial x}{\partial \zeta}\frac{\partial y}{\partial \xi} \\ \frac{\partial y}{\partial \eta}\frac{\partial z}{\partial \zeta} - \frac{\partial y}{\partial \zeta}\frac{\partial z}{\partial \eta} & \frac{\partial x}{\partial \eta}\frac{\partial z}{\partial \zeta} - \frac{\partial x}{\partial \zeta}\frac{\partial z}{\partial \eta} & -\frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \zeta} + \frac{\partial x}{\partial \zeta}\frac{\partial y}{\partial \eta} \\ \frac{\partial y}{\partial \eta}\frac{\partial z}{\partial \xi} - \frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \eta} & -\frac{\partial x}{\partial \eta}\frac{\partial z}{\partial \xi} + \frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \eta} & \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} \end{bmatrix}, \tag{2.54}$$

and determinant

$$\begin{aligned} |\mathrm{J}| = {} & \frac{\partial x}{\partial \eta}\left(\frac{\partial y}{\partial \xi}\frac{\partial z}{\partial \zeta} - \frac{\partial z}{\partial \xi}\frac{\partial y}{\partial \zeta}\right) - \frac{\partial y}{\partial \eta}\left(\frac{\partial x}{\partial \xi}\frac{\partial z}{\partial \zeta} - \frac{\partial z}{\partial \xi}\frac{\partial x}{\partial \zeta}\right) \\ & + \frac{\partial z}{\partial \eta}\left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \zeta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \zeta}\right). \end{aligned} \tag{2.55}$$

On a rectangular Cartesian grid,

$$\mathtt{J} = \begin{bmatrix} \frac{\Delta_x}{2} & 0 & 0 \\ 0 & \frac{\Delta_y}{2} & 0 \\ 0 & 0 & \frac{\Delta_z}{2} \end{bmatrix}, \quad \mathtt{J}^{-1} = \frac{1}{4|\mathtt{J}|} \begin{bmatrix} \Delta_y\Delta_z & 0 & 0 \\ 0 & \Delta_x\Delta_z & 0 \\ 0 & 0 & \Delta_x\Delta_y \end{bmatrix}, \tag{2.56}$$

$$|\mathtt{J}| = \frac{\Delta_x\Delta_y\Delta_z}{8}.$$

A Macsyma® program which produces the three-dimensional linear basis matrices is shown in Appendix III.

2.2.7 Factorized form and stencil matrix representation

Bilinear and trilinear basis forms use interpolation functions which can be represented as products of one-dimensional functions, and the assembled element matrices can be approximately represented by outer products of assembled one-dimensional element matrices (cf. Baker [8], Fletcher [34], and Wait and Mitchell [82]). The two-dimensional matrices are then equal to outer products of a one-dimensional matrix in the y-direction and a one-dimensional matrix in the x-direction, i.e.,

$$\begin{aligned} \mathtt{B200} &= \mathtt{A200L} \otimes \mathtt{A200L}, \\ \mathtt{B201} &= \mathtt{A200L} \otimes \mathtt{A201L}, \\ \mathtt{B202} &= \mathtt{A201L} \otimes \mathtt{A200L}, \\ \mathtt{B211} &= \mathtt{A200L} \otimes \mathtt{A211L}, \\ \mathtt{B212} &= \mathtt{B221} = \mathtt{A201L} \otimes \mathtt{A201L}, \\ \mathtt{B222} &= \mathtt{A222L} \otimes \mathtt{A200L}. \end{aligned} \tag{2.57}$$

The stencils of the two-dimensional matrices on rectangular grids are also outer products of one-dimensional stencils as verified by comparison with (2.48)-(2.50),

$$\begin{aligned} \mathcal{S}(\mathcal{A}_e(|\mathtt{J}|\,\mathtt{B200})) &= \mathcal{S}(\mathcal{A}_e(\Delta_y\,\mathtt{A200L})) \otimes \mathcal{S}(\mathcal{A}_e(\Delta_x\,\mathtt{A200L})) \\ &= \left(\frac{\Delta_y}{6}\right) \begin{vmatrix} \overline{1} \\ 4 \\ \underline{1} \end{vmatrix} \otimes \left(\frac{\Delta_x}{6}\right) \left|\overline{\underline{1 \;\; 4 \;\; 1}}\right|, \end{aligned} \tag{2.58}$$

$$
\begin{aligned}
S\left(\mathcal{A}_e(\frac{\partial y}{\partial \xi}\,\mathtt{B201})\right) &= S(\mathcal{A}_e(\Delta_y\,\mathtt{A200L})) \otimes S(\mathcal{A}_e(\,\mathtt{A201L})) \\
&= \left(\frac{\Delta_y}{6}\right) \left|\overline{\begin{matrix} 1 \\ 4 \\ 1 \end{matrix}}\right| \otimes \left(\frac{1}{2}\right) \left|\overline{\underline{\begin{matrix} -1 & 0 & 1 \end{matrix}}}\right| , \\
S\left(\mathcal{A}_e(\frac{\partial x}{\partial \eta}\,\mathtt{B202})\right) &= S(\mathcal{A}_e(\,\mathtt{A201L})) \otimes S(\mathcal{A}_e(\Delta_x\,\mathtt{A200L})) \\
&= \left(\frac{1}{2}\right) \left|\overline{\begin{matrix} -1 \\ 0 \\ 1 \end{matrix}}\right| \otimes \left(\frac{\Delta_x}{6}\right) \left|\overline{\underline{\begin{matrix} 1 & 4 & 1 \end{matrix}}}\right| ,
\end{aligned}
\tag{2.59}
$$

$$
\begin{aligned}
S\left(\mathcal{A}_e(\frac{1}{|\mathtt{J}|}\frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \xi}\,\mathtt{B211})\right) &= S(\mathcal{A}_e(\Delta_y\,\mathtt{A200L})) \otimes S\left(\mathcal{A}_e(\frac{1}{\Delta_x}\,\mathtt{A211L})\right) \\
&= \left(\frac{\Delta_y}{6}\right) \left|\overline{\begin{matrix} 1 \\ 4 \\ 1 \end{matrix}}\right| \otimes \left(\frac{1}{\Delta_x}\right) \left|\overline{\underline{\begin{matrix} -1 & 2 & -1 \end{matrix}}}\right| , \\
S\left(\mathcal{A}_e(\frac{1}{|\mathtt{J}|}\frac{\partial x}{\partial \eta}\frac{\partial x}{\partial \eta}\,\mathtt{B222})\right) &= S\left(\mathcal{A}_e(\frac{1}{\Delta_x}\,\mathtt{A211L})\right) \otimes S(\mathcal{A}_e(\Delta_y\,\mathtt{A200L})) \\
&= \left(\frac{1}{\Delta_x}\right) \left|\overline{\begin{matrix} -1 \\ 2 \\ -1 \end{matrix}}\right| \otimes \left(\frac{\Delta_y}{6}\right) \left|\overline{\underline{\begin{matrix} 1 & 4 & 1 \end{matrix}}}\right| ,
\end{aligned}
\tag{2.60}
$$

$$
\begin{aligned}
S\left(\mathcal{A}_e(\frac{1}{|\mathtt{J}|}\frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\,\mathtt{B221})\right) &= S(\mathcal{A}_e(\,\mathtt{A201L})) \otimes S(\mathcal{A}_e(\,\mathtt{A201L})) \\
&= \left(\frac{1}{2}\right) \left|\overline{\begin{matrix} -1 \\ 0 \\ 1 \end{matrix}}\right| \otimes \left(\frac{1}{2}\right) \left|\overline{\underline{\begin{matrix} -1 & 0 & 1 \end{matrix}}}\right| , \\
&= S\left(\mathcal{A}_e(\frac{1}{|\mathtt{J}|}\frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta}\,\mathtt{B212})\right) .
\end{aligned}
\tag{2.61}
$$

Three-dimensional trilinear matrices and stencils on regular grids may also be represented by outer products, as shown in Appendix III.

Chapter 3

Analysis methods

3.1 von Neumann frequency analysis

3.1.1 Linear basis elements

A frequency or Fourier analysis is readily applied to a simple model problem such as the one-dimensional linear advection-diffusion equation (1.7). The frequency analysis provides a necessary condition for local stability, and is attributed to von Neumann and Richtmeyer [80], who applied it to the advection equation

$$\mathcal{L}(q) = \frac{\partial q}{\partial t} + u\frac{\partial q}{\partial x} = 0\,. \tag{3.1}$$

Its solution is known in terms of a Fourier expansion,

$$q(x,t) = \sum_{m=-\infty}^{m=\infty} A_m \exp\Big(i\omega_m\,(x-ut)\Big)\,, \tag{3.2}$$

with coefficients A_m chosen to meet unspecified initial conditions. This expansion is to be evaluated only at discrete points $x_j = j\Delta_x$ and $t^n = n\Delta_t$, the nodes and timesteps of the approximate solution, as

$$q(j\Delta_x, n\Delta_t) = \sum_{m=-\infty}^{m=\infty} A_m \exp\Big(i\omega_m\,(j\Delta_x - un\Delta_t)\Big)\,. \tag{3.3}$$

The analysis assumes that a similar expansion exists which interpolates a discrete approximation q^h, to be specified, at the same discrete points,

$$q^h\,(j\Delta_x, n\Delta_t) = \sum_{m=-\infty}^{m=\infty} \hat{A}_m \exp\Big(i\hat{\omega}_m\,(j\Delta_x - U_m n\Delta_t)\Big)\,. \tag{3.4}$$

In the expansion of the approximation, U_m is variable and complex in general, and may differ from the analytical value u in (3.2), a real constant. Since the advection equation is linear, and since initial nodal values of the approximation and the exact solution are assumed identical, the expansion coefficients A_m and $\hat{A}_m$ and frequencies ω_m and $\hat{\omega}_m$ are

the same for the approximation and the exact solution. Thus, in comparing the approximation to the exact solution, a major simplification is possible in that only a single mode of each is considered. Omitting the index m,

$$\begin{aligned} q(j\Delta_x, n\Delta_t) &= \exp\Big(i\omega(j\Delta_x - un\Delta_t)\Big)\,, \\ q^h(j\Delta_x, n\Delta_t) &= \exp\Big(i\omega(j\Delta_x - Un\Delta_t)\Big)\,. \end{aligned} \tag{3.5}$$

Rescaling using the Courant number $C = \frac{u\Delta_t}{\Delta_x}$ and introducing a phase angle $\sigma = \Delta_x\omega$, we find

$$\begin{aligned} q(j\Delta_x, n\Delta_t) &= \exp\Big(i\sigma(j - nC)\Big)\,, \\ q^h(j\Delta_x, n\Delta_t) &= \exp\Big(i\sigma(j - nC\frac{U}{u})\Big)\,. \end{aligned} \tag{3.6}$$

The properties of the approximation are usually described in terms of the amplification factor, the ratio of two solutions adjacent in time. The amplification factor of the exact solution and approximation are

$$\begin{aligned} G &= \frac{q(j\Delta_x, (n+1)\Delta_t)}{q(j\Delta_x, n\Delta_t)} = \exp(-i\sigma C)\,, \\ G^h &= \frac{q^h(j\Delta_x, (n+1)\Delta_t)}{q^h(j\Delta_x, n\Delta_t)} = \exp(-i\sigma C\frac{U}{u})\,. \end{aligned} \tag{3.7}$$

Since C and σ are real, the argument of the analytical amplification factor G is purely imaginary and its modulus is one,

$$|G| = |\exp(-i\sigma C)| = 1\,. \tag{3.8}$$

The modulus of the amplification factor of the to-be-specified approximation G^h is, with notation $\Im(\cdots)$ for imaginary part and $\Re(\cdots)$ for real part,

$$|G^h| = \left|\exp\left(-i\sigma C\frac{U}{u}\right)\right| = \exp\left(\sigma C\Im\left(\frac{U}{u}\right)\right)\,. \tag{3.9}$$

The modulus $|G^h|$ should not exceed one for stability [80], else (3.4) would grow unboundedly.

The phase velocity, the nondimensional local velocity of the wave solution, is defined as the real part of the argument of the amplification factor. The phase velocity is

$-\sigma C$ for the analytical solution and $-\sigma C\Re\left(\frac{U}{u}\right)$ for the approximation. A relative phase velocity of the approximation Φ^h is its phase velocity divided by the analytical value of $-\sigma C$, i.e.,

$$\Phi^h = \Re\left(\frac{U}{u}\right) = \frac{\tan^{-1}\frac{\Im(G^h)}{\Re(G^h)}}{-\sigma C} . \tag{3.10}$$

The amplification factor G^h of the approximation q^h may be readily determined from an assembled equation system such as the one-dimensional (1.13) or the two-dimensional (2.51). Arranging the assembled systems in the form

$$[A]\{Q\}^{n+1} = [B]\{Q\}^n , \tag{3.11}$$

extracting a single interior row and assuming a tridiagonal or linear basis method with constant nodal spacing as an example, the approximate solution may be written as a recursion relation between times $n\Delta_t$ and $(n+1)\Delta_t$ involving nodes $(j\pm 1)\Delta_x$ as

$$a_{j-1}Q_{j-1}^{n+1} + a_jQ_j^{n+1} + a_{j+1}Q_{j+1}^{n+1} = b_{j-1}Q_{j-1}^n + b_jQ_j^n + b_{j+1}Q_{j+1}^n , \tag{3.12}$$

where a, b are the numerical coefficients of the recursion relation. Substituting (3.12) into (3.6) at the same two time levels and assuming u and Δ_x constant over the stencil defines the adjacent values of the approximation,

$$\begin{aligned} Q_{j-1} &= Q(x-\Delta_x) = Q_je^{-\imath\sigma} \\ Q_{j+1} &= Q(x+\Delta_x) = Q_je^{+\imath\sigma} . \end{aligned} \tag{3.13}$$

Substituting the above into the definition of the amplification factor, (3.7), yields a computable form,

$$G^h = \frac{b_{j-1}e^{-\imath\sigma} + b_j + b_{j+1}e^{+\imath\sigma}}{a_{j-1}e^{-\imath\sigma} + a_j + a_{j+1}e^{+\imath\sigma}} = \exp\left(-i\sigma C\frac{U}{u}\right) , \tag{3.14}$$

with modulus

$$|G^h| = \left(\Im(G^h)^2 + \Re(G^h)^2\right)^{\frac{1}{2}} . \tag{3.15}$$

The relative phase velocity Φ^h may be calculated using definition (3.10) with (3.14).

In two dimensions, a similar expansion may be performed for the unsteady linear

advection equation,

$$\mathcal{L}(q) = \frac{\partial q}{\partial t} + u_x \frac{\partial q}{\partial x} + u_y \frac{\partial q}{\partial y} = 0 \,. \tag{3.16}$$

Using directional Courant numbers $C_x = \frac{u_x \Delta_t}{\Delta_x}, C_y = \frac{u_y \Delta_t}{\Delta_y}$, independent directional phase angles $\sigma = \Delta_x \omega$, $\varsigma = \Delta_y \omega_m$, coordinates $x_j = j\Delta_x$, $y_k = k\Delta_y$, time $t^n = n\Delta_t$, and a single Fourier series component as in one dimension, leads to

$$\begin{aligned} q_{l,m}(j\Delta_x, k\Delta_y, n\Delta_t) &= \exp\Big[i\Big(\sigma(j - nC_x) + \varsigma(k - nC_y)\Big)\Big] \,, \\ q^h_{l,m}(j\Delta_x, k\Delta_y, n\Delta_t) &= \exp\Big[i\Big(\sigma(j - nC_x \frac{U_x}{u_x}) + \varsigma(k - nC_y \frac{U_y}{u_y})\Big)\Big] \,. \end{aligned} \tag{3.17}$$

Defining a nine-point recursion relation as the equivalent of the nine-point stencils (1.4) and (2.51), nodal values may be defined in terms of (3.17), for example,

$$Q_{j+1,k+1} = Q(x + \Delta_x, y + \Delta_y) = Q_{j,k} e^{+\imath\sigma} e^{+i\varsigma} \,. \tag{3.18}$$

The two-dimensional amplification factor defined similarly to (3.14) is

$$\begin{aligned} G^h = &\Big[(b_{j-1,k-1}e^{-\imath\sigma} + b_{j,k-1} + b_{j+1,k-1}e^{+\imath\sigma})\, e^{-\imath\varsigma} \\ &+(b_{j-1,k}e^{-\imath\sigma} + b_{j,k} + b_{j+1,k}e^{+\imath\sigma}) \\ &+(b_{j-1,k+1}e^{-\imath\sigma} + b_{j,k+1} + b_{j+1,k+1}e^{+\imath\sigma})\, e^{+\imath\varsigma}\Big] \\ \hline &\Big[(a_{j-1,k-1}e^{-\imath\sigma} + a_{j,k-1} + a_{j+1,k-1}e^{+\imath\sigma})\, e^{-\imath\varsigma} \\ &+(a_{j-1,k}e^{-\imath\sigma} + a_{j,k} + a_{j+1,k}e^{+\imath\sigma}) \\ &+(a_{j-1,k+1}e^{-\imath\sigma} + a_{j,k+1} + a_{j+1,k+1}e^{+\imath\sigma})\, e^{+\imath\varsigma}\Big] \\ =&\exp\left(-i(\sigma C_x \frac{U_x}{u_x} + \varsigma C_y \frac{U_y}{u_y})\right) . \end{aligned} \tag{3.19}$$

The three-dimensional case is the direct extension of (3.19) with another independent phase angle and, using a linear basis, a twenty-seven point recursion relation. This calculation for G^h is obviously cumbersome.

3.1.2 Quadratic and cubic basis elements

The frequency analysis differs for quadratic and cubic basis finite elements in that such elements span more than two nodes and produce different matrix stencils for element-vertex nodes (Q_v) and element-interior nodes (Q_i). Here the analysis follows Wait and Mitchell [82] using the assembled equation system in the form of $[A]\,Q^{n+1} = [B]\,Q^n$. In this example the assembled stencil form of the quadratic basis interpolation matrix `A200Q` is used as $[B]$, and using (3.13) to form the right-hand side vector Q^n, the equation system is

$$[A]\,\{Q\}^{n+1} = \frac{\Delta_x}{30}\begin{bmatrix} -1 & 2 & 8 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 16 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & 8 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 16 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & 8 & 2 & -1 \end{bmatrix}\begin{bmatrix} Q_v e^{-2\imath\sigma} \\ Q_i e^{-\imath\sigma} \\ Q_v \\ Q_i e^{+\imath\sigma} \\ Q_v e^{+2\imath\sigma} \end{bmatrix}. \tag{3.20}$$

Expanding the right-hand side matrix product yields

$$[A]\,\{Q\}^{n+1} = \frac{\Delta_x}{30}\begin{bmatrix} [(8 - e^{-2\imath\sigma} - e^{+2\imath\sigma})\,Q_v + 2(e^{-\imath\sigma} + e^{+\imath\sigma})\,Q_i]\,e^{-2\imath\sigma} \\ [16\,Q_i + 2(e^{-\imath\sigma} + e^{+\imath\sigma})\,Q_v]\,e^{-2\imath\sigma} \\ (8 - e^{-2\imath\sigma} - e^{+2\imath\sigma})\,Q_v + 2(e^{-\imath\sigma} + e^{+\imath\sigma})\,Q_i \\ 16\,Q_i + 2(e^{-\imath\sigma} + e^{+\imath\sigma})\,Q_v \\ [(8 - e^{-2\imath\sigma} - e^{+2\imath\sigma})\,Q_v + 2(e^{-\imath\sigma} + e^{+\imath\sigma})\,Q_i]\,e^{+2\imath\sigma} \end{bmatrix}. \tag{3.21}$$

Extracting any two adjacent rows of (3.21) yields a 2×2 matrix recursion relation,

$$[A]\,\{Q\}^{n+1} = \frac{\Delta_x}{30}\begin{bmatrix} (8 - e^{-2\imath\sigma} - e^{+2\imath\sigma}) & 2(e^{-\imath\sigma} + e^{+\imath\sigma}) \\ 2(e^{-\imath\sigma} + e^{+\imath\sigma}) & 16 \end{bmatrix}\begin{bmatrix} Q_v \\ Q_i \end{bmatrix}. \tag{3.22}$$

The left-hand side matrix $[A]$ in (3.22) may be arranged in a similar form, and the generalization of (3.14) is a matrix equation $G^h = [A]^{-1}[B]$ with modulus equal to the spectral radius of the matrix, $|G^h| = \rho([A]^{-1}[B])$. In this form, stability requires that the magnitude of the maximum eigenvalue should be bounded by one. In the extension to cubic or higher degree elements, a Galerkin finite element algorithm of basis degree k will generally produce a $k \times k$ amplification factor matrix whose eigenvalues are the amplification factors.

3.2 Modified equation analysis

The "modified equation" analysis of Warming and Hyett [83] was extended for this research and applied to finite element methods. The analysis is applied to a scalar linear differential equation without source terms in the form

$$\mathcal{L}(q) = \frac{\partial q}{\partial t} + F\left(\frac{\partial q}{\partial x}, \frac{\partial^2 q}{\partial x^2}, \ldots\right) = 0. \tag{3.23}$$

The differential equation (3.23) is assumed to have a discrete approximation in one dimension,

$$\mathcal{L}^h(Q) = \mathcal{L}^h\left(Q_{j-1}^n, Q_j^n, Q_{j+1}^n, Q_{j-1}^{n+1}, Q_j^{n+1}, Q_{j+1}^{n+1}\right). \tag{3.24}$$

The discrete nodal values (or some nodally accurate interpolant as in the von Neumann analysis) are assumed to have Taylor expansions in time and space around a central value $Q_j{}^n$,

$$\begin{aligned} Q^{n+1} &= Q^n + \Delta_t \frac{\partial Q^n}{\partial t} + \frac{\Delta_t^2}{2}\frac{\partial^2 Q^n}{\partial t^2} + \cdots, \\ Q_{j+1} &= Q_j + \Delta_x \frac{\partial Q_j}{\partial x} + \frac{\Delta_x^2}{2}\frac{\partial^2 Q_j}{\partial x^2} + \cdots, \\ Q_{j-1} &= Q_j - \Delta_x \frac{\partial Q_j}{\partial x} + \frac{\Delta_x^2}{2}\frac{\partial^2 Q_j}{\partial x^2} - \cdots. \end{aligned} \tag{3.25}$$

Extensions are obvious to larger bandwidth such as Q_{j-2} or multiple timesteps such as Q^{n-1}. The series of derivatives of the dependent variable may then be substituted for each discrete nodal value in (3.24), converting the approximation to a differential "modified" equation which is not equal to the original differential equation (3.23), but is equal in the limit to the differential equation actually solved by the approximation (3.24). Differences between the original differential equation and the modified equation are error terms in the approximation, and the form of the error terms may be used to make deductions about accuracy and stability of the approximation.

Warming and Hyett formed the modified equation for several example difference methods, but their procedure requires that the difference method be already known and explicit in time. The procedure may be generalized to a symbolic form by assuming an implicit or explicit difference method which is at least first order accurate in time. Assume that $O(\Delta_t) = O(\Delta_x)$ and expand the Taylor series (3.25) to fourth order combined in time and space. The coefficients of the analytical derivatives are calculated from the expansion;

here they are shown symbolically,

$$\begin{aligned}\mathcal{L}^h(Q) =& f_{01}\Delta_x\frac{\partial Q}{\partial x} + f_{02}\Delta_x^2\frac{\partial^2 Q}{\partial x^2} + f_{03}\Delta_x^3\frac{\partial^3 Q}{\partial x^3} + f_{04}\Delta_x^4\frac{\partial^4 Q}{\partial x^4} \\ &+ \Delta_t\left(f_{10} + f_{11}\Delta_x\frac{\partial}{\partial x} + f_{12}\Delta_x^2\frac{\partial^2}{\partial x^2} + f_{13}\Delta_x^3\frac{\partial^3}{\partial x^3}\right)\frac{\partial Q}{\partial t} \\ &+ \Delta_t^2\left(f_{20} + f_{21}\Delta_x\frac{\partial}{\partial x} + f_{22}\Delta_x^2\frac{\partial^2}{\partial x^2}\right)\frac{\partial^2 Q}{\partial t^2} \\ &+ \Delta_t^3\left(f_{30} + f_{31}\Delta_x\frac{\partial}{\partial x}\right)\frac{\partial^3 Q}{\partial t^3} + f_{40}\Delta_t^4\frac{\partial^4 Q}{\partial t^4} + \cdots = 0\,.\end{aligned} \tag{3.26}$$

In (3.26) the coefficients f_{ij} refer to the i-th derivative in time and the j-th derivative in space. Recalling the assumption of first-order time accuracy, the coefficient f_{10} must equal unity, and (3.26) may be rearranged to put $f_{10}\Delta_t\frac{\partial Q}{\partial t}$ on the left-hand side, and all other terms on the right-hand side. In addition, defining operators $X = \Delta_x\frac{\partial}{\partial x}$ and $T = \Delta_t\frac{\partial}{\partial t}$ for brevity yields

$$\begin{aligned}T =& -f_{01}X - f_{02}X^2 - f_{03}X^3 - f_{04}X^4 - (f_{11}X + f_{12}X^2 + f_{13}X^3 + f_{14}X^4)T \\ &- (f_{20} + f_{21}X + f_{22}X^2 + f_{23}X^3 + f_{24}X^4)T^2 - (f_{30} + f_{31}X)T^3 - f_{40}T^4 + \cdots .\end{aligned} \tag{3.27}$$

The right-hand side in (3.27) may be repeatedly substituted for T into itself until powers of the error terms are of negligible order. Details of the process are shown in Appendix IV. The fourth order solution, converting all time derivatives on the right-hand

side to space derivatives, is

$$
\begin{aligned}
T = &- f_{01}X + \left(-f_{02} + f_{01}f_{11} - f_{20}f_{01}^2\right)X^2 \\
&+ \Big(-f_{03} + f_{01}f_{12} + (f_{11} - 2f_{20}f_{01})f_{02} - f_{01}^2 f_{21} - f_{01}f_{11}^2 \\
&+ 3f_{20}f_{01}^2 f_{11} + (f_{30} - 2f_{20}^2)f_{01}^3\Big)X^3 \\
&+ \Big(-f_{04} + f_{01}f_{13} + (f_{11} - 2f_{20}f_{01})f_{03} - f_{01}^2 f_{22} + (f_{02} - 2f_{01}f_{11} + 3f_{20}f_{01}^2)f_{12} \\
&- f_{20}f_{02}^2 + \Big(-2f_{01}f_{21} - f_{11}^2 + 6f_{20}f_{01}f_{11} + (3f_{30} - 4f_{20}^2)f_{01}^2\Big)f_{02} \\
&+ f_{01}^3 f_{31} + 3f_{01}^2 f_{11}f_{21} + f_{01}f_{11}^3 - 5f_{20}f_{01}^2 f_{11}^2 \\
&+ (8f_{20}^2 - 2f_{30})f_{01}^3 f_{11} + (f_{20}f_{30} - f_{40})f_{01}^4\Big)X^4 + \cdots .
\end{aligned}
\tag{3.28}
$$

As an example of the application, consider the explicit Lax-Wendroff finite difference method in Courant number form with x-index j. The form is obtained by multiplying (1.8) through by Δ_t and clearing $C = \frac{U\Delta_t}{\Delta_x}$ to get

$$
\hat{\mathcal{L}}^h(Q) = (Q^{n+1} - Q^n) + \frac{C}{2}(Q_{j+1}^n - Q_{j-1}^n) - \frac{C^2}{2}(Q_{j+1}^n - 2Q_j^n + Q_{j-1}^n) . \tag{3.29}
$$

Substituting the Taylor series (3.25) and equating to (3.27) yields the coefficients

$$
\begin{aligned}
&f_{20} = 1/2 , \; f_{30} = 1/6 , \; f_{40} = 1/24 , \; f_{01} = C , \; f_{11} = 0 , \; f_{21} = 0 , \; f_{31} = 0 , \\
&f_{02} = -C^2/2 , \; f_{12} = 0 , \; f_{22} = 0 , \; f_{03} = C/6 , \; f_{13} = 0 , \; f_{04} = -C^2/24 .
\end{aligned}
\tag{3.30}
$$

Substituting these coefficients into (3.28) matches Warming and Hyett's result, with error terms arranged on the right-hand side in ascending order as

$$
T + CX = \frac{C(C^2-1)}{6}X^3 + \frac{C^2(C^2-1)}{8}X^4 + \frac{C(6C^4 - 5C^2 - 1)}{120}X^5 + \cdots . \tag{3.31}
$$

Warming and Hyett's stability analysis, an extension of the von Neumann Fourier analysis, showed that the leading even-order derivative in the error should have sign $-((-1)^{\frac{n}{2}})$ for order n, or positive for order 2 and negative for order 4. This criterion is satisfied in the example for all $C < 1$.

The form of (3.28) allows stability calculations for implicit methods, for example the fourth-order accurate TWS finite element method with diffusion ν (4.9), The stencil is

$$\begin{aligned}\hat{\mathcal{L}}^h(Q) = &\Big(\frac{2+C^2}{12}(Q_{j+1}^{n+1}-Q_{j+1}^{n})+\frac{8-C^2}{12}(Q_j^{n+1}-Q_j^{n})+\frac{2+C^2}{12}(Q_{j-1}^{n+1}-Q_{j-1}^{n})\Big)\\ &+\frac{C^2}{4}(Q_{j+1}^{n+\frac{1}{2}}-Q_{j-1}^{n+\frac{1}{2}})-\frac{\nu}{2}(Q_{j+1}^{n+\frac{1}{2}}-2Q_j^{n+\frac{1}{2}}+Q_{j-1}^{n+\frac{1}{2}}),\end{aligned} \tag{3.32}$$

and the associated modified equation shown in Appendix IV is

$$T+CX=-\nu X^2-\frac{\nu(2C^2-1)}{12}X^4-\frac{C(6\nu^2-C^4-5C^2-4)}{720}X^5+\cdots, \tag{3.33}$$

hence the fourth order accuracy (for $\nu=0$) is correctly predicted.

While the Fourier analysis is tractable in one dimension, the modified equation analysis is more easily computed in two and three dimensions. In two dimensions, the operator $f_{01}X$ in (3.27) is replaced by $f_{010}X+f_{001}Y$ with $Y=\Delta_y\frac{\partial}{\partial y}$, and $f_{02}X^2$ by $f_{020}X^2+$

$+f_{011}XY+f_{002}Y^2$, etc. The expansion to third order, replacing (3.28), is

$$
\begin{aligned}
T = &-f_{010}X+\left(-f_{010}^2f_{200}+f_{010}f_{110}-f_{020}\right)X^2\\
&+\Big(f_{010}^3f_{300}-f_{010}^2f_{210}-2f_{010}^3f_{200}^2+(3f_{010}^2f_{110}-2f_{010}f_{020})f_{200}\\
&+f_{010}f_{120}-f_{010}f_{110}^2+f_{020}f_{110}-f_{030}\Big)X^3\\
&-f_{001}Y+\left(-f_{001}^2f_{200}+f_{001}f_{101}-f_{002}\right)Y^2\\
&+\Big(f_{001}^3f_{300}-f_{001}^2f_{201}-2f_{001}^3f_{200}^2+\left(3f_{001}^2f_{101}-2f_{001}f_{002}\right)f_{200}\\
&+f_{001}f_{102}-f_{001}f_{101}^2+f_{002}f_{101}-f_{003}\Big)Y^3\\
&+\left(-2f_{001}f_{010}f_{200}+f_{001}f_{110}+f_{010}f_{101}-f_{011}\right)XY\\
&+\Big(3f_{001}f_{010}^2f_{300}-2f_{001}f_{010}f_{210}-f_{010}^2f_{201}-6f_{001}f_{010}^2f_{200}^2\\
&+\left(6f_{001}f_{010}f_{110}+3f_{010}^2f_{101}-2f_{001}f_{020}-2f_{010}f_{011}\right)f_{200}\\
&+f_{001}f_{120}+f_{010}f_{111}-f_{001}f_{110}^2+(f_{011}-2f_{010}f_{101})f_{110}+f_{020}f_{101}-f_{021}\Big)X^2Y\\
&+\Big(3f_{001}^2f_{010}f_{300}-f_{001}^2f_{210}-2f_{001}f_{010}f_{201}-6f_{001}^2f_{010}f_{200}^2\\
&+\left(3f_{001}^2f_{110}+6f_{001}f_{010}f_{101}-2f_{001}f_{011}-2f_{002}f_{010}\right)f_{200}+f_{001}f_{111}\\
&+(f_{002}-2f_{001}f_{101})f_{110}+f_{010}f_{102}-f_{010}f_{101}^2+f_{011}f_{101}-f_{012}\Big)XY^2+\cdots .
\end{aligned}
\tag{3.34}
$$

3.3 Diffusion direction analysis

Layton [54],[55] formulated a type of modified equation analysis for inviscid steady flow with artificial diffusion, using an assembled two-dimensional finite difference stencil of the form

$$
S\left(\mathcal{L}^h(Q)\right)=\begin{vmatrix} -a_{nw} & -a_n & -a_{ne}\\ -a_w & a_p & -a_e\\ -a_{sw} & -a_s & -a_{se}\end{vmatrix}. \tag{3.35}
$$

The convection terms form the antisymmetric part of the stencil and the diffusion terms form the symmetric part of the stencil. In two dimensions, the effective diffusion terms may be separated into xx, yy, and xy components using symmetry of the stencil about the vertical, horizontal, and diagonal planes respectively. Hence,

$$\begin{aligned} D_{xx} &= a_{ne} + a_{se} + a_{nw} + a_{sw} + a_e + a_w \,, \\ D_{yy} &= a_{ne} + a_{se} + a_{nw} + a_{sw} + a_n + a_s \,, \\ D_{xy} &= a_{ne} + a_{sw} - a_{nw} - a_{se} \,. \end{aligned} \tag{3.36}$$

As will be shown in Chapter 5 when using Warming and Hyett's method, the modified equation analysis predicts that artificial diffusion should be streamline diffusion, i.e.,

$$\begin{aligned} D_{xx} &= \beta U_x{}^2 \,, \\ D_{yy} &= \beta U_y{}^2 \,, \\ D_{xy} &= \beta U_x U_y \,. \end{aligned} \tag{3.37}$$

Layton equivalently shows that the direction of the diffusion is

$$tan(2\theta^h) = \frac{2D_{xy}}{D_{xx} - D_{yy}} \,, \tag{3.38}$$

and should equal the direction of the velocity given by

$$tan(2\theta) = \frac{2U_x U_y}{U_x{}^2 - U_y{}^2} \,. \tag{3.39}$$

3.4 Matrix analysis for stability

The properties of the assembled matrices may also be analyzed for stability. Given an assembled matrix system,

$$[A]\{Q\}^{n+1} = [B]Q^n \,, \tag{3.40}$$

which is equivalent, if $[A]$ is invertible, to

$$\{Q\}^{n+1} = ([A]^{-1}[B])\{Q\}^n \,. \tag{3.41}$$

The stability condition, similar to the von Neumann condition, is that the solution will be stable [76],[4] if the spectral radius $\rho([A]^{-1}[B]) \leq 1$. If periodic boundary conditions are used, the stability analysis will be equivalent to the von Neumann form [48], thus will not depend on the order of the matrix, and the smallest possible assembled matrix may be used for convenience. For example, assembling the Galerkin FEM for the advection-diffusion equation, (2.14), setting $\nu = 0$ and $\theta = \frac{1}{2}$, using linear basis matrices (2.18) and periodic boundary conditions in one dimension over two elements, yields

$$\begin{vmatrix} \frac{2}{3} & \frac{1}{6}+\frac{C}{2} & \frac{1}{6}-\frac{C}{2} \\ \frac{1}{6}-\frac{C}{2} & \frac{2}{3} & \frac{1}{6}+\frac{C}{2} \\ \frac{1}{6}+\frac{C}{2} & \frac{1}{6}-\frac{C}{2} & \frac{2}{3} \end{vmatrix} \{Q\}^{n+1} = \begin{vmatrix} \frac{2}{3} & \frac{1}{6}-\frac{C}{2} & \frac{1}{6}+\frac{C}{2} \\ \frac{1}{6}+\frac{C}{2} & \frac{2}{3} & \frac{1}{6}-\frac{C}{2} \\ \frac{1}{6}-\frac{C}{2} & \frac{1}{6}+\frac{C}{2} & \frac{2}{3} \end{vmatrix} \{Q\}^{n} . \tag{3.42}$$

Using Macsyma® to form $([A]^{-1}[B])$ and to find the eigenvalues yields

$$\{Q\}^{n+1} = \frac{1}{4+3C^2} \begin{vmatrix} (2+C)(2-C) & -2C(2-C) & 2C(2+C) \\ 2C(2+C) & (2+C)(2-C) & -2C(2-C) \\ -2C(2-C) & 2C(2+C) & (2+C)(2-C) \end{vmatrix} \{Q\}^{n} , \tag{3.43}$$

$$\rho([A]^{-1}[B]) = \left(1 , \frac{(4-3C^2) \pm 4\sqrt{3}Ci}{(4+3C^2)} \right) .$$

The magnitude of each eigenvalue is one, so the matrix analysis, like the von Neumann analysis, predicts stability for any Courant number. However, at $C = \pm 2$, this trapezoidal rule method produces zero diagonals and becomes singular and unusable, a result not easily obtainable from the von Neumann analysis. Conversely, the frequency information from the von Neumann analysis is not easily obtained from the matrix analysis. The matrix analysis can also incorporate the effect of boundary conditions [4], but the results, although potentially more useful, will not be equivalent to the von Neumann analysis.

3.5 Matrix analysis for monotonicity

An important measure of solution quality is an oscillation-free or monotone solution. A sufficient condition for monotonicity is that the assembled system matrix $[A]$ from

(3.40) have the M-matrix property [78],[5]. An M-matrix is diagonally dominant with positive diagonals, negative off-diagonals, is invertible, and has real positive eigenvalues.

The monotonicity analysis is primarily used for steady-state problems [48] and the right-hand side matrix $[B]$ in (3.40) is considered data. For example, using the stencil of a centered FDM for the steady-state advection-diffusion equation with ν to be determined,

$$\boxed{-\frac{U}{2\Delta_x}-\frac{\nu}{\Delta_x^2} \quad 1+\frac{2\nu}{\Delta_x^2} \quad +\frac{U}{2\Delta_x}-\frac{\nu}{\Delta_x^2}}\ \{Q\}=0\,. \tag{3.44}$$

The necessary monotonicity criterion is for the last element of the stencil to be negative with $U \geq 0$ or the first element to be negative with $U \leq 0$. If (3.44) is not modified, monotonicity requires that the cell Peclet number $\frac{U\Delta_x}{\nu} \leq 2$, which constitutes a very strict limit.

3.6 Norms for measurement of approximation quality

Several different norms will be used for measuring the quality of an approximation. For verification problems where the analytical solution is known, discrete norms such as the L^2 or L^∞ norms are convenient [41]. Measured at the N nodes,

$$\begin{aligned} ||q||_2^2 &= \sum_{n=1}^{n=N} \left(q^h(j\Delta_x, n\Delta_t) - q(x,t)\right)^2\,, \\ ||q||_\infty &= \max_{n=1}^{n=N} |q^h(j\Delta_x, n\Delta_t) - q(x,t)|\,. \end{aligned} \tag{3.45}$$

The typical integral finite element norm is the energy norm [8],

$$||q||_E^2 = \frac{1}{2}\int_\Omega q^h \mathcal{L}_d^h(q^h)\,d\Omega\,, \tag{3.46}$$

where $\mathcal{L}_d^h(q^h)$ is only the second-order part of the differential approximation, i.e. $\nu\nabla\cdot\nabla q^h$ for (1.1). The approximation energy norm by analogy to (2.11) is

$$||q^h||_E^2 = \frac{1}{2}\int_{\Omega^h} -\nu\{\psi\}^T\{Q\}\nabla\cdot\nabla\{\psi\}^T\{Q\}\,d\Omega^h\,. \tag{3.47}$$

Substituting, assuming space dependence on the basis function only, integrating by parts,

and substituting the boundary condition,

$$\begin{aligned}||q^h||_E^2 =& \frac{\nu}{2}\int_{\Omega^h} \nabla\{\psi\}^T\{Q\}\cdot\nabla\{\psi\}^T\{Q\}\,d\Omega^h - \frac{\nu}{2}\int_{\Omega^h}\{\psi\}^T\{Q\}\vec{n}\cdot\nabla\{\psi\}^T\{Q\}\,d\Omega^h \\ =& \frac{\nu}{2}\{Q\}^T\int_{\Omega^h}\nabla\{\psi\}\cdot\nabla\{\psi\}^T\,d\Omega^h\{Q\} - \frac{\nu}{2}\{Q\}^T\int_{\Omega^h}\{\psi\}\{\psi\}^T\,d\Omega^h\{G\}\end{aligned} \tag{3.48}$$

with symbolic form $\frac{\nu}{2}\{Q\}^T$ M2KK$\{Q\} - \frac{\nu}{2}\{Q\}^T$ M200$\{G\}$.

Franca and Haghighi [35] developed a summed energy norm of velocity for the two-dimensional Navier-Stokes equations,

$$||q||_E^2 = -\frac{\nu}{2}\int_{\Omega^h} u_i\frac{\partial}{\partial x_j}\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\,d\Omega^h\,. \tag{3.49}$$

The Galerkin approximation after integration by parts, produces

$$||q||_E^2 = \frac{\nu}{2}\int_{\Omega^h}\frac{\partial\{\psi\}^T}{\partial x_j}\{U_i\}\left(\frac{\partial\{\psi\}^T}{\partial x_j}\{U_i\}+\frac{\partial\{\psi\}^T}{\partial x_i}\{U_j\}\right)\,d\Omega^h\,, \tag{3.50}$$

with expanded symbolic form

$$\begin{aligned}||q^h||_E^2 =& \frac{\nu}{2}\{U_x\}^T(\,\texttt{M2XX}+\texttt{M2YY})\{U_x\}+\frac{\nu}{2}\{U_y\}^T(\,\texttt{M2XX}+\texttt{M2YY})\{U_y\} \\ &+\frac{\nu}{2}\,\texttt{M2XY}\{U_x\}\,.\end{aligned} \tag{3.51}$$

Norms of scalar variables such as temperature, pressure, and pressure correction may be evaluated similarly to (3.48).

An asymptotic error analysis based on energy norms may be used when an analytical solution is unknown. Assuming the solution is convergent as an elliptic PDE and that h is sufficiently small [8], the energy norm of the error is bounded by the error in the energy norm, in turn bounded by a function of the mesh size, i.e.,

$$||\varepsilon^h||_E = ||q||_E - ||q^h||_E \le ch^{2k}\,, \tag{3.52}$$

where the three energy norms indicated are of the unknown error, the unknown analytical solution, and the known approximate solution respectively. The mesh measure is h, k is the basis degree, and c is a constant independent of h. If the approximation is computed on a sufficiently fine mesh, the inequality approaches an equality, hence the unknowns c and

$||q||_E$ may be cancelled out using two approximate solutions on different meshes. Convergence according to (3.52) may then be verified using a third solution. If the solutions do not exhibit asymptotic convergence, an error or an insufficiently refined mesh is indicated. Therefore, in contrast to FDM and FVM methods, the FEM supplies a useful measure of convergence without requiring reference to a known solution.

Chapter 4

Modified finite element methods for advection-dominated problems

4.1 Taylor finite element methods

Lax and Wendroff [53] formulated a higher-order correction to an explicit finite difference approximation in 1960. It was applied to a finite volume method by Stone and Brian [73] in 1963 and Harten and Tal-Ezer [46] in 1981. Similar corrections were made to a finite element method by van Genuchten and Gray [77] in 1978; the method became generally known as the Taylor-Galerkin method of Donea [28] in 1984. The methodology was extended via adjustable parameters as the Taylor weak statement of Baker and Kim [9] in 1987.

The frequency analysis of von Neumann and Richtmeyer [80] in 1950 was originally derived to determine stability, and was extended by Stone and Brian [73] to determine the phase velocity and dispersion error for a finite difference method. The frequency analysis was applied to various forms of the FEM by Vichnevetsky and De Schutter [79] in 1975, Gray and Pinder [42] in 1976, Raymond and Garder [65] in 1976, Gresho et al. [45] in 1978, Chin et al. [19] in 1979, Wait and Mitchell [82] in 1985, Cathers and O'Connor [16] in 1985, Baker and Kim [9], Gresho and Lee [44] in 1987, and Donea et al. [29] in 1987. Vichnevetsky and De Schutter [79], Raymond and Garder [65], and Baker and Kim [9] each attempted some optimization of parameters to reduce dispersion error in a linear basis FEM. Vichnevetsky and De Schutter [79], Gray and Pinder [42], Gresho et al. [45], Cathers and O'Connor [16], Wait and Mitchell [82], and Gresho and Lee [44] each applied various analyses to quadratic basis finite elements. Vichnevetsky and De Schutter [79] and Gresho et al. [45] analyzed Hermite cubic basis finite elements.

4.1.1 Original Lax-Wendroff

The original Lax and Wendroff [53] higher-order correction was applied to an explicit finite difference model of the one-dimensional unsteady linear advection equation (1.7). Using a Taylor series in time, the typical Euler time finite difference was augmented

by higher order terms,

$$\frac{\partial q}{\partial t}^n = \frac{q^{n+1} - q^n}{\Delta_t} - \frac{\Delta_t}{2}\frac{\partial^2 q}{\partial t^2}^n - \frac{\Delta_t^2}{6}\frac{\partial^3 t}{\partial t^3}^n - \frac{\Delta_t^3}{24}\frac{\partial^4 q}{\partial t^4}^n + O(\Delta_t^4) . \tag{4.1}$$

The second and higher order time derivatives were equated to space derivatives using the PDE; substituting in the explicit discrete approximation yields

$$\begin{aligned} \hat{\mathcal{L}}^h(Q) = \frac{Q^{n+1} - Q^n}{\Delta_t} + \frac{U}{2\Delta_x}(Q^n_{j+1} - Q^n_{j-1}) \\ - \frac{U^2\Delta_t}{2\Delta_x^2}(Q^n_{j+1} - 2Q^n_j + Q^n_{j-1}) = O(\Delta_x^2, \Delta_t^2) . \end{aligned} \tag{4.2}$$

The correction term (with coefficient U^2) appears as artificial diffusion. Lax and Wendroff found the approximation (4.2) to be both more accurate and more stable than the previous explicit approximation (1.8).

4.1.2 Donea Taylor-Galerkin algorithm

The Lax-Wendroff method was derived in the context of finite difference methods and is essentially independent of space differencing technique. Donea [28],[29] first applied Lax-Wendroff to a finite element weak statement yielding the "Taylor-Galerkin" algorithm, although the application of Lax-Wendroff method to finite elements appears to originate with van Genuchten and Gray [77]. Donea developed variations using explicit time-stepping, "Euler Taylor-Galerkin" or ETG, and trapezoidal time-stepping, "Crank-Nicolson Taylor-Galerkin" or CNTG. The Euler Taylor-Galerkin method resembles the original Lax-Wendroff method (4.2), as discretized using finite elements, although explicit time-stepping is not commonly used with finite elements since the non-diagonal time-term interpolation matrix results in the algebraic matrix of an implicit form regardless of time-stepping method.

4.1.3 Lax-Wendroff Taylor Weak Statement algorithm

A generalized Lax-Wendroff method is developed here for the linear advection equation (1.7). Assuming sufficient continuity and semi-discretizing in time, the forward

time Taylor series (1.9) is used along with the companion backward series to obtain

$$q^n = q^{n+1} - \Delta_t \frac{\partial q}{\partial t}^{n+1} + \frac{\Delta_t^2}{2}\frac{\partial^2 q}{\partial t^2}^{n+1} - \frac{\Delta_t^3}{6}\frac{\partial^3 t}{\partial t^3}^{n+1} + \frac{\Delta_t^4}{24}\frac{\partial^4 q}{\partial t^4}^{n+1} - O(\Delta_t^5)\,. \tag{4.3}$$

The forward and backward series are also formed for a partial step at $q^{n+\theta}$ and equated, i.e.,

$$\begin{aligned} q^{n+\theta} &= q^{n+1} - (1-\theta)\Delta_t \frac{\partial q}{\partial t}^{n+1} + \frac{(1-\theta)^2\Delta_t^2}{2}\frac{\partial^2 q}{\partial t^2}^{n+1} - \frac{(1-\theta)^3\Delta_t^3}{6}\frac{\partial^3 t}{\partial t^3}^{n+1} \\ &\quad + \frac{(1-\theta)^4\Delta_t^4}{24}\frac{\partial^4 q}{\partial t^4}^{n+1} - O(\Delta_t^5) \\ &= q^n + \theta\Delta_t \frac{\partial q}{\partial t}^{n} + \frac{\theta^2\Delta_t^2}{2}\frac{\partial^2 q}{\partial t^2}^{n} + \frac{\theta^3\Delta_t^3}{6}\frac{\partial^3 t}{\partial t^3}^{n} + \frac{\theta^4\Delta_t^4}{24}\frac{\partial^4 q}{\partial t^4}^{n} + O(\Delta_t^5)\,. \end{aligned} \tag{4.4}$$

A similar forward and backward series are formed for a partial step at $q^{n+1-\theta}$ and equated to obtain

$$\begin{aligned} q^{n+1-\theta} &= q^n + (1-\theta)\Delta_t \frac{\partial q}{\partial t}^{n} + \frac{(1-\theta)^2\Delta_t^2}{2}\frac{\partial^2 q}{\partial t^2}^{n} + \frac{(1-\theta)^3\Delta_t^3}{6}\frac{\partial^3 t}{\partial t^3}^{n} \\ &\quad + \frac{(1-\theta)^4\Delta_t^4}{24}\frac{\partial^4 q}{\partial t^4}^{n} + O(\Delta_t^5) \\ &= q^{n+1} - \theta\Delta_t \frac{\partial q}{\partial t}^{n+1} + \frac{\theta^2\Delta_t^2}{2}\frac{\partial^2 q}{\partial t^2}^{n+1} - \frac{\theta^3\Delta_t^3}{6}\frac{\partial^3 t}{\partial t^3}^{n+1} + \frac{\theta^4\Delta_t^4}{24}\frac{\partial^4 q}{\partial t^4}^{n+1} - O(\Delta_t^5)\,. \end{aligned} \tag{4.5}$$

Taking a linear combination of the four series (1.9),(4.3),(4.4),(4.5), and eliminating third order terms for $\frac{1}{3} \le \theta \le \frac{2}{3}$,

$$\begin{aligned} \hat{\mathcal{L}}(q) &= (q^{n+1} - q^n) - \Delta_t\left(\theta\frac{\partial q}{\partial t}^{n+1} + (1-\theta)\frac{\partial q}{\partial t}^{n}\right) \\ &\quad + \frac{\Delta_t^2}{6}\left((3\theta-1)\frac{\partial^2 q}{\partial t^2}^{n+1} + (3\theta-2)\frac{\partial^2 q}{\partial t^2}^{n}\right) \\ &\quad - \frac{\Delta_t^4}{72}\left((\theta^2+2\theta-1)\frac{\partial^4 q}{\partial t^4}^{n+1} - (\theta^2-4\theta+2)\frac{\partial^4 q}{\partial t^4}^{n}\right) + O(\Delta_t^5) = 0\,. \end{aligned} \tag{4.6}$$

The first two terms of (4.6) are the standard θ-implicit approximation to (1.7), while the remaining terms are corrections resulting from the replacement of truncation error terms. With $\theta = \frac{1}{3}$, $\frac{1}{2}$, or $\frac{2}{3}$, the resulting correction is formally equivalent to various Padé compact

difference approximations to e^{-t} [64]. Corrected equations for $\theta = 0$ or $\theta = 1$ may be developed directly from (1.9) or (4.3).

Assuming sufficient continuity, the time derivatives in (4.6) may be replaced with space derivatives using (1.11) expressed for advection only, i.e. neglecting diffusion terms, which yields

$$\begin{aligned}\hat{\mathcal{L}}(q) = q^{n+1} - q^n + u\Delta_t\left(\theta\frac{\partial q}{\partial x}^{n+1} + (1-\theta)\frac{\partial q}{\partial x}^{n}\right) \\ + \frac{u^2\Delta_t^2}{6}\left((3\theta-1)\frac{\partial^2 q}{\partial x^2}^{n+1} + (3\theta-2)\frac{\partial^2 q}{\partial x^2}^{n}\right) \\ - \frac{u^4\Delta_t^4}{72}\left((\theta^2+2\theta-1)\frac{\partial^4 q}{\partial x^4}^{n+1} - (\theta^2-4\theta+2)\frac{\partial^4 q}{\partial x^4}^{n}\right) + O(\Delta_t^5) = 0\,.\end{aligned} \tag{4.7}$$

For $\theta = \frac{1}{2}$, (4.7) simplifies to

$$\begin{aligned}\hat{\mathcal{L}}(q) = q^{n+1} - q^n + \frac{u\Delta_t}{2}\left(\frac{\partial q}{\partial x}^{n+1} + \frac{\partial q}{\partial x}^{n}\right) \\ + \frac{u^2\Delta_t^2}{12}\left(\frac{\partial^2 q}{\partial x^2}^{n+1} - \frac{\partial^2 q}{\partial x^2}^{n}\right) - \frac{u^4\Delta_t^4}{288}\left(\frac{\partial^4 q}{\partial x^4}^{n+1} - \frac{\partial^4 q}{\partial x^4}^{n}\right) + O(\Delta_t^5) = 0\,.\end{aligned} \tag{4.8}$$

Donea [28]) developed (4.8) to second order as the CNTG method. The development is questionable as Donea [28] assumed in (28) that $2(q^{n+1} - q^n) = \Delta_t(\frac{\partial q}{\partial t}^{n+1} - \frac{\partial q}{\partial t}^{n})$ which only holds to first order using (1.9) and (4.3). However, the development does produce the equivalent of (4.8) to second order for $\theta = \frac{1}{2}$.

The resultant one-dimensional linear finite element fourth-order accurate Galerkin weak statement of (4.8) is

$$\begin{aligned}\hat{\mathcal{W}}^h(\{Q\}) = \left(\Delta_x\,\mathtt{A200L} - \frac{U^2\Delta_t^2}{12\Delta_x}\,\mathtt{A211L}\right)\left(\{Q\}^{n+1} - \{Q\}^n\right) \\ + \frac{U\Delta_t}{2}\,\mathtt{A201L}\left(\{Q\}^{n+1} + \{Q\}^n\right) = \{0\}\,.\end{aligned} \tag{4.9}$$

4.1.4 Baker and Kim algorithm

Baker and Kim [9] generalized the algorithms of Lax-Wendroff and Donea as the

"Taylor weak statement" (TWS) method for a one-dimensional flux vector equation system. The TWS algorithm method is extended here for multi-dimensional linear advection,

$$\mathcal{L}(q) = \frac{\partial q}{\partial t} + u_j \frac{\partial q}{\partial x_j} = 0 \,. \tag{4.10}$$

Higher derivatives are taken in the manner of Lax and Wendroff, i.e.,

$$\begin{aligned} \frac{\partial^2 q}{\partial t^2} &= -u_j \frac{\partial}{\partial x_j}\frac{\partial q}{\partial t} = u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial q}{\partial x_j}\right) , \\ \frac{\partial^3 q}{\partial t^3} &= u_k \frac{\partial}{\partial x_k}(u_j \frac{\partial}{\partial x_j}\frac{\partial q}{\partial t}) = -u_l \frac{\partial}{\partial x_l}\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial q}{\partial x_j}\right)\right) . \end{aligned} \tag{4.11}$$

Weighted combinations of the higher-order derivatives are taken with $O(1)$ weight coefficients $-\alpha+\beta \approx 1$ and $\gamma-\mu \approx 1$ to produce

$$\begin{aligned} \frac{\partial^2 q}{\partial t^2} &= 2\alpha\left(u_j \frac{\partial}{\partial x_j}\frac{\partial q}{\partial t}\right) + 2\beta\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial q}{\partial x_j}\right)\right) , \\ \frac{\partial^3 q}{\partial t^3} &= 6\gamma\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial}{\partial x_j}\frac{\partial q}{\partial t}\right)\right) + 6\mu\left[u_l \frac{\partial}{\partial x_l}\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial q}{\partial x_j}\right)\right)\right] . \end{aligned} \tag{4.12}$$

Substituting the Euler time derivative approximation $\frac{\partial q}{\partial t} \approx \frac{q^{n+1}-q^n}{\Delta_t}$ into (4.12) and using the θ-implicit finite difference time approximation (1.3) yields the TWS modified equation for 4.10,

$$\begin{aligned} \hat{\mathcal{L}}(q) = \frac{q^{n+1}-q^n}{\Delta_t} + u_j \frac{\partial q^{n+\theta}}{\partial x_j} - \Delta_t \left[\alpha\left(u_j \frac{\partial}{\partial x_j}\frac{q^{n+1}-q^n}{\Delta_t}\right) + \beta\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial q^{n+\theta}}{\partial x_j}\right)\right)\right] \\ - \Delta_t^2 \left[\gamma\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial}{\partial x_j}\frac{q^{n+1}-q^n}{\Delta_t}\right)\right) + \mu\left[u_l \frac{\partial}{\partial x_l}\left(u_k \frac{\partial}{\partial x_k}\left(u_j \frac{\partial q^{n+\theta}}{\partial x_j}\right)\right)\right]\right] = 0 . \end{aligned} \tag{4.13}$$

An additional discrete form in the manner of (2.11) may be introduced for the last term in (4.13) with the coefficient μ, since the typical linear basis FEM does not support three differentiations. Using a Green-Gauss theorem, neglecting boundary integrals, and inter-

changing tensor indices to facilitate eventual simplification,

$$
\begin{aligned}
&\mu \int_{\Omega^h} \{\chi\}\{\psi\}^T\{U_l\}\frac{\partial}{\partial x_l}\left(\{\psi\}^T\{U_k\}\frac{\partial}{\partial x_k}\{\psi\}^T\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\}\right) d\Omega^h \\
&= -\mu \int_{\Omega^h} \frac{\partial}{\partial x_l}\left(\{\chi\}\{\psi\}^T\{U_l\}\right)\{\psi\}^T\{U_k\}\frac{\partial}{\partial x_k}\{\psi\}^T\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} d\Omega^h \\
&= -\mu\{U_k\}^T \int_{\Omega^h} \{\psi\}\left(\frac{\partial\{\chi\}}{\partial x_l}\{\psi\}^T + \{\chi\}\frac{\partial\{\psi\}^T}{\partial x_l}\right)\{U_l\}\frac{\partial}{\partial x_k}\{\psi\}^T d\Omega^h \{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} \\
&= -\mu\{U_k\}^T\{U_l\}^T \int_{\Omega^h}\left(\{\psi\}\{\psi\}\frac{\partial\{\chi\}}{\partial x_l} + \frac{\partial\{\psi\}}{\partial x_l}\{\psi\}\{\chi\}\right)\frac{\partial}{\partial x_k}\{\psi\}^T d\Omega^h \{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} \\
&= -\mu\{U_k\}^T\{U_l\}^T\,(\,\mathtt{M400LK} + \mathtt{M4L00K})\,\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} \\
&= -\mu\{U_l\}^T\{U_k\}^T\,(\,\mathtt{M400LK} + \mathtt{M40L0K})\,\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} \\
&= -\mu\{U_k\}^T\{U_l\}^T\,(\,\mathtt{M400KL} + \mathtt{M40K0L})\,\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\}\,.
\end{aligned}
\tag{4.14}
$$

The term $\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\}$ requires finite difference approximation when implemented using linear basis elements. The terms in (4.13) with coefficients γ and β are discretized similarly to (4.14), replacing $\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\}$ by $\{Q\}$. Multiplying (4.13) through by Δ_t, the fully discrete form of (4.13) for variable velocity is

$$
\begin{aligned}
\hat{\mathcal{W}}^h(\{Q\}) = &\Big[\,\mathtt{M200} + \alpha\Delta_t\{U_j\}^T\,\mathtt{M300J} \\
&+ \gamma\Delta_t^2\{U_j\}^T\{U_k\}^T(\,\mathtt{M400JK} + \mathtt{M40J0K})\Big]\,(\{Q\}^{n+1} - \{Q\}^n) \\
&+ \Delta_t\Big[\{U_k\}^T\,\mathtt{M300K} + \beta\Delta_t\{U_j\}^T\{U_k\}^T(\,\mathtt{M400JK} + \mathtt{M40J0K})\Big]\,\{Q\}^{n+\theta} \\
&+ \mu\Delta_t^3\{U_k\}^T\{U_l\}^T(\,\mathtt{M400KL} + \mathtt{M40K0L})\,\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} = \{0\}\,.
\end{aligned}
\tag{4.15}
$$

Assuming the velocities $\{U_j\}$, etc., may be represented by element averages $\bar{U}_j$, etc.,

results in a simplified form,

$$\hat{\mathcal{W}}^h(\{Q\}) = \Big(\texttt{M200} + \alpha\Delta_t\bar{U}_j\,\texttt{M2J0} + \gamma\Delta_t^2\bar{U}_j\bar{U}_k\,\texttt{M2JK}\Big)\,(\{Q\}^{n+1} - \{Q\}^n)$$
$$+\Delta_t\Big(\bar{U}_k\,\texttt{M20K} + \beta\Delta_t\bar{U}_j\bar{U}_k\,\texttt{M2JK}\Big)\,\{Q\}^{n+\theta} + \mu\Delta_t^3\bar{U}_k\bar{U}_l\,\texttt{M2KL}\,\{U_j\frac{\partial Q^{n+\theta}}{\partial x_j}\} = \{0\}\,. \tag{4.16}$$

Here α,β,γ, and μ are taken as arbitrary coefficients to be defined. The one-dimensional linear finite element implementation is

$$\hat{\mathcal{W}}^h(\{Q\}) = \Big(\Delta_x\,\texttt{A200L} + \alpha\bar{U}\Delta_t\,\texttt{A210L} + \frac{\gamma\bar{U}^2\Delta_t^2}{\Delta_x}\,\texttt{A211L}\Big)\,(\{Q\}^{n+1} - \{Q\}^n)$$
$$+\Big(\bar{U}\Delta_t\,\texttt{A201L} + \frac{\beta\bar{U}^2\Delta_t^2}{\Delta_x}\,\texttt{A211L}\Big)\,\{Q\}^{n+\theta} + \frac{\mu\bar{U}^2\Delta_t^3}{\Delta_x}\,\texttt{A211L}\frac{\partial}{\partial x}\{(UQ^{n+\theta})\} = \{0\}\,. \tag{4.17}$$

Donea's CNTG method, (4.9), is equivalent to $\theta = \frac{1}{2}, \alpha = \beta = \mu = 0\,,\ \gamma = -\frac{1}{12}$ in (4.17). The Courant number form, dividing (4.17) through by Δ_x, further simplifies the one-dimensional TWS implementation appearance to

$$\hat{\mathcal{W}}^h(\{Q\}) = \Big(\texttt{A200L} + \alpha C\,\texttt{A210L} + \gamma C^2\,\texttt{A211L}\Big)\,(\{Q\}^{n+1} - \{Q\}^n)$$
$$+\Big(C\,\texttt{A201L} + \beta C^2\,\texttt{A211L}\Big)\,\{Q\}^{n+\theta} + \mu C^2\,\texttt{A211L}\Delta_t\frac{\partial}{\partial x}\{(UQ^{n+\theta})\} = \{0\}\,. \tag{4.18}$$

4.2 Upwind and Petrov-Galerkin methods

4.2.1 SUPG method

The "streamline upwind Petrov-Galerkin (SUPG)" method of Hughes and Brooks [12] results from an artificial diffusion term with coefficient $\hat{k}$ added to a steady advection-diffusion equation (2.1). Using a non-dimensional form with Peclet number $\mathrm{Pe} = \frac{UL}{\alpha}$, and including an artificial diffusion term with coefficient $\hat{k}$, the SUPG modified form is

$$\hat{\mathcal{L}}(q) = \vec{u}(\vec{x})\cdot\nabla q - \Big(\frac{1}{\mathrm{Pe}} + \hat{k}\Big)\nabla\cdot\nabla q = 0\,. \tag{4.19}$$

The Green-Gauss weak form of (4.19), neglecting boundary terms, is

$$\hat{\mathcal{W}}(q) = \int_\Omega \left[w u_j \frac{\partial q}{\partial x_j} + \left(\frac{1}{\mathrm{Pe}} + \hat{k} \right) \frac{\partial w}{\partial x_j} \frac{\partial q}{\partial x_j} \right] d\Omega = 0 \,. \tag{4.20}$$

The artificial diffusion coefficient $\hat{k}$ is then replaced by the streamline-oriented tensor diffusion form $\bar{k} u_j u_k$ with coefficient $\bar{k}$. The diffusion term is combined with the convection term in (4.20) as

$$\hat{\mathcal{W}}(q) = \int_\Omega \left[\left(w + \bar{k} u_k \frac{\partial w}{\partial x_k} \right) u_j \frac{\partial q}{\partial x_j} + \frac{1}{\mathrm{Pe}} \frac{\partial w}{\partial x_j} \frac{\partial q}{\partial x_j} \right] d\Omega = 0 \,, \tag{4.21}$$

where continuity was used to eliminate one diffusion term containing $\frac{\partial u_j}{\partial x_j}$. Hence, (4.21) may be interpreted as a Petrov-Galerkin method with the Galerkin test function w replaced by $\hat{w} = w + \bar{k} u_k \frac{\partial w}{\partial x_k}$. This Petrov-Galerkin test function may then also be applied to (initially missing) transient and source terms, yielding

$$\hat{\mathcal{W}}(q) = \int_\Omega \left[\left(w + \bar{k} u_k \frac{\partial w}{\partial x_k} \right) \left(\frac{\partial q}{\partial t} + u_j \frac{\partial q}{\partial x_j} - s \right) + \frac{1}{\mathrm{Pe}} \frac{\partial w}{\partial x_j} \frac{\partial q}{\partial x_j} \right] d\Omega = 0 \,. \tag{4.22}$$

The resulting finite element weak statement, using element-averaged velocity, is

$$\begin{aligned} \hat{\mathcal{W}}^h(\{Q\}) = & \left(\mathtt{M200} + \bar{k} \bar{U}_j \, \mathtt{M2J0} \right) \left(\frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} - \{S\}^{n+\theta} \right) \\ & + \left(\bar{U}_j \, \mathtt{M20J} + \bar{k} \bar{U}_k \bar{U}_j \, \mathtt{M2KJ} + \frac{1}{\mathrm{Pe}} \, \mathtt{M2JJ} \right) \{Q\}^{n+\theta} = \{0\} \,. \end{aligned} \tag{4.23}$$

Therefore, the SUPG algorithm (without physical diffusion) is contained in the TWS form (4.16) by the definition $\alpha \Delta_t = \beta \Delta_t = \bar{k}$ and $\gamma = \mu = 0$. Tezduyar and Ganjoo [75] defined the diffusivity constant $\bar{k}$ as

$$\bar{k} = \min(1, \frac{\mathrm{Pe}}{3}) \left(\sum_n \left| u_k \frac{\partial w}{\partial x_k} \right| \right)^{-1} , \tag{4.24}$$

which, on a rectangular mesh, simplifies to

$$\bar{k} = \min(1, \frac{\mathrm{Pe}}{3}) \frac{\Delta_x \Delta_y}{2(|\{U_x\}|\Delta_y + |\{U_y\}|\Delta_x)} \,. \tag{4.25}$$

Swaminathan and Voller [74] set the diffusivity constant on a rectangular mesh to

$$\bar{k} = \min(1, \frac{\mathrm{Pe}}{3}) \frac{|\{U_x\}|\Delta_x + |\{U_y\}|\Delta_y}{2({U_x}^2 + {U_y}^2)} , \tag{4.26}$$

which is identical to (4.24) only for one-dimensional flow.

Swaminathan and Voller [74] show an example calculation for large-Peclet steady-state convection with $U_x = U_y$ on a square mesh. The stencil of the assembled Petrov-Galerkin convection term is

$$\begin{aligned} \mathcal{S}\Big(\bar{U}_k \,\mathtt{M20K}\Big)\{Q\} &= \mathcal{S}\left(\frac{\bar{U}_x\Delta_y}{2}\,\mathtt{B201B} + \frac{\bar{U}_y\Delta_x}{2}\,\mathtt{B202B}\right)\{Q\} \\ &= \frac{\bar{U}_x\Delta_x}{12}\begin{vmatrix} 0 & 4 & 2 \\ -4 & 0 & 4 \\ -2 & -4 & 0 \end{vmatrix}\{Q\} , \end{aligned} \tag{4.27}$$

and the assembled SUPG term, expressed separately as a diffusion term, is

$$\begin{aligned} &\frac{\Delta_x}{2|\bar{U}_x|}\mathcal{S}\Big(\bar{U}_k\bar{U}_j\,\mathtt{M2KJ}\Big)\{Q\} \\ &= \frac{\Delta_x}{2|\bar{U}_x|}\mathcal{S}\Big(\bar{U}_x^2\,\mathtt{M211} + \bar{U}_x\bar{U}_y(\,\mathtt{M212} + \,\mathtt{M221}) + \bar{U}_y^2\,\mathtt{M222}\Big)\{Q\} \\ &= \frac{\Delta_x}{2|\bar{U}_x|}\mathcal{S}\Big(\bar{U}_x^2\frac{\Delta_y}{\Delta_x}\,\mathtt{B211L} + \bar{U}_x\bar{U}_y(\,\mathtt{B212} + \,\mathtt{B221}) + \bar{U}_y^2\frac{\Delta_x}{\Delta_y}\,\mathtt{B222}\Big)\{Q\} \\ &= \frac{\bar{U}_x\Delta_x}{12}\begin{vmatrix} 1 & -2 & -5 \\ -2 & 16 & -2 \\ -5 & -2 & 1 \end{vmatrix}\{Q\} . \end{aligned} \tag{4.28}$$

For example, a large-Peclet SUPG weak statement for the one-dimensional unsteady convection equation (without physical diffusion) is

$$\begin{aligned} \hat{\mathcal{W}}^h(\{Q\}) = &\Big(\,\mathtt{M200} + \frac{\Delta_x}{2|U|}\bar{U}\,\mathtt{M2X0}\Big)\frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} \\ &+ \Big(\bar{U}\,\mathtt{M20X} + \frac{\Delta_x}{2|U|}\bar{U}^2\,\mathtt{M2XX}\Big)\{Q\}^{n+\theta} = \{0\} . \end{aligned} \tag{4.29}$$

Its linear basis assembled stencil is

$$\begin{aligned} S\left(\hat{W}^h(\{Q\})\right) = &\left(\frac{\Delta_x}{6}\left|\underline{\overline{1\ \ 4\ \ 1}}\right| + \frac{\Delta_x \bar{U}}{4|U|}\left|\underline{\overline{1\ \ 4\ \ 1}}\right|\right)\frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} \\ &+\left(\frac{\bar{U}}{2}\left|\underline{\overline{-1\ \ 0\ \ 1}}\right| + \frac{\Delta_x \bar{U}^2}{2|U|}\left|\underline{\overline{-1\ \ 2\ \ -1}}\right|\right)\{Q\}^{n+\theta}, \end{aligned} \tag{4.30}$$

which coincides with the SUPG construction of Bouloutas and Celia [10] and is similar to the TWS form (4.16) with $\alpha\Delta_t = \beta\Delta_t = \frac{\Delta_x}{2|U|}$ and $\gamma = \mu = 0$.

4.2.2 Exponential Petrov-Galerkin method of Chee and Kwon

Monotone Petrov-Galerkin methods using an exponential test function are derived by Ahues and Telias [1], Sheu et al. [72], and Chee and Kwon [18]. Following the latter in two dimensions, the bilinear test function set $N_i(\eta,\xi)$ (2.42) is rewritten as

$$\{N(\eta,\xi)\} = \begin{bmatrix} f_\eta f_\xi \\ (1-f_\eta) f_\xi \\ (1-f_\eta)(1-f_\xi) \\ f_\eta (1-f_\xi) \end{bmatrix}, \tag{4.31}$$

for the tensor product functions $f_\eta = \frac{1-\eta}{2}$ and $f_\xi = \frac{1-\xi}{2}$. Chee and Kwon [18] replaced these by $f_\eta = \frac{\exp(\mathrm{Pe}_{\Delta_x}\frac{1-\eta}{2})-1}{\exp(\mathrm{Pe}_{\Delta_x})-1}$ and $f_\xi = \frac{\exp(\mathrm{Pe}_{\Delta_y}\frac{1-\xi}{2})-1}{\exp(\mathrm{Pe}_{\Delta_y})-1}$, for directional element Peclet numbers defined as $\mathrm{Pe}_{\Delta_x} = \frac{U_x \Delta_x}{\mathrm{Pe}}$ and $\mathrm{Pe}_{\Delta_y} = \frac{U_y \Delta_y}{\mathrm{Pe}}$. The resulting modified finite element convection terms are denoted here as B2[·]E. In the η direction, the convection term is

$$\mathtt{B201E} = \frac{1}{a_{00}} \begin{bmatrix} -a_{11}b_{11} & a_{11}b_{11} & a_{11}b_{13} & -a_{11}b_{13} \\ -a_{21}b_{11} & a_{21}b_{11} & a_{21}b_{13} & -a_{21}b_{13} \\ -a_{21}b_{31} & a_{21}b_{31} & a_{21}b_{33} & -a_{21}b_{33} \\ -a_{11}b_{31} & a_{11}b_{31} & a_{11}b_{33} & -a_{11}b_{33} \end{bmatrix}, \tag{4.32}$$

with

$$
\begin{aligned}
a_{00} &= \mathrm{Pe}_{\Delta_x}\left(\exp(\mathrm{Pe}_{\Delta_x})-1\right)\mathrm{Pe}_{\Delta_y}^2\left(\exp(\mathrm{Pe}_{\Delta_y})-1\right),\\
a_{11} &= \exp(\mathrm{Pe}_{\Delta_x})-\mathrm{Pe}_{\Delta_x}-1,\\
a_{21} &= (\mathrm{Pe}_{\Delta_x}-1)\exp(\mathrm{Pe}_{\Delta_x})+1,\\
b_{11} &= 2(\mathrm{Pe}_{\Delta_y}-1)\exp(\mathrm{Pe}_{\Delta_y})-\mathrm{Pe}_{\Delta_y}^2+2,\\
b_{13} &= 2\exp(\mathrm{Pe}_{\Delta_y})-\mathrm{Pe}_{\Delta_y}{}^2-2\mathrm{Pe}_{\Delta_y}-2,\\
b_{31} &= (\mathrm{Pe}_{\Delta_y}^2-2\mathrm{Pe}_{\Delta_y}+2)\exp(\mathrm{Pe}_{\Delta_y})-2,\\
b_{33} &= (\mathrm{Pe}_{\Delta_y}^2-2)\exp(\mathrm{Pe}_{\Delta_y})+2\mathrm{Pe}_{\Delta_y}+2.
\end{aligned}
\tag{4.33}
$$

In the ξ direction, the companion finite element convection matrix is

$$
\mathrm{B202E}=\frac{1}{a_{00}}\begin{bmatrix}
-a_{11}b_{11} & -a_{11}b_{13} & a_{11}b_{11} & a_{11}b_{13}\\
-a_{11}b_{31} & -a_{11}b_{33} & a_{11}b_{31} & a_{11}b_{33}\\
-a_{21}b_{31} & -a_{21}b_{33} & a_{21}b_{31} & a_{21}b_{33}\\
-a_{21}b_{11} & -a_{21}b_{13} & a_{21}b_{11} & a_{21}b_{13}
\end{bmatrix},
\tag{4.34}
$$

with Δ_x and Δ_y interchanged in definition of the coefficients shown in (4.33).

For comparison, an unmodified bilinear Galerkin finite element algorithm on a rectangular Cartesian grid is assembled for the steady portion of (2.51), yielding

$$
\begin{aligned}
S\left(\mathcal{W}^h(\{Q\})\right) &= S\left(\frac{\bar{U}_x\Delta_y}{2}\,\mathrm{B201B}+\frac{\bar{U}_y\Delta_x}{2}\,\mathrm{B202B}\right)+\frac{1}{\mathrm{Pe}}S\left(\frac{\Delta_y}{\Delta_x}\,\mathrm{B211B}+\frac{\Delta_x}{\Delta_y}\,\mathrm{B222B}\right)\\
&= \frac{U_x\Delta_y}{12}\begin{vmatrix}-1 & 0 & 1\\ -4 & 0 & 4\\ -1 & 0 & 1\end{vmatrix}+\frac{U_y\Delta_x}{12}\begin{vmatrix}1 & 4 & 1\\ 0 & 0 & 0\\ -1 & -4 & -1\end{vmatrix}\\
&\quad+\frac{\Delta_y}{\mathrm{Pe}\,6\Delta_x}\begin{vmatrix}-1 & 2 & -1\\ -4 & 8 & -4\\ -1 & 2 & -1\end{vmatrix}+\frac{\Delta_x}{\mathrm{Pe}\,6\Delta_y}\begin{vmatrix}-1 & -4 & -1\\ 2 & 8 & 2\\ -1 & -4 & -1\end{vmatrix}.
\end{aligned}
\tag{4.35}
$$

Substituting unit velocities and element Peclet numbers $\mathrm{Pe}_{\Delta_x} = \mathrm{Pe}_{\Delta_y} = 10$, (4.35) becomes

$$\mathcal{S}\left(\mathcal{W}^h(\{Q\})\right) = \frac{1}{30}\begin{vmatrix} -1 & 9 & 4 \\ -11 & 8 & 9 \\ -6 & -11 & -1 \end{vmatrix}. \tag{4.36}$$

The Chee and Kwon algorithm with the same data produces

$$\mathcal{S}\left(\hat{\mathcal{W}}^h(\{Q\})\right) = \frac{1}{30}\begin{vmatrix} -0.04 & 0.98 & -0.94 \\ -6.82 & 35.84 & 0.98 \\ -23.14 & -6.82 & -0.04 \end{vmatrix}. \tag{4.37}$$

Since some off-diagonal coefficients are positive, the Chee and Kwon algorithm will not produce a monotone solution at these cell Peclet numbers, but the GWS algorithm will not produce a monotone solution at a cell Peclet number greater than unity (cf. calculation of 4.62).

4.3 Hierarchical basis methods

4.3.1 Static condensation

Hierarchical basis methods incorporate additional element-level higher-degree functions in the finite element basis construction, e.g. (2.42). "Static condensation" [8], [15], [69] , is a type of Gauss elimination used to remove degrees of freedom associated with those higher-degree functions, thus avoiding undesirable growth in the size of the solution matrix statement, while presumably retaining the benefits of higher-order accuracy.

Static condensation is typically used to eliminate element-internal nodes from a finite element mesh, since, because of the compact interpolation property, the matrix solution of the internal nodes will depend only on element-boundary nodes. This is illustrated for one-dimensional quadratic basis elements via a two-element example for a diffusion problem adapted from [8]. The governing equation is $\frac{\partial}{\partial x}\left(\nu\frac{\partial q}{\partial x}\right) = 0$ with a Neumann boundary condition and a variable conductivity which is interpolated on the element. The

resulting quadratic basis element matrices are, for two elements with conductivity k_1 and k_2,

$$\nu_1 \, \texttt{A3001Q}_1 = \begin{bmatrix} 15.5 & -18 & 2.5 \\ -18 & 40 & -22 \\ 2.5 & -22 & 19.5 \end{bmatrix},$$
$$\nu_2 \, \texttt{A3001Q}_2 = \begin{bmatrix} 22.5 & -26 & 3.5 \\ -26 & 56 & -30 \\ 3.5 & -30 & 26.5 \end{bmatrix}. \tag{4.38}$$

The assembled form, incorporating the Neumann boundary conditions at the first node and clearing a constant multiplier, is

$$S(\nu \, \texttt{A3001Q})\{Q\} = \left|\begin{array}{rrrrr} 15.5 & -18 & 2.5 & 0 & 0 \\ -18 & 40 & -22 & 0 & 0 \\ 2.5 & -22 & 42 & -26 & 3.5 \\ 0 & 0 & -26 & 56 & -30 \\ 0 & 0 & 3.5 & -30 & 26.5 \end{array}\right| \{Q\} = \left|\begin{array}{r} 9000 \\ 0 \\ 0 \\ 0 \\ 306.85282 \end{array}\right|, \tag{4.39}$$

with solution $\{Q\} = \{999.9703, \quad 776.9841, \quad 594.541, \quad 440.4233, \quad 306.8527\}^T$.

For the statically condensed solution, the even-numbered rows in (4.39) represent the two element-internal nodes. Since Gauss row elimination operations may be used on a matrix without affecting its solution, the even-numbered rows can be reduced out of the matrix without affecting the solution for the odd-numbered rows. In this example the row eliminations are $\frac{18}{40}$ times the second row onto the first row, $\frac{22}{40}$ times the second row onto the third row, $\frac{26}{56}$ times the fourth row onto the third row, and $\frac{30}{56}$ times the fourth row onto the fifth row, producing a matrix statement equivalent to (4.39) in the form

$$S(\nu \, \texttt{A3001Q})\{Q\} = \left|\begin{array}{rrrrr} 13.4 & 0 & -7.4 & 0 & 0 \\ -18 & 40 & -22 & 0 & 0 \\ -7.4 & 0 & 17.86 & 0 & -10.43 \\ 0 & 0 & -26 & 56 & -30 \\ 0 & 0 & -10.43 & 0 & 10.43 \end{array}\right| \{Q\} = \left|\begin{array}{r} 9000 \\ 0 \\ 0 \\ 0 \\ 306.85282 \end{array}\right|. \tag{4.40}$$

The odd-row solution then does not depend on the even-row data; removing the even rows

produces the reduced rank system for the odd rows only as

$$S(\text{v A3001Q})\{\hat{Q}\} = \left|\begin{array}{rrr} 13.4 & -7.4 & 0 \\ -7.4 & 17.86 & -10.43 \\ 0 & -10.43 & 10.43 \end{array}\right| \{\hat{Q}\} = \left|\begin{array}{r} 9000 \\ 0 \\ 306.85282 \end{array}\right|, \qquad (4.41)$$

with identical odd-row solution $\{\hat{Q}\} = \{999.9703, \quad 594.541, \quad 306.8527\}^T$. The even-node solution can be recovered (if necessary) by substituting this solution into (4.39). In the condensation process, the global matrix (4.39) is thus in this example reduced from pentadiagonal to tridiagonal and is also reduced by almost half of its degrees of freedom, yielding an efficient method with quadratic basis accuracy.

Since this particular example assembly uses only the matrix A3001Q, static condensation could also be performed at the element level. Normalizing the even-numbered rows of the assembled matrix requires multiplication by the inverse of the assembled diagonal terms, which are 40 and 56 in (4.39). Those terms coincide with the (2,2) terms of the element matrices (4.38) and the element matrices themselves can be condensed as in [8].

In general, these assembled diagonal terms would be summed from the (2,2) terms of several element matrices, for example in the advection-diffusion equation, U_x A201Q $-$ $\frac{1}{\text{Pe}}$ A211Q. Since the inverse of sums $\frac{1}{a+b}$ is not equal to the sum of inverses $\frac{1}{a} + \frac{1}{b}$, static condensation cannot be performed individually on element matrices which are to be summed, unless the element matrices are first summed over all contributions. This observation appears contrary to the procedure of [69], to be discussed. In addition, some matrices such as the convection matrix A201Q cannot be statically condensed individually since the diagonal terms are zero when assembled on a uniform grid.

A more general example presenting this issue is the non-dimensional one-dimensional quadratic basis FEM for transient convection, e.g.(2.14) with $\theta = \frac{1}{2}$,

$$\begin{aligned} &S\Big(\text{A200Q} + \theta C\ \text{A201Q}\Big)\{Q\}^{n+1} = \\ &S\Big(\text{A200Q} + (1-\theta) C\ \text{A201Q}\Big)\{Q\}^{n}. \end{aligned} \qquad (4.42)$$

Its interior assembled form, showing one end node as the second row surrounded by two

mid-element node rows, is

$$\left|\begin{array}{ccccc} 1-5C & 8 & 1+5C & 0 & 0 \\ -1+\frac{5C}{2} & 2(1-5C) & 8 & 2(1+5C) & -1-\frac{5C}{2} \\ 0 & 0 & 1-5C & 8 & 1+5C \end{array}\right| \{Q\}^{n+1} =$$
$$\left|\begin{array}{ccccc} 1+5C & 8 & 1-5C & 0 & 0 \\ -1-\frac{5C}{2} & 2(1+5C) & 8 & 2(1-5C) & -1+\frac{5C}{2} \\ 0 & 0 & 1+5C & 8 & 1-5C \end{array}\right| \{Q\}^{n} . \tag{4.43}$$

Using the Gauss elimination procedure illustrated above, its reduced form is

$$\left|\begin{array}{ccccc} 1-5C & 8 & 1+5C & 0 & 0 \\ -1+4C-5C^2 & 0 & 2(3+5C^2) & 0 & -1-4C-5C^2 \\ 0 & 0 & 1-5C & 8 & 1+5C \end{array}\right| \{Q\}^{n+1} =$$
$$\left|\begin{array}{ccccc} 1+5C & 8 & 1-5C & 0 & 0 \\ -1-2C+5C^2 & 16C & 2(3-5C^2) & -16C & -1+2C+5C^2 \\ 0 & 0 & 1+5C & 8 & 1-5C \end{array}\right| \{Q\}^{n} . \tag{4.44}$$

While the (2,2) and (2,4) entries on the left-hand side of the reduced form have been zeroed out, thus decoupling end and mid-element nodes in Q^{n+1}, the corresponding entries in Q^n on the right-hand side ($\pm 16C$) cannot be zeroed simultaneously. Therefore the mid-element node data must be available in the computation and the eventual computational benefit is reduced. An additional difficulty in two dimensions is that biquadratic elements have only one element-internal node suitable for elimination, compared to eight element-boundary nodes.

4.3.2 Bubble function method

The "hierarchical bubble function" method, cf. Oden and Carey [15], is illustrated here for the two-dimensional steady advection-diffusion equation,

$$\mathbf{U}\cdot\nabla q - \frac{1}{\mathrm{Pe}}\nabla\cdot\nabla q = 0 . \tag{4.45}$$

The Galerkin finite element weak statement approximation, neglecting boundary integrals,

is

$$\mathcal{W}^h(\{Q\}) = \int_\Omega \{\psi\}\mathbf{U}\cdot\nabla\{\psi\}^T d\tau\,\{Q\} + \frac{1}{\mathrm{Pe}}\int_\Omega \nabla\{\psi\}\cdot\nabla\{\psi\}^T d\tau\,\{Q\} = \{0\}\,. \quad (4.46)$$

A hierarchical bubble function is a basis function which is generally implemented as an incomplete polynomial of degree two or three on a linear basis element. The example bubble function for a two-dimensional bilinear basis element is defined as a biquadratic term, e.g. $(1-\eta^2)(1-\xi^2)/4$, at a the mid-element coordinate of a quadrilateral spanned by four Lagrange bilinear shape functions $(1\pm\eta)(1\pm\xi)/4$. With the bubble function associated with a fifth node and fifth degree-of-freedom, the hierarchical element functions have been integrated for this study using Macsyma®. The leading $4\otimes 4$ submatrices are then identical to the $4\otimes 4$ Lagrange bilinear basis element matrices. The heirarchical $5\otimes 5$ element matrices, denoted here as B2[·]H, are, for convection,

$$\mathtt{B201H} = \frac{1}{18}\begin{bmatrix} -6 & 6 & 3 & -3 & 2 \\ -6 & 6 & 3 & -3 & -2 \\ -3 & 3 & 6 & -6 & -2 \\ -3 & 3 & 6 & -6 & 2 \\ -2 & 2 & 2 & -2 & 0 \end{bmatrix}, \mathtt{B202H} = \frac{1}{18}\begin{bmatrix} -6 & -3 & 3 & 6 & 2 \\ -3 & -6 & 6 & 3 & 2 \\ -3 & -6 & 6 & 3 & -2 \\ -6 & -3 & 3 & 6 & -2 \\ -2 & -2 & 2 & 2 & 0 \end{bmatrix}. \quad (4.47)$$

The corresponding diffusion matrices are

$$\mathtt{B211H} = \frac{1}{6}\begin{bmatrix} 2 & -2 & -1 & 1 & 0 \\ -2 & 2 & 1 & -1 & 0 \\ -1 & 1 & 2 & -2 & 0 \\ 1 & -1 & -2 & -2 & 0 \\ 0 & 0 & 0 & 0 & \frac{16}{5} \end{bmatrix}, \mathtt{B222H} = \frac{1}{6}\begin{bmatrix} 2 & 1 & -1 & -2 & 0 \\ 1 & 2 & -2 & -1 & 0 \\ -1 & -2 & 2 & 1 & 0 \\ -2 & -1 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & \frac{16}{5} \end{bmatrix}. \quad (4.48)$$

The assembly of the Galerkin Lagrange bilinear form $\mathcal{W}^h$ (4.35), the steady por-

tion of (2.51), is repeated here as

$$S\Big(\mathcal{W}^h(\{Q\})\Big) = \left\{ \frac{\bar{U}_x\Delta_y}{12}\begin{vmatrix} -1 & 0 & 1 \\ -4 & 0 & 4 \\ -1 & 0 & 1 \end{vmatrix} + \frac{\bar{U}_y\Delta_x}{12}\begin{vmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{vmatrix} \right.$$
$$\left. + \frac{1}{6\,\mathrm{Pe}}\left[\frac{\Delta_y}{\Delta_x}\begin{vmatrix} -1 & 2 & -1 \\ -4 & 8 & -4 \\ -1 & 2 & -1 \end{vmatrix} + \frac{\Delta_x}{\Delta_y}\begin{vmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{vmatrix} \right] \right\} \{Q\} = \{0\}\,. \tag{4.49}$$

The hierarchical bubble function form is produced by assembling the $5 \otimes 5$ element matrices (4.47)-(4.48) and using Gauss elimination on the assembled form to eliminate the bubble (center) nodes. In this steady formulation with no boundary conditions specified, no data appears on the right-hand side as found in (4.44), and the matrices are fully assembled before Gauss elimination. The hierarchical form, expressed as an addition to (4.49), is

$$S\Big(\hat{\mathcal{W}}^h(\{Q\})\Big) = S\Big(\mathcal{W}^h(\{Q\})\Big) + \frac{5(\Delta_x\Delta_y)^2\,\mathrm{Pe}}{2(\Delta_x{}^2+\Delta_y{}^2)}\left\{ \frac{\Delta_y\bar{U}_x^2}{6\Delta_x}\begin{vmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{vmatrix} \right.$$
$$\left. + \frac{\Delta_x\bar{U}_y^2}{6\Delta_y}\begin{vmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{vmatrix} + \frac{2\bar{U}_x\bar{U}_y}{6}\begin{vmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{vmatrix} \right\} \{Q\} = \{0\}\,, \tag{4.50}$$

where the last matrix coincides with the GWS assembled form of `B212B + B221B`, corresponding to a mixed-direction diffusion term. The first two matrices in (4.50) may be decomposed by tensor products of the form of (2.61) as

$$\frac{\Delta_y}{6\Delta_x}\begin{vmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{vmatrix} = \left(\frac{\Delta_y}{6}\right)\begin{vmatrix} 1 \\ 2 \\ 1 \end{vmatrix} \otimes \left(\frac{1}{\Delta_x}\right)\begin{vmatrix} -1 & 2 & -1 \end{vmatrix}\,,$$
$$\frac{\Delta_x}{6\Delta_y}\begin{vmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{vmatrix} = \left(\frac{1}{\Delta_x}\right)\begin{vmatrix} -1 \\ 2 \\ -1 \end{vmatrix} \otimes \left(\frac{\Delta_y}{6}\right)\begin{vmatrix} 1 & 2 & 1 \end{vmatrix}\,. \tag{4.51}$$

The two tensor products do not appear to match any Galerkin matrix, but do "resemble"

the diffusion matrices B211B and B222B with the $\{1, \quad 4, \quad 1\}$ in (2.61) replaced by $\{1, \quad 2, \quad 1\}$ in (4.51). Thus the bubble function indeed adds artificial diffusion to the Galerkin formulation.

The leading coefficient in (4.50) is of order $\mathrm{Pe}\Delta_x{}^2$ and is analogous to $\bar{k}$ in the SUPG formulation, which is also proportional to Peclet number, but only at very small Peclet numbers. The bubble function algorithm (4.51) thus appears to produce an unboundedly large artificial diffusion term at large Peclet number.

4.3.3 SGM method

The SGM method of Roy and Baker [69] is considered for the steady advection-diffusion equation,

$$\mathbf{U}\cdot\nabla q - \frac{1}{\mathrm{Pe}}\nabla\cdot\nabla q = 0\,, \tag{4.52}$$

which has a Galerkin finite element weak statement approximation, neglecting boundary integrals, of

$$\mathcal{W}^h(\{Q\}) = \int_\Omega \{\psi\}\mathbf{U}\cdot\nabla\{\psi\}^T d\tau\,\{Q\} + \frac{1}{\mathrm{Pe}}\int_\Omega \nabla\{\psi\}\cdot\nabla\{\psi\}^T d\tau\,\{Q\} = \{0\}\,. \tag{4.53}$$

The SGM theory produces a modified bilinear basis diffusion matrix generated from an element-level static condensation of a biquadratic basis diffusion matrix. A parameter c_i determines the level of artificial diffusion in the x_i direction [69] (37,38) as

$$c_i = \frac{3U_i h_i \mathrm{Pe}}{2\,\mathcal{F}} - \frac{1}{2}\,,\quad \mathcal{F} = 2\left(\frac{\Lambda|\mathrm{J}|\mathrm{Pe}^{n-1}}{h_i}\right)^{1/n}\,. \tag{4.54}$$

With Λ an adjustable constant to be determined in n dimensions, $1 \le i \le n$. Combining the expressions in (4.54) for $n = 2$ yields

$$c_i = \frac{3U_i}{4}\left(\frac{h_i^3\,\mathrm{Pe}}{\Lambda|\mathrm{J}|}\right)^{\frac{1}{2}} - \frac{1}{2}\,. \tag{4.55}$$

Curve fitting data from the numerical example cases [70] to be discussed yielded an explicit form of

$$\Lambda = 0.45425\,\mathrm{Pe}^{-0.023481}\,. \tag{4.56}$$

The SGM replacement bilinear basis diffusion matrix over $n = 2$ dimensions, denoted here as B2KKS, is [68]

$$\text{B2KKS} = \frac{1}{d_{00}} \begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} \\ d_{12} & d_{11} & d_{14} & d_{13} \\ d_{13} & d_{14} & d_{11} & d_{12} \\ d_{14} & d_{13} & d_{12} & d_{11} \end{bmatrix}, \tag{4.57}$$

with

$$\begin{aligned}
d_{00} &= (1971360c_x{}^2 + 2184480c_x + 569160)c_y{}^2 \\
&\quad + (2184480c_x{}^2 + 3227040c_x + 1077480)c_y \\
&\quad + 569160c_x{}^2 + 1077480c_x + 479610, \\
d_{11} &= (194176c_x{}^3 + 456864c_x{}^2 + 313224c_x + 64036)c_y{}^3 \\
&\quad + (456864c_x{}^3 + 1088192c_x{}^2 + 793564c_x + 176830)c_y{}^2 \\
&\quad + (313224c_x{}^3 + 793564c_x{}^2 + 625680c_x + 152182)c_y \\
&\quad + (64036c_x{}^3 + 176830c_x{}^2 + 152182c_x + 40077), \\
d_{12} &= -(87616c_x{}^3 + 146624c_x{}^2 + 78984c_x + 15076)c_y{}^3 \\
&\quad + (222784c_x{}^3 + 403312c_x{}^2 + 236364c_x + 49990)c_y{}^2 \\
&\quad + (165684c_x{}^3 + 342484c_x{}^2 + 224660c_x + 52822)c_y \\
&\quad + 35516c_x{}^3 + 83730c_x{}^2 + 61642c_x + 15987, \\
d_{13} &= -((18944c_x{}^3 + 87456c_x{}^2 + 68456c_x + 13444)c_y{}^3 \\
&\quad + (87456c_x{}^3 + 281568c_x{}^2 + 214716c_x + 43110)c_y{}^2 \\
&\quad + (68456c_x{}^3 + 214716c_x{}^2 + 176460c_x + 37718)c_y \\
&\quad + 13444c_x{}^3 + 43110c_x{}^2 + 37718c_x + 8103), \\
d_{14} &= -((87616c_x{}^3 + 222784c_x{}^2 + 165684c_x + 35516)c_y{}^3 \\
&\quad + (146624c_x{}^3 + 403312c_x{}^2 + 342484c_x + 83730)c_y{}^2 \\
&\quad + (78984c_x{}^3 + 236364c_x{}^2 + 224660c_x + 61642)c_y \\
&\quad + 15076c_x{}^3 + 49990c_x{}^2 + 52822c_x + 15987).
\end{aligned} \tag{4.58}$$

An example calculation with $c_x = c_y = 1$ in [69] (35), is replicated by (4.58) as

$$\texttt{B2KKS} = \frac{1}{66}\begin{bmatrix} 29 & -11 & -7 & -11 \\ -11 & 29 & -11 & -7 \\ -7 & -11 & 29 & -11 \\ -11 & -7 & -11 & 29 \end{bmatrix}. \tag{4.59}$$

Some examples are also calculated in [69] for (4.53) formed on a uniform square mesh with $\Delta_x = \Delta_y = \frac{1}{8}$ and $U_x = U_y = 1$. Two examples are (a) $\mathrm{Pe} = 2000, \Lambda = 0.38, \mathcal{F} = 9.75, c = 38$ and (b) $\mathrm{Pe} = 20000, \Lambda = 0.36, \mathcal{F} = 30, c = 125$. In each case, the assembled Galerkin bilinear convection stencil for (4.53) is

$$\mathcal{S}\left(\frac{\bar{U}_x\Delta_y}{2}\,\texttt{B201B} + \frac{\bar{U}_y\Delta_x}{2}\,\texttt{B202B}\right) = \frac{1}{48}\overline{\begin{vmatrix} 0 & 2 & 1 \\ -2 & 0 & 2 \\ -1 & -2 & 0 \end{vmatrix}}, \tag{4.60}$$

and the assembled Galerkin bilinear diffusion stencil is

$$\frac{1}{\mathrm{Pe}}\mathcal{S}\left(\frac{\Delta_y}{\Delta_x}\,\texttt{B211B} + \frac{\Delta_x}{\Delta_y}\,\texttt{B222B}\right) = \frac{1}{3\,\mathrm{Pe}}\overline{\begin{vmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{vmatrix}}. \tag{4.61}$$

The sum of (4.61) and (4.60) is

$$\mathcal{S}\Big(\mathcal{W}^h(\{Q\})\Big) = \frac{1}{48\,\mathrm{Pe}}\overline{\begin{vmatrix} -16 & 2\mathrm{Pe}-16 & \mathrm{Pe}-16 \\ -2\mathrm{Pe}-16 & 128 & 2\mathrm{Pe}-16 \\ -\mathrm{Pe}-16 & -2\mathrm{Pe}-16 & -16 \end{vmatrix}}\{Q\}. \tag{4.62}$$

Provided the center (2,2) entry is positive, and the other entries are nonpositive, the matrix (4.62) is an M-matrix producing a monotone solution. This condition is controlled in the GWS case (4.62) by the (1,2) and (2,3) entries, satisfied for $\mathrm{Pe} \leq 8$ on the given mesh or by the cell Peclet number $\frac{\mathrm{Pe}\Delta_x}{L} \leq 1$.

Using (4.55-4.57), the assembled condensed SGM diffusion matrix stencil for

$\mathrm{Pe} = 2000, c_x = c_y = 38$ evaluates as

$$\frac{1}{\mathrm{Pe}} S\Big(\texttt{B2KKS}\Big) = \frac{1}{121.4} \begin{vmatrix} -1 & -8.203 & -1 \\ -8.203 & 36.811 & -8.203 \\ -1 & -8.203 & -1 \end{vmatrix}, \tag{4.63}$$

and for $\mathrm{Pe} = 20000, c_x = c_y = 125$, as

$$\frac{1}{\mathrm{Pe}} S\Big(\texttt{B2KKS}\Big) = \frac{1}{126.1} \begin{vmatrix} -1 & -8.897 & -1 \\ -8.897 & 39.586 & -8.897 \\ -1 & -8.897 & -1 \end{vmatrix}. \tag{4.64}$$

Summing (4.63) or (4.64) with the Galerkin convection matrix stencil (4.60) yields

$$\begin{aligned} \mathrm{Pe} = 2000 : S\Big(\hat{\mathcal{W}}^h(\{Q\})\Big) &= \begin{vmatrix} -0.008240 & -0.025922 & 0.012594 \\ -0.109256 & 0.303315 & -0.025922 \\ -0.029073 & -0.109256 & -0.008240 \end{vmatrix}, \\ \mathrm{Pe} = 20000 : S\Big(\hat{\mathcal{W}}^h(\{Q\})\Big) &= \begin{vmatrix} -0.007931 & -0.028892 & 0.012902 \\ -0.112226 & 0.313960 & -0.028892 \\ -0.028764 & -0.112226 & -0.007931 \end{vmatrix}. \end{aligned} \tag{4.65}$$

Either of the matrices (4.65) is "nearly" an M-matrix in that the center entry is positive and all outlying entries have a negative sign, except the upper right (1,3) entry, which is small compared with the center entry. Presumably a different choice in (4.56) would complete the M-matrix, although the actual values from [69] were used for this calculation.

4.4 Other methods

4.4.1 Galerkin least-squares method

The Galerkin least-squares method, cf. Jiang and Carey [51], is applicable only to first order PDE systems, e.g. the unsteady one-dimensional advection equation,

$$\frac{\partial q}{\partial t} + u\frac{\partial q}{\partial x} = 0\,. \tag{4.66}$$

The Galerkin least-squares approximation to $\mathcal{L}(q) = s$ is constructed using a symmetric functional I, the variational precursor to a weak statement, or

$$\mathrm{I}\Big(\mathcal{L}\Big(q(\vec{x},t)\Big)-s(\vec{x},t)\Big) = \int_\Omega \Big[\mathcal{L}\Big(q(\vec{x},t)\Big)-s(\vec{x},t)\Big]\Big[\mathcal{L}\Big(q(\vec{x},t)\Big)-s(\vec{x},t)\Big]d\Omega = 0\,. \quad (4.67)$$

Since each term in brackets is zero by definition, the integral obviously holds for the analytical case. In this variational form, the variables q and s are approximated as in the Galerkin method (2.3), i.e. $q(\vec{x},t) \equiv \sum_{l=1}^{\infty} \psi_l(\vec{x})Q_l(t)$. Inserting the interpolations yields

$$\mathrm{I}^h\Big(\mathcal{L}(q)-s\Big) = \int_\Omega \sum_{l=1}^{M}\Big[\mathcal{L}\Big(\chi_l Q_l(t)\Big)-\chi_l S_l\Big]\sum_{m=1}^{M}\Big[\mathcal{L}\Big(\psi_m Q_m(t)\Big)-\psi_m S_m\Big]d\Omega = 0\,. \quad (4.68)$$

Taking the stationary point of I with respect to the expansion coefficients Q_l (which replace W_l, the expansion coefficients of the test function in the Galerkin method, from (2.4)) yields

$$\mathcal{W}^h\Big(\mathcal{L}(q)-s\Big) = \int_\Omega \sum_{l=1}^{M}\Big[\mathcal{L}\Big(\chi_l\Big)\Big]\sum_{m=1}^{M}\Big[\mathcal{L}\Big(\psi_m\{Q\}(t)\Big)-\psi_m S_m\Big]d\Omega = 0\,. \quad (4.69)$$

Therefore the Galerkin least-squares method is an attempt to construct a variational method for a hyperbolic problem, while the Galerkin method is equivalent to a variational method only if $\mathcal{L}$ is elliptic.

For this specific application, discretize (4.66) in time and rescale to Courant number form, yielding a corresponding semi-discrete form

$$q^{n+1} - q^n + C\frac{\partial}{\partial x}\left(\theta q^{n+1} + (1-\theta)q^n\right) = 0\,. \quad (4.70)$$

Separating (4.70) by time levels yields

$$\left(1 + C\theta\frac{\partial}{\partial x}\right)q^{n+1} = \left(1 + C(\theta-1)\frac{\partial}{\partial x}\right)q^n\,. \quad (4.71)$$

Taking q^{n+1} as the unknown and q^n as known data, the left-hand side of (4.71) is a linear operation on the unknown and is substituted for $\mathcal{L}(q)$ in (4.69), while the right-hand side

of (4.71) is known data and is substituted for s, yielding

$$\begin{aligned}\int_\Omega \left[\left(1+C\theta\frac{\partial}{\partial x}\right)\{\psi\}\right]\left[\left(1+C\theta\frac{\partial}{\partial x}\right)\{\psi\}^T\{Q\}^{n+1}\right]d\tau = \\ \int_\Omega \left[\left(1+C\theta\frac{\partial}{\partial x}\right)\{\psi\}\right]\left[\left(1+C(\theta-1)\frac{\partial}{\partial x}\right)\{\psi\}^T\{Q\}^{n}\right]d\tau\,,\end{aligned} \tag{4.72}$$

or

$$\begin{aligned}\int_\Omega \Big[\{\psi\}\{\psi\}^T + C\theta\left(\{\psi\}\frac{\partial\{\psi\}^T}{\partial x}+\frac{\partial\{\psi\}}{\partial x}\{\psi\}^T\right) \\ + C^2\theta^2\frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}\Big]d\tau\{Q\}^{n+1} = \\ \int_\Omega \Big[\{\psi\}\{\psi\}^T + C\left(\theta\{\psi\}\frac{\partial\{\psi\}^T}{\partial x}+(\theta-1)\frac{\partial\{\psi\}}{\partial x}\{\psi\}^T\right) \\ + C^2(\theta-1)^2\frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}\Big]d\tau\{Q\}^{n}\,.\end{aligned} \tag{4.73}$$

Its one-dimensional linear basis finite element matrix form is

$$\begin{aligned}\Big[\,\mathtt{A200L}+C\theta(\,\mathtt{A210L}+\,\mathtt{A201L})+C^2\theta^2\,\mathtt{A211L}\Big]\,\{Q\}^{n+1} = \\ \Big[\,\mathtt{A200L}+C\theta\,\mathtt{A210L}+C(\theta-1)\,\mathtt{A201L}+C^2\theta(\theta-1)\,\mathtt{A211L}\Big]\,\{Q\}^{n}.\end{aligned} \tag{4.74}$$

Converting to delta form yields

$$\begin{aligned}\Big[\,\mathtt{A200L}+C\theta(\,\mathtt{A201L}+\,\mathtt{A210L})+C^2\theta^2\,\mathtt{A211L}\Big]\,(\{Q\}^{n+1}-\{Q\}^{n}) = \\ -\Big[C\,\mathtt{A201L}+C^2\theta\,\mathtt{A211L}\Big]\,\{Q\}^{n+\theta}.\end{aligned} \tag{4.75}$$

Comparing (4.75) to the TWS form (4.18), the Galerkin least-squares algorithm for (4.66) may be duplicated using TWS coefficients $\alpha=\theta$, $\beta=\gamma=\theta^2$, and $\mu=0$, and the left-hand side of (4.75) is symmetric as intended.

4.4.2 Space filtering methods

The "space filtering" method of Aldama [2] and Cantekin et al. [13] derives from A. Leonard [56]. Leonard's method uses the Reynolds decomposition (although for laminar flow) and the inverse of a low pass filter on the $2\Delta_x$ mesh size to presumably extend

the frequency content to the Δ_x^2 range using

$$\overline{U_i U_j} = \bar{u}_i \bar{u}_j + \overline{u_i' u_j'} + \frac{4\Delta_x^2}{6} \frac{\partial \bar{u}_i}{\partial x_k} \frac{\partial \bar{u}_j}{\partial x_k} , \tag{4.76}$$

with the turbulent term $\overline{u_i' u_j'}$ neglected in laminar flow. These authors, as do others with different methods, regard their various methods for the advection term as "improved accuracy" rather than artificial diffusion although they report reductions in $2\Delta_x$ dispersive waves. Both Aldama [2] and Cantekin et al. [13] use quadratic elements with the filtering scheme since, as an $O(\Delta_x^2)$ correction, it should require a space accuracy of higher order. Here, however, the method will be examined as an artificial diffusion term with linear elements.

The Galerkin weak form of (4.76) for the laminar Navier-Stokes equations, assuming conservation form and neglecting surface integrals, is

$$\begin{aligned} \int_\Omega \psi \frac{\partial(\overline{U_i U_j})}{\partial x_j} d\tau &= \int_\Omega \psi \frac{\partial(\bar{u}_i \bar{u}_j)}{\partial x_j} d\tau + \frac{4\Delta_x^2}{6} \int_\Omega \psi \frac{\partial}{\partial x_j} \left(\frac{\partial \bar{u}_i}{\partial x_k} \frac{\partial \bar{u}_j}{\partial x_k} \right) d\tau \\ &= - \int_\Omega \frac{\partial \psi}{\partial x_k} \bar{u}_i \bar{u}_j \, d\tau - \frac{4\Delta_x^2}{6} \int_\Omega \frac{\partial \psi}{\partial x_j} \frac{\partial \bar{u}_i}{\partial x_k} \frac{\partial \bar{u}_j}{\partial x_k} d\tau . \end{aligned} \tag{4.77}$$

A finite element form of the last term is

$$
\begin{aligned}
&-\frac{4\Delta_x^2}{6}\int_\Omega \frac{\partial\{\psi\}}{\partial x_j}\frac{\partial \bar{u}_i}{\partial x_k}\frac{\partial \bar{u}_j}{\partial x_k}d\tau = -\frac{4\Delta_x^2}{6}\int_\Omega \frac{\partial\{\psi\}}{\partial x_j}\frac{\partial\{\psi\}^T}{\partial x_k}\{U_i\}\frac{\partial\{\psi\}^T}{\partial x_k}\{U_j\}d\tau \\
&= -\frac{4\Delta_x^2}{6}\{U_i\}^T\int_\Omega \frac{\partial\{\psi\}}{\partial x_j}\frac{\partial\{\psi\}}{\partial x_k}\frac{\partial\{\psi\}^T}{\partial x_k}d\tau\{U_j\} \\
&= -\frac{4\Delta_x^2}{6}\{U_i\}^T\left(\int_\Omega \frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}}{\partial x_k}\frac{\partial\{\psi\}^T}{\partial x_k}d\tau\{U_x\} + \int_\Omega \frac{\partial\{\psi\}}{\partial y}\frac{\partial\{\psi\}}{\partial x_k}\frac{\partial\{\psi\}^T}{\partial x_k}d\tau\{U_y\}\right) \\
&= -\frac{4\Delta_x^2}{6}\{U_i\}^T\left(\int_\Omega \frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}d\tau\{U_x\} + \int_\Omega \frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}}{\partial y}\frac{\partial\{\psi\}^T}{\partial y}d\tau\{U_x\}\right. \\
&\left. + \int_\Omega \frac{\partial\{\psi\}}{\partial y}\frac{\partial\{\psi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}d\tau\{U_y\} + \int_\Omega \frac{\partial\{\psi\}}{\partial y}\frac{\partial\{\psi\}}{\partial y}\frac{\partial\{\psi\}^T}{\partial y}d\tau\{U_y\}\right) \\
&= -\frac{4\Delta_x^2}{6}\{U_i\}^T\left(\texttt{M3XXX}\{U_x\} + \texttt{M3XYY}\{U_x\} + \texttt{M3YXX}\{U_y\} + \texttt{M3YYY}\{U_y\}\right) \\
&= -\frac{4\Delta_x^2}{6}\{U_x\}^T\left(\texttt{M3XXX} + \texttt{M3YYX}\right)\{U_i\} + \{U_y\}^T\left(\texttt{M3XXY} + \texttt{M3YYY}\right)\{U_i\}\,.
\end{aligned}
\tag{4.78}
$$

Including the element coordinate transformation on a rectangular grid for the x-direction, that term of (4.78) is

$$
\begin{aligned}
-\frac{4\Delta_x^2}{6}\{U_x\}^T\ \texttt{M3XXX}\{U_i\} &= -\frac{4\Delta_x^2}{6}\frac{\frac{\partial\eta}{\partial x}^3}{|\mathrm{J}|^2}\{U_x\}^T\ \texttt{B3111B}\{U_i\} \\
&= -\frac{4\Delta_y}{3}\{U_x\}^T\ \texttt{B3111B}\{U_i\}\,,
\end{aligned}
\tag{4.79}
$$

and the coefficient of the term is $O(\Delta_y)$, large on most grids compared to physical diffusion terms of $O(\frac{1}{\mathrm{Pe}})$. With a correction of that magnitude, the necessity for quadratic elements to maintain order of accuracy is not apparent. Evaluating the constant hypermatrix B3111B,

$$
\texttt{B3111B} = \frac{1}{12}
\begin{bmatrix}
(-3,3,1,-1) & (3,-3,-1,1) & (1,-1,-1,1) & (-1,1,1,-1) \\
(3,-3,-1,1) & (-3,3,1,-1) & (-1,1,1,-1) & (1,-1,-1,1) \\
(1,-1,-1,1) & (-1,1,1,-1) & (-1,1,3,-3) & (1,-1,-3,3) \\
(-1,1,1,-1) & (1,-1,-1,1) & (1,-1,-3,3) & (-1,1,3,-3)
\end{bmatrix}.
\tag{4.80}
$$

The left product of a constant onto B3111B is 0, and the left product of a $2\Delta_x$ wave is

$$\begin{bmatrix} 1 & -1 \\ 1 & -1 \end{bmatrix} \texttt{B3111B} = \frac{1}{12} \begin{bmatrix} -8 & 8 & 4 & -4 \\ 8 & -8 & -4 & 4 \\ 4 & -4 & -8 & 8 \\ -4 & 4 & 8 & -8 \end{bmatrix} . \tag{4.81}$$

Assembled as a stencil, (4.81) represents the diffusion matrix,

$$\frac{1}{3} \begin{vmatrix} \overline{-1 & 2 & -1} \\ -4 & 8 & -4 \\ \underline{-1 & 2 & -1} \end{vmatrix} = S\Big(2\,\texttt{B211B}\Big) , \tag{4.82}$$

and the method has the desirable property of adding a diffusion matrix to $2\Delta_x$ waves while adding nothing in smooth regions. Unfortunately, the sign of (4.82) depends on the sign of the $2\Delta_x$ wave, thus it would add anti-diffusion for the equally likely opposite wave.

Modifying (4.78) to generate only a positive diffusion contribution,

$$-\frac{4\Delta_x^2}{6}\{U_i\}^T \ \texttt{B3XXX}\ \{U_j\} \Rightarrow sign(\frac{\partial u}{\partial x})\frac{4\Delta_x^2}{6}\{U_i\}^T \ \texttt{B3XXX}\ \{U_j\} . \tag{4.83}$$

Considering the integration by parts, (4.83) appears equivalent to the early artificial diffusion form of von Neumann and Richtmeyer [80] with parameter $c = \frac{4}{6}$, i.e.,

$$\frac{\partial q}{\partial x} = -\frac{\partial}{\partial x}\left(c\Delta_x^2 \frac{\partial u}{\partial x}\left|\frac{\partial u}{\partial x}\right|\right) . \tag{4.84}$$

4.4.3 M-matrix monotone methods

A monotone artificial diffusion method is now derived for a steady form of a generalized two-dimensional TWS algorithm. An abstract TWS form (4.16) for the advection-diffusion equation is the beginning point, setting $\mu = 0$ and taking the steady part. The artificial diffusion coefficient $\beta\Delta_t$ is replaced with functions $\beta_{xx}, \beta_{yy}, \beta_{xy}$, to be determined and having units of time. Assembling this steady TWS method on a rectangular

mesh yields

$$\hat{\mathcal{L}}^h(\{Q\}) = \left(\frac{\bar{U}_x \Delta_y}{2} \texttt{B201} + \frac{\bar{U}_y \Delta_x}{2} \texttt{B202} + \beta_{xx} \bar{U}_x^2 \frac{\Delta_y}{\Delta_x} \texttt{B211} \right.$$
$$\left. + \beta_{yy} \bar{U}_y^2 \frac{\Delta_x}{\Delta_y} \texttt{B222} + \beta_{xy} \bar{U}_x \bar{U}_y (\texttt{B212} + \texttt{B221}) \right) \{Q\} = \{0\} . \tag{4.85}$$

Physical diffusion, if present, would be added to the β_{ij} artificial diffusion terms and is not considered here. The discrete bilinear Galerkin assembled form on a rectangular grid is

$$S\left(\hat{W}^h(\{Q\})\right) = \left\{ \frac{\bar{U}_x \Delta_y}{12} \begin{vmatrix} -1 & 0 & 1 \\ -4 & 0 & 4 \\ -1 & 0 & 1 \end{vmatrix} + \frac{\bar{U}_y \Delta_x}{12} \begin{vmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{vmatrix} \right.$$
$$+ \frac{\beta_{xx} \bar{U}_x^2 \Delta_y}{6\Delta_x} \begin{vmatrix} -1 & 2 & -1 \\ -4 & 8 & -4 \\ -1 & 2 & -1 \end{vmatrix} + \frac{\beta_{yy} \bar{U}_y^2 \Delta_x}{6\Delta_y} \begin{vmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{vmatrix} \tag{4.86}$$
$$\left. + \frac{\beta_{xy} \bar{U}_x \bar{U}_y}{2} \begin{vmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{vmatrix} \right\} \{Q\} = \{0\} .$$

The criterion [78] for a monotone solution requires that the coefficients in (4.86) form an M-matrix, or that the diagonal matrix coefficient on $Q_{i,j}$ be positive with all other coefficients zero or negative. For $\beta > 0$, the criterion will also be satisfied if the assembled matrix coefficients at downwind nodes are zero. For example, if velocity is positive, the downwind nodes are $Q_{i+1,j}, Q_{i,j+1}, Q_{i+1,j+1}$ or the $(1,2)$, $(1,3)$, and $(2,3)$ entries of the $3 \otimes 3$ matrix, the three entries potentially yielding three equations for the three β_{ij} coefficient. Considering the four possible combinations for $\bar{U}_x$ and $\bar{U}_y$ and solving

for zero coefficients at the resulting downwind nodes yields

$$
\begin{aligned}
&(1)\ \bar{U}_x = |\bar{U}_x|\ ,\ \bar{U}_y = |\bar{U}_y|: \\
&\qquad \beta_{xx}\bar{U}_x^2 = \frac{2\Delta_x\Delta_y|\bar{U}_x| + \Delta_x^2|\bar{U}_y|}{3\Delta_y}\ ,\ \beta_{yy}\bar{U}_y^2 = \frac{\Delta_y^2|\bar{U}_x| + 2\Delta_x\Delta_y|\bar{U}_y|}{3\Delta_x}\ , \\
&\qquad \beta_{xy}\bar{U}_x\bar{U}_y = \frac{-\Delta_y|\bar{U}_x| - \Delta_x|\bar{U}_y|}{6}\ , \\
&(2)\ \bar{U}_x = -|\bar{U}_x|\ ,\ \bar{U}_y = |\bar{U}_y|: \\
&\qquad \beta_{xx}\bar{U}_x^2 = \frac{2\Delta_x\Delta_y|\bar{U}_x| + \Delta_x^2|\bar{U}_y|}{3\Delta_y}\ ,\ \beta_{yy}\bar{U}_y^2 = \frac{\Delta_y^2|\bar{U}_x| + 2\Delta_x\Delta_y|\bar{U}_y|}{3\Delta_x}\ , \\
&\qquad \beta_{xy}\bar{U}_x\bar{U}_y = \frac{\Delta_y|\bar{U}_x| + \Delta_x|\bar{U}_y|}{6}\ , \\
&(3)\ \bar{U}_x = |\bar{U}_x|\ ,\ \bar{U}_y = -|\bar{U}_y|: \\
&\qquad \beta_{xx}\bar{U}_x^2 = \frac{2\Delta_x\Delta_y|\bar{U}_x| + \Delta_x^2|\bar{U}_y|}{3\Delta_y}\ ,\ \beta_{yy}\bar{U}_y^2 = \frac{\Delta_y^2|\bar{U}_x| + 2\Delta_x\Delta_y|\bar{U}_y|}{3\Delta_x}\ , \\
&\qquad \beta_{xy}\bar{U}_x\bar{U}_y = \frac{\Delta_y|\bar{U}_x| + \Delta_x|\bar{U}_y|}{6}\ , \\
&(4)\ \bar{U}_x = -|\bar{U}_x|\ ,\ \bar{U}_y = -|\bar{U}_y|: \\
&\qquad \beta_{xx}\bar{U}_x^2 = \frac{2\Delta_x\Delta_y|\bar{U}_x| + \Delta_x^2|\bar{U}_y|}{3\Delta_y}\ ,\ \beta_{yy}\bar{U}_y^2 = \frac{\Delta_y^2|\bar{U}_x| + 2\Delta_x\Delta_y|\bar{U}_y|}{3\Delta_x}\ , \\
&\qquad \beta_{xy}\bar{U}_x\bar{U}_y = \frac{-\Delta_y|\bar{U}_x| - \Delta_x|\bar{U}_y|}{6}\ .
\end{aligned}
\tag{4.87}
$$

The expressions in (4.87) may be combined as

$$
\begin{aligned}
\beta_{xx}\bar{U}_x^2 &= \frac{2\Delta_x\Delta_y|\bar{U}_x| + \Delta_x^2|\bar{U}_y|}{3\Delta_y}\ , \\
\beta_{yy}\bar{U}_y^2 &= \frac{\Delta_y^2|\bar{U}_x| + 2\Delta_x\Delta_y|\bar{U}_y|}{3\Delta_x}\ , \\
\beta_{xy}\bar{U}_x\bar{U}_y &= \frac{-\mathrm{sign}(\bar{U}_x)\mathrm{sign}(\bar{U}_y)}{6}(\Delta_y|\bar{U}_x| + \Delta_x|\bar{U}_y|)\ .
\end{aligned}
\tag{4.88}
$$

This construction will be designated in computational tests as the "first monotone Petrov-Galerkin method" or MPG-1. One possible disadvantage of this construction is the

presence of large cross-wind diffusion contribution from $|\bar{U}_y|$ in β_{xx}, etc, which does not reduce to zero for a one-dimensional velocity in two dimensions.

A modified monotone method was formed by "row-condensing" the diffusion matrices B211 and B222 to a finite difference form denoted here as B2[·]R,

$$\begin{aligned} \mathtt{B211R} &= \frac{1}{6}\begin{bmatrix} 3 & -3 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & -3 & -3 \end{bmatrix}, \\ \mathtt{B222R} &= \frac{1}{6}\begin{bmatrix} 3 & 0 & 0 & -3 \\ 0 & 3 & -3 & 0 \\ 0 & -3 & 3 & 0 \\ -3 & 0 & 0 & 3 \end{bmatrix}. \end{aligned} \tag{4.89}$$

The discrete assembled form of (4.85) using the row-condensed diffusion matrices (4.89) is

$$\begin{aligned} S\left(\hat{W}^h(\{Q\})\right) = \Bigg\{ & \frac{\bar{U}_x\Delta_y}{12}\begin{vmatrix} -1 & 0 & 1 \\ -4 & 0 & 4 \\ -1 & 0 & 1 \end{vmatrix} + \frac{\bar{U}_y\Delta_x}{12}\begin{vmatrix} 1 & 4 & 1 \\ 0 & 0 & 0 \\ -1 & -4 & -1 \end{vmatrix} \\ & + \frac{\beta_{xx}\Delta_y}{6\Delta_x}\begin{vmatrix} 0 & 0 & 0 \\ -6 & 12 & -6 \\ 0 & 0 & 0 \end{vmatrix} + \frac{\beta_{yy}\Delta_x}{6\Delta_y}\begin{vmatrix} 0 & -6 & 0 \\ 0 & 12 & 0 \\ 0 & -6 & 0 \end{vmatrix} \\ & + \frac{\beta_{xy}}{6}\begin{vmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{vmatrix} \Bigg\} \{Q\} = \{0\}. \end{aligned} \tag{4.90}$$

The coefficients of (4.90) are solved by direction, as in (4.87), then combined, eliminating cross-wind diffusion. The result is

$$\begin{aligned} \beta_{xx}\bar{U}_x^2 &= \frac{\Delta_x|\bar{U}_x|}{2}, \\ \beta_{yy}\bar{U}_y^2 &= \frac{\Delta_y|\bar{U}_y|}{2}, \\ \beta_{xy}\bar{U}_x\bar{U}_y &= \frac{-\text{sign}(\bar{U}_x)\text{sign}(\bar{U}_y)}{6}\min(\Delta_y|\bar{U}_x|\,,\,\Delta_x|\bar{U}_y|). \end{aligned} \tag{4.91}$$

This method will be designated in computational tests as the "second monotone Petrov-Galerkin method" or MPG-2. This method uses the row-condensed diffusion matrices B211R and B222R (4.89) along with the Galerkin diffusion matrix sum B212 + B221.

4.5 Dissipation length and time scales

Here an abstract multi-dimensional TWS method such as (4.16) for the advection-diffusion equation is considered; equivalently it may represent an SUPG method with directional diffusion coefficients (4.23). Modifications are made to preceding versions, as the TWS coefficients are given directions by subscript, the TWS μ term is neglected, and the physical diffusion term $\frac{1}{\mathrm{Pe}}(\cdots)$ is incorporated. The weak statement form is

$$\begin{aligned}\hat{\mathcal{W}}^h(q) = &\left(\mathtt{M200} + \alpha_j \bar{U}_j \Delta_t \, \mathtt{M2J0} + \gamma_{kj} \bar{U}_k \bar{U}_j \Delta_t^2 \, \mathtt{M2KJ} \right) \frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} \\ &+ \left(\bar{U}_j \, \mathtt{M20J} + \beta_{kj} \bar{U}_k \bar{U}_j \Delta_t \, \mathtt{M2KJ} + \frac{1}{\mathrm{Pe}} \, \mathtt{M2JJ} \right) \{Q\}^{n+\theta} = \{0\} ,\end{aligned} \tag{4.92}$$

where $\{Q\}^{n+\theta} = \theta\{Q\}^{n+1} + (1-\theta)\{Q\}^n$. Expanding on a two-dimensional rectangular grid of linear elements,

$$\begin{aligned}\hat{\mathcal{W}}^h(q) = &\left[\frac{\Delta_x \Delta_y}{4} \, \mathtt{B200} + \frac{\alpha_x \bar{U}_x \Delta_t \Delta_y}{2} \, \mathtt{B210} + \frac{\alpha_y \bar{U}_y \Delta_t \Delta_x}{2} \, \mathtt{B220} \right. \\ &+ \frac{\gamma_{xx} \bar{U}_x^2 \Delta_t^2 \Delta_y}{\Delta_x} \, \mathtt{B211} + \gamma_{xy} \bar{U}_x \bar{U}_y \Delta_t^2 (\mathtt{B212} + \mathtt{B221}) \\ &\left. + \frac{\gamma_{yy} \bar{U}_y^2 \Delta_t^2 \Delta_x}{\Delta_y} \, \mathtt{B222} \right] \frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} + \left[\frac{\bar{U}_x \Delta_y}{2} \, \mathtt{B201} + \frac{\bar{U}_y \Delta_x}{2} \, \mathtt{B202} \right. \\ &+ \frac{\beta_{xx} \bar{U}_x^2 \Delta_t \Delta_y}{\Delta_x} \, \mathtt{B211} + \beta_{xy} \bar{U}_x \bar{U}_y \Delta_t^2 (\mathtt{B212} + \mathtt{B221}) + \frac{\beta_{yy} \bar{U}_y^2 \Delta_t \Delta_x}{\Delta_y} \, \mathtt{B222} \\ &\left. + \frac{1}{\mathrm{Pe}} \left(\frac{\Delta_y}{\Delta_x} \, \mathtt{B211} + \frac{\Delta_x}{\Delta_y} \, \mathtt{B222} \right) \right] \{Q\}^{n+\theta} = \{0\} .\end{aligned} \tag{4.93}$$

The one-dimensional results of Chaffin and Baker [17] show that a phase accuracy or time accuracy optimization (typically using γ terms only) should be based on Courant number.

Multiplying through by $\frac{\Delta_t}{4|J|} = \frac{\Delta_t}{\Delta_x \Delta_y}$, the directional Courant number form is

$$\begin{aligned}\hat{\mathcal{W}}^h(q) = &\left[\frac{1}{4}\,\mathrm{B200} + \frac{\alpha_x \bar{C}_x}{2}\,\mathrm{B210} + \frac{\alpha_y \bar{C}_y}{2}\,\mathrm{B220} + \gamma_{xx}\bar{C}_x{}^2\,\mathrm{B211}\right.\\ &\left.+\gamma_{xy}\bar{C}_x\bar{C}_y(\,\mathrm{B212} + \,\mathrm{B221}) + \gamma_{yy}\bar{C}_y{}^2\,\mathrm{B222}\right](\{Q\}^{n+1} - \{Q\}^n)\\ &+\left[\frac{\bar{C}_x}{2}\,\mathrm{B201} + \frac{\bar{C}_y}{2}\,\mathrm{B202} + \beta_{xx}\bar{C}_x{}^2\,\mathrm{B211} + 2\beta_{xy}\bar{C}_x\bar{C}_y(\,\mathrm{B212} + \,\mathrm{B221})\right.\\ &\left.+\beta_{yy}\bar{C}_y{}^2\,\mathrm{B222} + \frac{1}{\mathrm{Pe}}\left(\frac{\Delta_t}{\Delta_x^2}\,\mathrm{B211} + \frac{\Delta_t}{\Delta_y^2}\,\mathrm{B222}\right)\right]\{Q\}^{n+\theta} = \{0\}\,,\end{aligned} \tag{4.94}$$

with θ normally set to $\frac{1}{2}$ for time accuracy.

If convergence to steady-state is desired, the α and γ terms and the Courant number form are typically not used, and (4.93) becomes

$$\begin{aligned}\hat{\mathcal{W}}^h(q) = &\frac{\Delta_x \Delta_y}{4}\,\mathrm{B200}\,\frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} + \Delta_t\left[\frac{\bar{U}_x \Delta_y}{2}\,\mathrm{B201} + \frac{\bar{U}_y \Delta_x}{2}\,\mathrm{B202}\right.\\ &+\frac{\beta_{xx}\bar{U}_x^2 \Delta_t^* \Delta_y}{\Delta_x}\,\mathrm{B211} + 2\beta_{xy}\bar{U}_x\bar{U}_y\Delta_t^*\,\mathrm{B212} + \frac{\beta_{yy}\bar{U}_y^2\Delta_t^*\Delta_x}{\Delta_y}\,\mathrm{B222}\\ &\left.+\frac{1}{\mathrm{Pe}}\left(\frac{\Delta_y}{\Delta_x}\,\mathrm{B211} + \frac{\Delta_x}{\Delta_y}\,\mathrm{B222}\right)\right]\{Q\}^{n+\theta} = \{0\}\,,\end{aligned} \tag{4.95}$$

with θ normally set to 1 for L-stability [4].

The steady-state solution is determined by the terms multiplying $Q^{n+\theta}$ in (4.95), and it is desirable to have those terms independent of the actual time step Δ_t which may change near convergence to steady state. The leading Δ_t on the terms multiplying $Q^{n+\theta}$ is immaterial since it multiplies all of the steady-state terms and does not change the relationship between them. However, the artificial diffusion terms with coefficients β_{ij} include an additional Δ_t, shown in (4.95) as Δ_t^*. To make the steady-state solution independent of Δ_t, it is replaced in the β terms with a dissipation time scale Δ_t^*, dependent on the flow and mesh.

Examples of previously used dissipation time scales include those of Noronha [59],

Freels [36], and Zhang [87], respectively for a $n = 2$ rectangular mesh,

$$\begin{aligned}
\Delta_t^* &\approx \frac{|\mathrm{J}|e^{\frac{1}{n}}}{\|u_j\|} = \left(\frac{\Delta_x\Delta_y}{\bar{U}_x^2+\bar{U}_y^2}\right)^{\frac{1}{2}}, \\
\Delta_t^* &\approx \frac{\Delta_j}{|u_j|} = \frac{\Delta_i}{|\bar{U}_x|+|\bar{U}_y|} \text{ for direction } i\,, \\
\Delta_t^* &\approx \frac{1}{\max|\tilde{u}|_j} = \frac{1}{2\max(\frac{\bar{U}_x}{\Delta_x},\frac{\bar{U}_y}{\Delta_y})}\,,
\end{aligned} \tag{4.96}$$

where Zhang's nondimensional $\tilde{u}$ is transformed to the dimensional form. Additional examples are found in the Petrov-Galerkin literature, where $\bar{k} \sim \Delta_t^*$, including the scale of Tezduyar and Ganjoo [75],

$$\Delta_t^* = \frac{\Delta_x\Delta_y}{2(|\bar{U}_x|\Delta_y+|\bar{U}_y|\Delta_x)}\,, \tag{4.97}$$

and the scale of Swaminathan and Voller [74],

$$\Delta_t^* = \frac{|\bar{U}_x|\Delta_x+|\bar{U}_y|\Delta_y}{2(\bar{U}_x^2+\bar{U}_y^2)}\,. \tag{4.98}$$

A non-linear formulation, which is similar to SUPG when linearized, is that of Iannelli [49], where the linear portion of the dissipation term in the simplified rectangular mesh model is

$$\begin{aligned}
\varphi &= \frac{\partial}{\partial x_i}\left(\varepsilon a_i\frac{\partial f_j}{\partial x_j}\right), \\
a_i &= \frac{U_i}{\|\vec{U}\|}\,, \\
\varepsilon &= \frac{\|\vec{U}\|}{\max(\frac{|U_x|}{\Delta_x},\frac{|U_y|}{\Delta_y})}\,, \\
f_j &= U_j q\,.
\end{aligned} \tag{4.99}$$

Combining the steps of (4.99) yields

$$\varphi = \frac{\partial}{\partial x_i}\left(\frac{\|\vec{U}\|}{\max(\frac{|U_x|}{\Delta_x},\frac{|U_y|}{\Delta_y})}\frac{U_i}{\|\vec{U}\|}U_j\frac{\partial q}{\partial x_j}\right) = \frac{\partial}{\partial x_i}\left(\frac{\bar{U}_i\bar{U}_j}{\max(\frac{|U_x|}{\Delta_x},\frac{|U_y|}{\Delta_y})}\frac{\partial q}{\partial x_j}\right), \tag{4.100}$$

and

$$\Delta_t^* = \frac{1}{\max\left(\frac{|U_x|}{\Delta_x}, \frac{|U_y|}{\Delta_y}\right)} . \tag{4.101}$$

Thus Iannelli's method [49] in rectangular geometry is similar to Zhang's [87] to within a constant factor of $\frac{1}{2}$.

Another candidate time scale may be derived from the steady-state monotone Petrov-Galerkin forms MPG-1 (4.88) and MPG-2 (4.91). Incorporating the Δ_t^* implicitly included in the diffusion coefficients by definition, the MPG-1 form is

$$\begin{aligned} \text{x} - \text{direction}: \ \beta_{xx}\frac{U_x{}^2\Delta_t^*}{\Delta_x^2} &\approx \frac{2|U_x|}{3\Delta_x} + \frac{|U_y|}{3\Delta_y} , \\ \text{y} - \text{direction}: \ \beta_{yy}\frac{U_y{}^2\Delta_t^*}{\Delta_y^2} &\approx \frac{|U_x|}{3\Delta_x} + \frac{2|U_y|}{3\Delta_y} , \end{aligned} \tag{4.102}$$

and for $\beta = O(1)$,

$$\begin{aligned} \text{x} - \text{direction}: \ \Delta_t^* &\approx \frac{\Delta_x}{3|U_x|}\left(2 + \frac{\Delta_x|U_y|}{\Delta_y|U_x|}\right) , \\ \text{y} - \text{direction}: \ \Delta_t^* &\approx \frac{\Delta_y}{3|U_y|}\left(2 + \frac{\Delta_y|U_x|}{\Delta_x|U_y|}\right) . \end{aligned} \tag{4.103}$$

Similarly, the MPG-2 form is

$$\begin{aligned} \text{x} - \text{direction}: \ \Delta_t^* &\approx \frac{\Delta_x}{|U_x|} , \\ \text{y} - \text{direction}: \ \Delta_t^* &\approx \frac{\Delta_y}{|U_y|} . \end{aligned} \tag{4.104}$$

A theory for rigorous analysis of the various dissipation scales is not apparent. However, some special cases may be informative. A limiting case is one-dimensional advection with $U_y = 0$ and $\frac{1}{\text{Pe}} = 0$ in two-dimensional geometry, hence

$$\begin{aligned} \hat{\mathcal{W}}^h(q) = \Delta_y \Bigg[\frac{\Delta_x}{4}\,\texttt{B200}\,\frac{\{Q\}^{n+1} - \{Q\}^n}{\Delta_t} + \Delta_t \Bigg(\frac{\bar{U}_x}{2}\,\texttt{B201} \\ + \frac{\beta_{xx}\bar{U}_x^2\Delta_t^*}{\Delta_x}\,\texttt{B211} \Bigg) \Bigg] \{Q\}^{n+\theta} = \{0\} . \end{aligned} \tag{4.105}$$

The two-dimensional algorithm should deliver the same result as a one-dimensional algo-

rithm in the x-direction for this mesh-aligned flow, thus Δ_t^* should not be dependent on Δ_y in this case. The methods of Noronha [59] and MPG-1 fail this criterion.

Another special case is $\bar{U}_y = \bar{U}_x$, $\Delta_y = \Delta_x$ on a rectangular element. Since the velocity has been increased from the first case in the y-direction, the dissipation scale should be no less than in the previous calculation. The methods of Freels [36] and Tezduyar and Ganjoo [75] fail here.

A third case is $\bar{U}_y = \bar{U}_x$, $\Delta_y = 2\Delta_x$ on a rectangular element. Since the mesh has been coarsened from the second case in the y-direction without changing the velocity, the dissipation scale should again be no less than in the previous calculation. The MPG-1 scale fails in the x-direction. In addition, the y-direction diffusion with half the mesh should be larger than the x-direction diffusion; each bidirectional scale passes while each unidirectional scale fails.

The various dissipation scales and cases are summarized in Table 4.1. Vector-valued scales are shown in parentheses for the x and y directions.

Table 4.1. Summary of time scale calculations

Method	Case 1 $U_y = 0$	Case 2 $(U_y = U_x\,,\ \Delta_y = \Delta_x)$	Case 3 $(U_y = U_x\,, \Delta_y = 2\Delta_x)$
Noronha	$\frac{(\Delta_x\Delta_y)^{\frac{1}{2}}}{\lvert U_x\rvert}$	$\frac{\Delta_x}{\sqrt{2}\lvert U_x\rvert}$	$\frac{\Delta_x}{\lvert U_x\rvert}$
Freels	$(\frac{\Delta_x}{\lvert U_x\rvert}\,,\ \frac{\Delta_y}{\lvert U_x\rvert})$	$(\frac{\Delta_x}{2\lvert U_x\rvert}\,,\ \frac{\Delta_y}{2\lvert U_y\rvert})$	$(\frac{\Delta_x}{2\lvert U_x\rvert}\,,\ \frac{\Delta_y}{\lvert U_y\rvert})$
Zhang [87]	$\frac{\Delta_x}{2\lvert U_x\rvert}$	$\frac{\Delta_x}{2\lvert U_x\rvert}$	$\frac{\Delta_x}{2\lvert U_x\rvert}$
Tezduyar and Ganjoo [75]	$\frac{\Delta_x}{2\lvert U_x\rvert}$	$\frac{\Delta_x}{4\lvert U_x\rvert}$	$\frac{\Delta_x}{3\lvert U_x\rvert}$
Swaminathan and Voller [74],	$\frac{\Delta_x}{2\lvert U_x\rvert}$	$\frac{\Delta_x}{2\lvert U_x\rvert}$	$\frac{3\Delta_x}{4\lvert U_x\rvert}$
MPG-1	$(\frac{2\Delta_x}{3\lvert U_x\rvert}\,,\ 0^*)$	$(\frac{\Delta_x}{\lvert U_x\rvert}\,,\ \frac{\Delta_y}{\lvert U_y\rvert})$	$(\frac{5\Delta_x}{6\lvert U_x\rvert}\,,\ \frac{4\Delta_y}{3\lvert U_y\rvert})$
MPG-2	$(\frac{\Delta_x}{\lvert U_x\rvert}\,,\ 0^*)$	$(\frac{\Delta_x}{\lvert U_x\rvert}\,,\ \frac{\Delta_y}{\lvert U_y\rvert})$	$(\frac{5\Delta_x}{6\lvert U_x\rvert}\,,\ \frac{2\Delta_y}{\lvert U_y\rvert})$

*: after multiplication by U_i^2

The unidirectional methods of Zhang, Iannelli, or Swaminathan appear the most consistent on the limiting cases considered. The bidirectional method MPG-2 appears the most consistent of that group.

Chapter 5

Application of analyses to the finite element method

5.1 Analyses applied to one-dimensional linear basis methods

The one-dimensional linear finite element TWS form from (4.17), rearranged with $\alpha = \mu = 0$, variable θ, and coefficients expressed as Courant numbers, is

$$\begin{aligned}\hat{\mathcal{W}}^h(q) =& \Delta_x\Big(\,\mathtt{A200L} + \gamma C^2\,\mathtt{A211L}\Big)\left(\{Q\}^{n+1} - \{Q\}^n\right)\\ &+\Delta_x\Big(C\,\mathtt{A201L} + \beta C^2\,\mathtt{A211L}\Big)\,\{Q\}^{n+\theta} = \{0\}\,.\end{aligned} \tag{5.1}$$

5.1.1 Frequency analysis of one-dimensional linear methods

The well-known frequency analysis for the linear basis finite element method [42] is established here for later comparison. With element integrals assembled on a uniform grid to an equivalent finite difference stencil, using (3.13) to eliminate $Q_{j\pm1}$ and $\Delta_x = \Delta_e$, the stencils from (2.19) of the matrix integrals for (5.1) are

$$\begin{aligned}
S(\,\mathtt{M200})\,\{Q\} &= S(\mathcal{A}_e(\Delta_e\,\mathtt{A200L}))\,\{Q\} = \frac{\Delta_x}{6}\left|\underline{\overline{\begin{array}{ccc}1 & 4 & 1\end{array}}}\right| \otimes \left|\underline{\overline{\begin{array}{c}e^{-\imath\sigma}\\ 1\\ e^{+\imath\sigma}\end{array}}}\right|\{Q\}\\
S(\Delta_x\,\mathtt{M20X})\,\{Q\} &= S(\mathcal{A}_e(\Delta_x\,\mathtt{A201L}))\,\{Q\} = \frac{\Delta_x}{2}\left|\underline{\overline{\begin{array}{ccc}-1 & 0 & 1\end{array}}}\right| \otimes \left|\underline{\overline{\begin{array}{c}e^{-\imath\sigma}\\ 1\\ e^{+\imath\sigma}\end{array}}}\right|\{Q\}\\
S(\Delta_x^2\,\mathtt{M2XX})\,\{Q\} &= S\left(\mathcal{A}_e(\frac{\Delta_x^2}{\Delta_e}\,\mathtt{A211L})\right)\,\{Q\} = \Delta_x\left|\underline{\overline{\begin{array}{ccc}-1 & 2 & -1\end{array}}}\right| \otimes \left|\underline{\overline{\begin{array}{c}e^{-\imath\sigma}\\ 1\\ e^{+\imath\sigma}\end{array}}}\right|\{Q\}\,.
\end{aligned} \tag{5.2}$$

The assembled matrices A200L, A201L, and A211L are seen to be finite element equivalents of the finite difference operators $(1+\frac{\delta^2}{6})$, (δ), and $(-\delta^2)$ respectively, each scaled by Δ_x. Substituting (5.2) into (3.14) with $\varepsilon^* = 0$ (zero physical diffusion), $\theta = \frac{1}{2}$, $\hat{\gamma} = \gamma C^2$,

and $\hat{\beta} = \beta C^2$ yields the amplification factor,

$$G^h = \frac{(\frac{2}{3}+C-2\hat{\gamma}-\hat{\beta})e^{-\imath\sigma}+(\frac{8}{3}+4\hat{\gamma}+2\hat{\beta})+(\frac{2}{3}-C-2\hat{\gamma}-\hat{\beta})e^{+\imath\sigma}}{(\frac{2}{3}-C-2\hat{\gamma}+\hat{\beta})e^{-\imath\sigma}+(\frac{8}{3}+4\hat{\gamma}-2\hat{\beta})+(\frac{2}{3}+C-2\hat{\gamma}+\hat{\beta})e^{+\imath\sigma}} . \tag{5.3}$$

With $\hat{\beta} = 0$, the phase velocity is, using (3.10) and Macsyma®,

$$\Phi^h = \frac{-2}{\sigma C}\tan^{-1}\left[\frac{3C\sin(\sigma)}{(12\hat{\gamma}-2)\cos(\sigma)-(12\hat{\gamma}+4)}\right] . \tag{5.4}$$

Gresho and Lee [44] developed a similar relation for the GWS, to which (5.4) reduces for $\hat{\gamma} = 0$. In their development, Gresho and Lee used a trigonometric identity which allowed some algebraic simplification (without further approximation) of (5.4) and later (5.24) . Given the assumptions in (5.4) of $\theta = \frac{1}{2}$ and $\hat{\beta} = 0$, the algorithm is non-diffusive, hence the amplification factor can be factored to the form $G^h = \frac{a-\imath}{a+\imath}$ which has the particularly simple complex argument $-\tan^{-1}(\frac{a}{a^2-1})$.

Several finite difference and finite volume methods can be compared to (5.3)-(5.4). Substituting the TWS $\hat{\gamma} = \frac{1}{6}$ into (5.1) effectively converts the interpolation ("mass") matrix `A200L` to a scaled identity matrix, similar to a finite difference method on a uniform mesh. The resulting scheme with $\theta = \frac{1}{2}$ coincides with the well-known Crank-Nicolson finite difference method, with amplification factor

$$G^h = \frac{1-\frac{C}{4}(-e^{-\imath\sigma}+e^{+\imath\sigma})}{1+\frac{C}{4}(-e^{-\imath\sigma}+e^{+\imath\sigma})} . \tag{5.5}$$

Modifications to finite volume methods are usually stated in terms of the convection operator, which in its unmodified form (δ) is identical to the assembled linear basis finite element uniform mesh convection matrix (`A201L`). Among these modified methods are the widely-used group of QUICK finite volume methods ([57], [32]), intended to improve standard finite volume methods via higher order approximations to the convection term. Assuming positive velocity on a uniform 1-D mesh, the third-order upwind QUICK approximation [57] for the first derivative results in the finite difference stencil

$$\delta_{Q3}(Q) = \frac{1}{8}Q_{j-2} - \frac{7}{8}Q_{j-1} + \frac{3}{8}Q_{j+1} + \frac{3}{8}Q_{j+2} , \tag{5.6}$$

designated here as δ_{Q3}. The resulting amplification factor is

$$G^h = \frac{1 - \frac{C}{16}\left(e^{-2\imath\sigma} - 7e^{-\imath\sigma} + 3 + 3e^{+\imath\sigma}\right)}{1 + \frac{C}{16}\left(e^{-2\imath\sigma} - 7e^{-\imath\sigma} + 3 + 3e^{+\imath\sigma}\right)} . \tag{5.7}$$

Assuming positive velocity on a uniform 1-D mesh and neglecting small multi-dimensional terms, the fifth-order upwind QUICK approximation [58] for the first derivative produces the finite difference stencil

$$\begin{aligned}\delta_{Q5}(q) = &-\frac{9}{384}Q_{j-3} + \frac{77}{384}Q_{j-2} - \frac{346}{384}Q_{j-1} + \frac{90}{384}Q_j \\ &+ \frac{211}{384}Q_{j+1} - \frac{23}{384}Q_{j+2} ,\end{aligned} \tag{5.8}$$

designated here as δ_{Q5}. The resulting amplification factor is

$$G^h = \frac{1 - \frac{C}{768}\left(-9e^{-3\imath\sigma} + 77e^{-2\imath\sigma} - 346e^{-\imath\sigma} + 90 + 211e^{+\imath\sigma} - 23e^{+2\imath\sigma}\right)}{1 + \frac{C}{768}\left(-9e^{-3\imath\sigma} + 77e^{-2\imath\sigma} - 346e^{-\imath\sigma} + 90 + 211e^{+\imath\sigma} - 23e^{+2\imath\sigma}\right)} . \tag{5.9}$$

The phase velocity distribution of several finite element, finite difference, and QUICK methods at $C = \frac{1}{8}$ is graphed in Figure 5.1. In each case the trapezoidal rule $\theta = \frac{1}{2}$ was used. The phase velocity should ideally be as near one as possible for the widest possible range of wave number. At this small Courant number, the cubic and quadratic basis finite element methods are the most accurate, with the linear basis finite element method more accurate than any of the finite volume methods. The TWS-γ finite element methods (not shown) are not distinguishable at this small Courant number from Galerkin finite element methods for the same basis degree.

The phase velocity at $C = \frac{1}{2}$ is graphed in Figure 5.2. At this intermediate Courant number, the cubic and quadratic TWS finite element methods remain the most accurate, followed by the GWS methods in order of basis degree, then QUICK and Crank-Nicolson methods.

The phase velocity at $C = 1$ is graphed in Figure 5.3. At this Courant number, the three TWS finite element methods are by far the most accurate, with little difference by basis degree. Again the GWS, QUICK, and Crank-Nicolson methods follow in order of accuracy. The phase velocity of the three TWS methods appears to approach one at wave

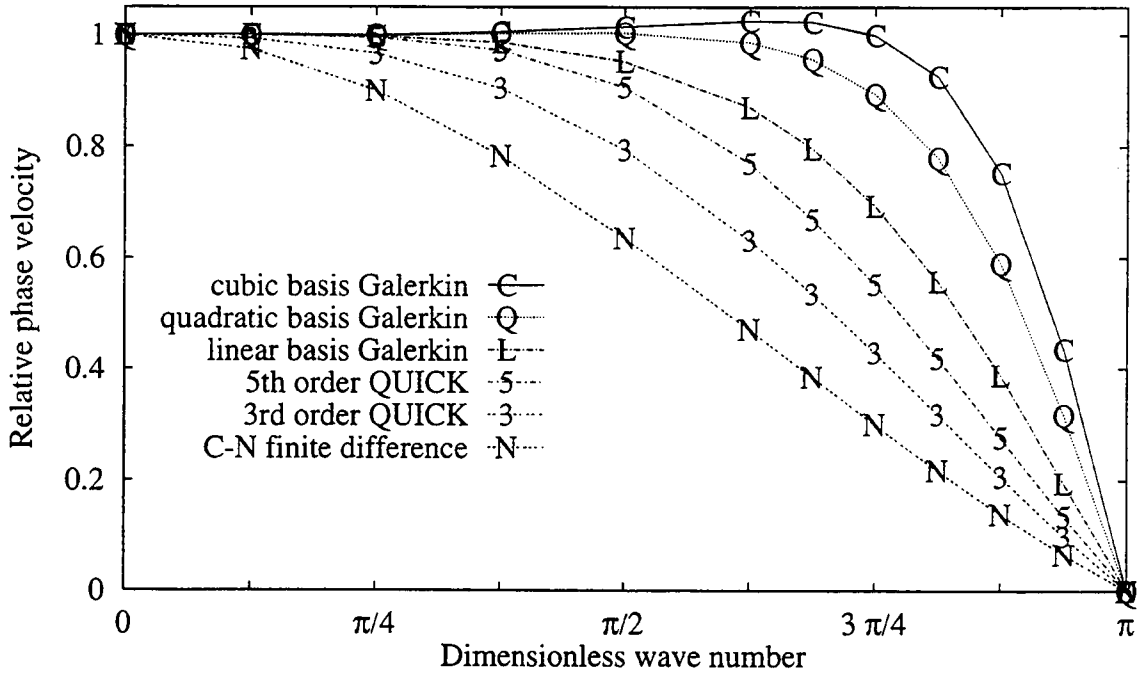

Figure 5.1: Phase velocity distribution for various methods at $C = \frac{1}{8}$

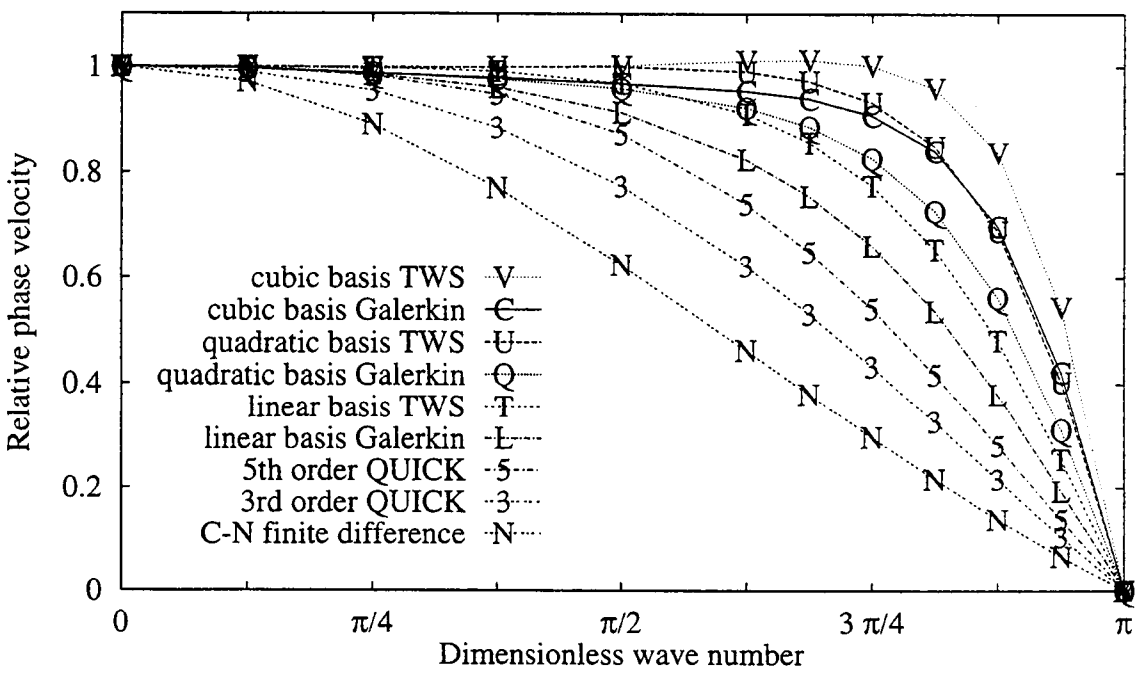

Figure 5.2: Phase velocity distribution for various methods at $C = \frac{1}{2}$

number π; the actual phase velocity near π is a steep approach from one to zero, omitted to reduce visual clutter.

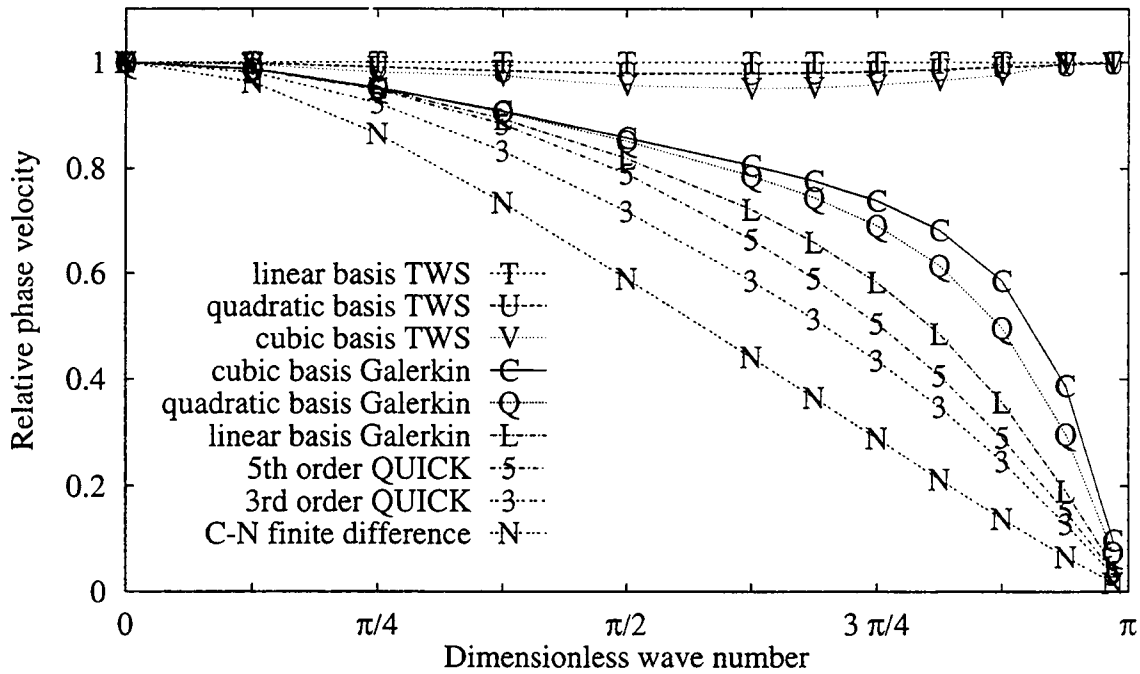

Figure 5.3: Phase velocity distribution for various methods at $C = 1$

5.1.2 Optimization of one-dimensional linear methods for frequency

The phase velocity data confirm that the linear basis TWS correction of $\hat{\gamma} = -\frac{1}{12}$ is most effective for $C = 1$, for which the method has the correct phase velocity over most of the range of wave number σ. This TWS correction does increase accuracy but is comparatively less effective at smaller Courant number.

This can be theoretically quantified by a Taylor expansion of amplification factor G^h in powers of wave number σ about $\sigma = 0$. The resulting expansion for a one-dimen-

sional TWS method with coefficient γ and $\theta = \frac{1}{2}$ is, using Macsyma®,

$$\begin{aligned} G^h = 1 - i\sigma C - \frac{(\sigma C)^2}{2} + \frac{i(\sigma C)^3(1+4\gamma)}{4} + \frac{(\sigma C)^4(1+8\gamma)}{8} \\ - \frac{i\sigma^5 C\left(45C^4 + 60\gamma C^2(1-9C^2) + 720\gamma^2 C^4 - 4\right)}{720} \\ - \frac{\sigma^6 C^2\left(45C^4 + 120\gamma C^2(1-6C^2) + 2160\gamma^2 - 8\right)}{1440} + \cdots, \end{aligned} \tag{5.10}$$

following the procedure of Baker and Kim [9]. The expansion may be compared with the Taylor expansion of the analytical amplification factor $G = \exp(-i\sigma C)$ (3.7),

$$\begin{aligned} G = e^{-i\sigma C} = 1 - i\sigma C - \frac{(\sigma C)^2}{2} + \frac{i(\sigma C)^3}{6} + \frac{(\sigma C)^4}{24} - \frac{i(\sigma C)^5}{120} - \frac{(\sigma C)^6}{720} \\ + \frac{i(\sigma C)^7}{5040} + \frac{(\sigma C)^8}{40320} - \frac{i(\sigma C)^9}{362880} - \frac{(\sigma C)^{10}}{3628800} + \cdots. \end{aligned} \tag{5.11}$$

The analytical series (5.11) converges slowly for wave numbers near π and Courant numbers near or greater than one, e.g., the eighth-order term has a magnitude of about 0.23 and the tenth-order term a magnitude of about 0.03 for $\sigma = \pi$ and $C = 1$. Therefore, an accurate discrete approximation at a Courant number near (but not equal to) one would require about ten to twelve orders of accuracy. Comparing equations (5.10) and (5.11), the Galerkin linear basis method with $\gamma = 0$ is second-order accurate in wave number. The linear basis TWS method with $\gamma = -\frac{1}{12}$ is fourth-order accurate in wave number, except for the special case $C = 1$ which is at least twelfth-order accurate. Since the maximum one-dimensional order of accuracy generally available is one greater than the stencil bandwidth [50], a tenth-order accurate approximation for $C \neq 1$ would require an impractically large 9-bandwidth stencil.

Phase accuracy can be improved by adapting an idea from the early work of Stone and Brian [73]. A two-step method is defined, one step with phase velocity too high, and another step with phase velocity too low, such that mean phase velocity is approximately correct for an even number of steps. Defining a variable TWS correction term γ to be used at alternating timesteps allows equations (5.10) and (5.11) to be equated through order five, yielding

$$\gamma = \frac{-1}{12}\left[1 \pm \frac{1}{C^2}\left(\frac{4 - 5C^2 + C^4}{5}\right)^{\frac{1}{2}}\right]. \tag{5.12}$$

The sixth-order term, also quadratic in γ, cannot be satisfied simultaneously in this form, as it requires

$$\gamma = \frac{-1}{12}\left[1 \pm \frac{1}{C^2}\left(\frac{2(4-5C^2+C^4)}{15}\right)^{\frac{1}{2}}\right] . \tag{5.13}$$

For $C = \frac{1}{2}$ the solution of the fifth-order expression (5.12) is

$$\gamma = \frac{-1}{12}(1 \pm 3) . \tag{5.14}$$

This method will be referred to here as the "fifth-order" linear basis TWS method, not to be confused with the differently measured order of the "fifth-order QUICK" method.

The relative phase velocity of the fifth-order linear basis TWS method is shown by individual steps in Figure 5.4. The relative phase velocity of the "odd" steps is higher than

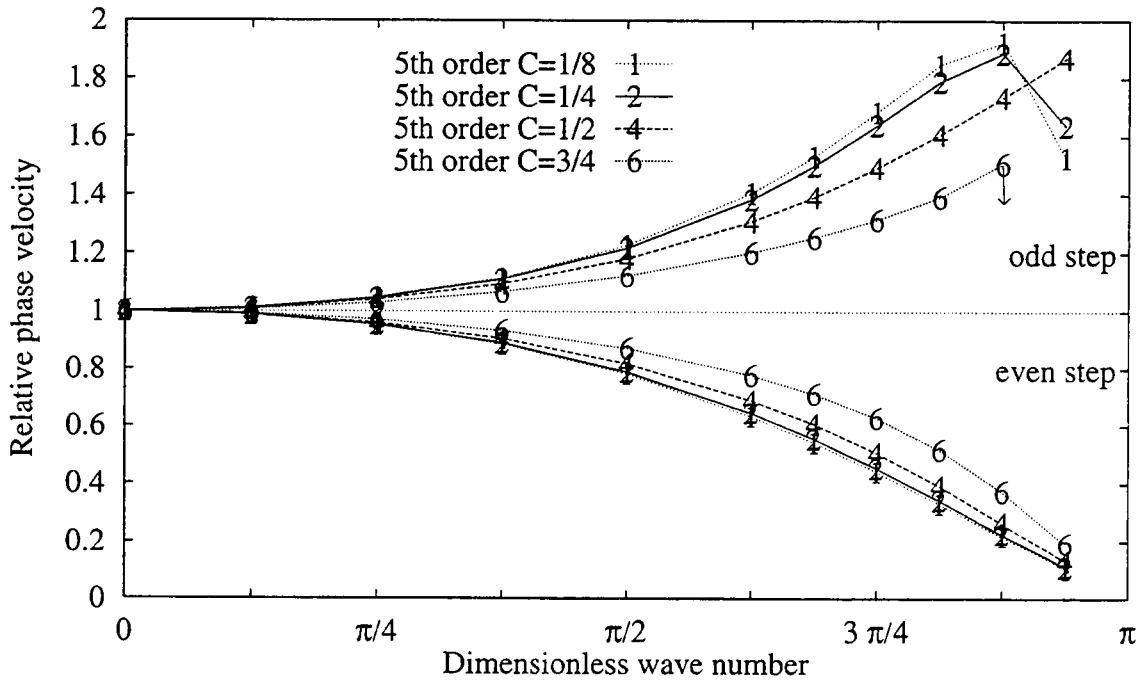

Figure 5.4: Phase velocity of two-step fifth-order algorithm

one, decreasing with Courant number. The phase velocity of the "even" steps is smaller than one, increasing with Courant number, such that the mean phase velocity is nearly one. The resulting mean phase velocity is shown in Figure 5.5 and compared to those of the fourth-order TWS method. Figure 5.5 shows that the fifth-order mean phase velocity

is high for $C < \frac{1}{2}$, correct for $C = \frac{1}{2}$, and low for $C > \frac{1}{2}$, but is higher than the fourth-order TWS phase velocity at each Courant number.

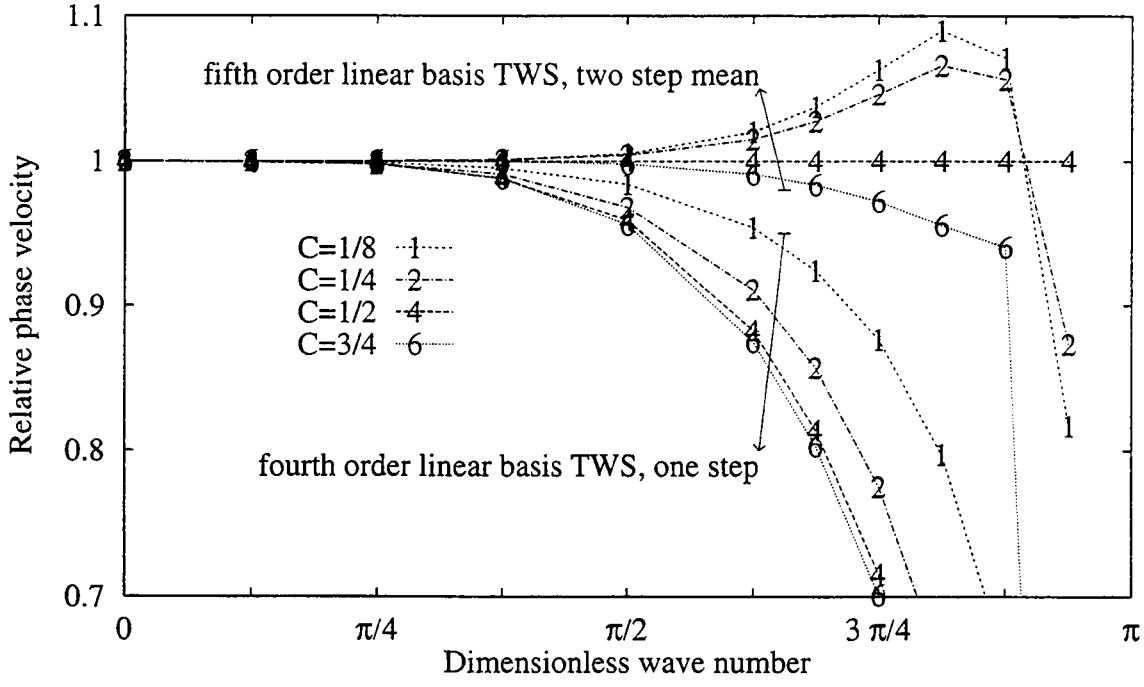

Figure 5.5: Phase velocity comparison, linear basis fifth-order mean and fourth-order

5.1.3 Optimization of one-dimensional methods for phase velocity

The preceding phase velocity results (and verification problem results to be discussed) show that the fifth-order method, while improved over the standard fourth-order method, is not optimal except at $C = \frac{1}{2}$. Higher-order matching of the amplification factor expansion does not appear practical for the ten or so significant terms at Courant numbers near one. The fifth-order method has the additional disadvantage of lacking diagonal dominance for $C > \frac{1}{2}$.

An alternative approach is to minimize the error in phase velocity rather than exactly matching a low-order expansion of the amplification factor. A multiple-step method is used with two steps for $C < \frac{1}{2}$ and four steps for $C > \frac{1}{2}$. The phase error from (5.4) does not appear to have a known analytical integral over σ, so the phase error is numerically integrated over σ from 0 to π for many possible values of γ, and the multiple steps are

combined so as to minimize phase error. Following the somewhat arbitrary accuracy criterion of Vichnevetsky and De Schutter [79], the numerical solution (γ_1, γ_2) is used which maintains 1% phase accuracy over the widest possible range of σ. This range turns out to be $\sigma = 0$ to about $\frac{3\pi}{4}$ for most values of C.

The individual steps are combined as (γ_1, γ_2) for two-step solutions ($C < \frac{1}{2}$) and $(\gamma_1, \gamma_1, \gamma_1, \gamma_2)$ for four-step solutions ($C > \frac{1}{2}$). The selected values of γ at various Courant numbers are then curve-fit to Courant number for $C \leq \frac{1}{2}$ as

$$(\gamma_1, \gamma_2) = \frac{-1}{12} \pm \left(\frac{0.07025 - 0.02609\,C^2 - 0.00982\,C^3}{C^2} \right), \tag{5.15}$$

and for $C > \frac{1}{2}$ as

$$\gamma_1 = -\min\left(\frac{1}{12}, \frac{2C+1}{30}\right), \tag{5.16}$$

with, for $\frac{1}{2} \leq C \leq \frac{3}{4}$:

$$\gamma_2 = -0.061 + 0.793\,(1-C) - 0.792\,(1-C)^2, \tag{5.17}$$

and for $\frac{3}{4} \leq C \leq 1$:

$$\gamma_2 = -\frac{1}{12} + 0.3427\,(1-C) + 1.3364\,(1-C)^2. \tag{5.18}$$

This approximate procedure is not presented to be the "best" solution of this type, but as a "good" solution for phase accuracy. The resulting phase velocity, the mean of two or four steps, is shown in Figure 5.6 for various Courant number. Note the highly expanded ordinate scale, within one percent accurate, and the difficulty of maintaining the phase accuracy for wavenumbers above $\frac{3\pi}{4}$.

The preceding figures show that the approximately optimized method maintains better than 1% accuracy in phase velocity up to wave numbers of at least $\frac{3\pi}{4}$ for the range of $0 \leq C \leq 1$. The method is superior in phase accuracy over most of that range compared to the fourth-order and fifth-order linear basis TWS methods.

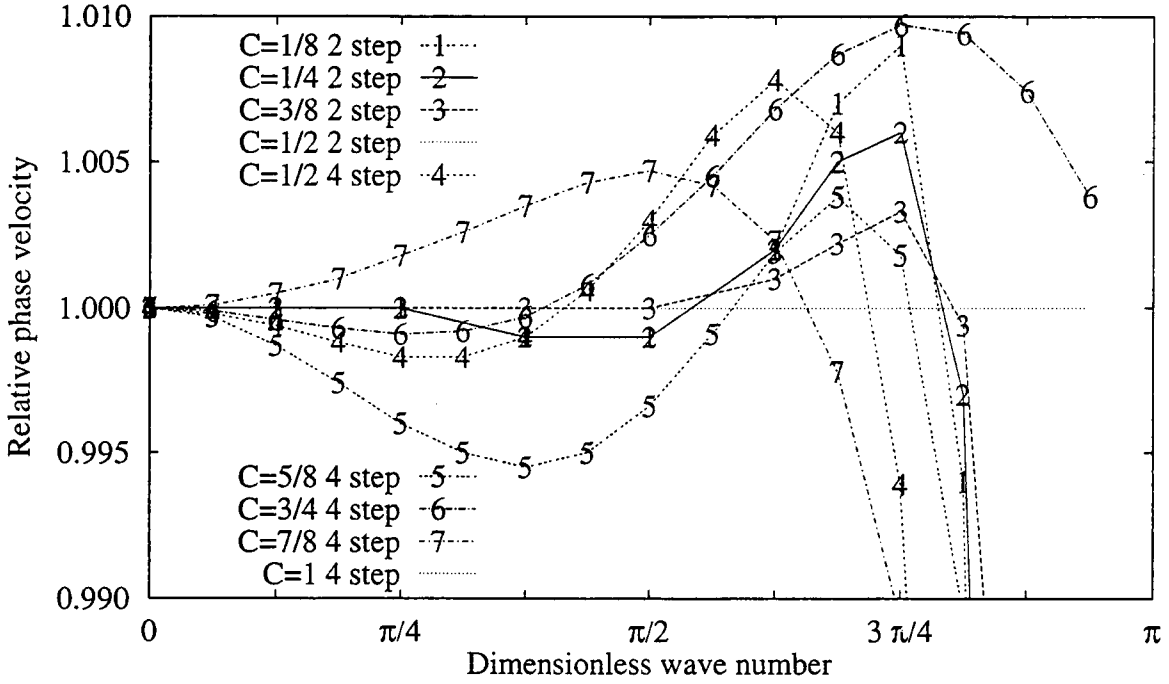

Figure 5.6: Phase velocity of optimized method at various Courant numbers

5.2 Analyses applied to one-dimensional quadratic basis methods

Assembled Lagrange quadratic basis finite element stencils are shown below on a uniform grid of $\Delta_e = 2\Delta_x$. The assembled stencils are given on the right-hand side, with the upper row in each two-row matrix representing an element-vertex node and the lower row a mid-element node.

$$\mathcal{S}(\,\mathtt{M200})\,\{Q\} = \mathcal{S}(\mathcal{A}_e(\Delta_e\,\mathtt{A200Q}))\,\{Q\}$$

$$= \frac{\Delta_x}{15}\begin{vmatrix} -1 & 2 & 8 & 2 & -1 \\ 0 & 0 & 2 & 16 & 2 \end{vmatrix} \otimes \begin{vmatrix} e^{-2\imath\sigma} \\ e^{-\imath\sigma} \\ 1 \\ e^{+\imath\sigma} \\ e^{+2\imath\sigma} \end{vmatrix} \{Q\}\,, \tag{5.19}$$

$$S(\Delta_x\,\mathtt{M20X})\,\{Q\} = S(\mathcal{A}_e(\Delta_x\,\mathtt{A201Q}))\,\{Q\}$$

$$= \frac{\Delta_x}{6}\left|\begin{array}{ccccc} 1 & -4 & 0 & 4 & -1 \\ 0 & 0 & -4 & 0 & -4 \end{array}\right| \otimes \left|\begin{array}{c} e^{-2\imath\sigma} \\ e^{-\imath\sigma} \\ 1 \\ e^{+\imath\sigma} \\ e^{+2\imath\sigma} \end{array}\right| \{Q\}\,, \tag{5.20}$$

$$S(\Delta_x^2\,\mathtt{M2XX})\,\{Q\} = S\left(\mathcal{A}_e(\frac{\Delta_x^2}{\Delta_e}\,\mathtt{A211Q})\right)\,\{Q\}$$

$$= \frac{\Delta_x}{6}\left|\begin{array}{ccccc} 1 & -8 & 8 & -8 & 1 \\ 0 & 0 & -8 & 16 & -8 \end{array}\right| \otimes \left|\begin{array}{c} e^{-2\imath\sigma} \\ e^{-\imath\sigma} \\ 1 \\ e^{+\imath\sigma} \\ e^{+2\imath\sigma} \end{array}\right| \{Q\} \tag{5.21}$$

Dividing through by Δ_x and following Wait and Mitchell [82] to compute the amplification factor, the Fourier stencil matrices are

$$\begin{aligned} S(\,\mathtt{A200Q}) &= \frac{1}{15}\left|\begin{array}{cc} 8-(e^{+2\imath\sigma}+e^{-2\imath\sigma}) & 2(e^{+\imath\sigma}+e^{-\imath\sigma}) \\ 2(e^{+\imath\sigma}+e^{-\imath\sigma}) & 16 \end{array}\right|, \\ S(\,\mathtt{A201Q}) &= \frac{1}{6}\left|\begin{array}{cc} -(e^{+2\imath\sigma}-e^{-2\imath\sigma}) & 4(e^{+\imath\sigma}-e^{-\imath\sigma}) \\ 4(e^{+\imath\sigma}-e^{-\imath\sigma}) & 0 \end{array}\right|, \\ S(\,\mathtt{A211Q}) &= \frac{1}{6}\left|\begin{array}{cc} 14+(e^{+2\imath\sigma}+e^{-2\imath\sigma}) & -8(e^{+\imath\sigma}+e^{-\imath\sigma}) \\ -8(e^{+\imath\sigma}+e^{-\imath\sigma}) & 16 \end{array}\right|. \end{aligned} \tag{5.22}$$

The amplification factor becomes a complex-valued 2x2 matrix, summed from (5.22), yielding

$$G^h = \left(\,\mathtt{A200Q} + \frac{C}{2}\,\mathtt{A201Q} + \hat{\gamma}\,\mathtt{A211Q}\right)^{-1}\left(\,\mathtt{A200Q} - \frac{C}{2}\,\mathtt{A201Q} + \hat{\gamma}\,\mathtt{A211Q}\right). \tag{5.23}$$

With $\beta = 0$, the phase velocity is

$$\Phi^h = \frac{2}{\sigma C}\tan^{-1}\Big[C\sin(\sigma) \Big(\frac{\sqrt{(2+5\hat{\gamma})[18+45\hat{\gamma}+(2+15\hat{\gamma})\sin^2(\sigma)]} \pm (4-15\hat{\gamma})\cos(\sigma)}{4[(1-4\hat{\gamma}+15\hat{\gamma}^2)\sin^2(\sigma)+(1+15\hat{\gamma})]}\Big)\Big], \tag{5.24}$$

with the principal root having the negative sign.

5.3 Frequency analysis of one-dimensional cubic basis methods

The analysis for the cubic (superscript c) basis finite element method is similar to that for quadratic basis, except that the cubic Lagrange element spans four nodes with differing Fourier stencils for vertex nodes and for two interior nodes, shown respectively as the rows of the matrices following. A uniform mesh is assumed with $\Delta_e = 3\Delta_x$ such that

$$\mathcal{S}(\texttt{M200})\{Q\} = \mathcal{S}(\mathcal{A}_e(\Delta_e\,\texttt{A200C}))\{Q\}$$

$$= \frac{\Delta_x}{560}\left|\begin{array}{ccccccc} 19 & -36 & 99 & 256 & 99 & -36 & 19 \\ 0 & 0 & 0 & 99 & 648 & -81 & -36 \\ 0 & 0 & 0 & -36 & -81 & 648 & 99 \end{array}\right| \otimes \left|\begin{array}{c} e^{-3\imath\sigma} \\ e^{-2\imath\sigma} \\ e^{-\imath\sigma} \\ 1 \\ e^{+\imath\sigma} \\ e^{+2\imath\sigma} \\ e^{+3\imath\sigma} \end{array}\right| \{Q\}, \tag{5.25}$$

$$\mathcal{S}(\Delta_x\,\texttt{M20X})\{Q\} = \mathcal{S}(\mathcal{A}_e(\Delta_x\,\texttt{A201C}))\{Q\}$$

$$= \frac{\Delta_x}{80}\left|\begin{array}{ccccccc} -7 & 24 & -56 & 0 & 56 & -24 & 7 \\ 0 & 0 & 0 & -56 & 0 & 81 & -24 \\ 0 & 0 & 0 & 24 & -81 & 0 & 56 \end{array}\right| \otimes \left|\begin{array}{c} e^{-3\imath\sigma} \\ e^{-2\imath\sigma} \\ e^{-\imath\sigma} \\ 1 \\ e^{+\imath\sigma} \\ e^{+2\imath\sigma} \\ e^{+3\imath\sigma} \end{array}\right| \{Q\}, \tag{5.26}$$

$$S(\Delta_x^2\,\texttt{M2XX})\,\{Q\} = S\left(\mathcal{A}_e(\frac{\Delta_x^2}{\Delta_e}\,\texttt{A211C})\right)\,\{Q\}$$

$$= \frac{\Delta_x}{120}\begin{vmatrix} -13 & 54 & -189 & 296 & -189 & 54 & -13 \\ 0 & 0 & 0 & -189 & 432 & -297 & 54 \\ 0 & 0 & 0 & -297 & 54 & 432 & -189 \end{vmatrix} \otimes \begin{vmatrix} e^{-3\imath\sigma} \\ e^{-2\imath\sigma} \\ e^{-\imath\sigma} \\ 1 \\ e^{+\imath\sigma} \\ e^{+2\imath\sigma} \\ e^{+3\imath\sigma} \end{vmatrix}\,\{Q\}\,. \tag{5.27}$$

The matrix form of the amplification factor is similar to that for the quadratic basis (5.24), except that the matrix stencils become 3x3 matrices, i.e.,

$$\texttt{A200C} = \frac{\Delta_x}{560}\begin{vmatrix} 256+19(e^{+3\imath\sigma}+e^{-3\imath\sigma}) & -36e^{+2\imath\sigma}+99e^{-\imath\sigma} & 99e^{+\imath\sigma}-36e^{-2\imath\sigma} \\ 99e^{+\imath\sigma}-36e^{-2\imath\sigma} & 648 & -81e^{-\imath\sigma} \\ -36e^{+2\imath\sigma}+99e^{-\imath\sigma} & -81e^{+\imath\sigma} & 648 \end{vmatrix},$$

$$\texttt{A201C} = \frac{\Delta_x}{80}\begin{vmatrix} 7(e^{+3\imath\sigma}-e^{-3\imath\sigma}) & -24e^{+2\imath\sigma}-56e^{-\imath\sigma} & 56e^{+\imath\sigma}+24e^{-2\imath\sigma} \\ 56e^{+\imath\sigma}+24e^{-2\imath\sigma} & 0 & -81e^{-\imath\sigma} \\ -24e^{+2\imath\sigma}-56e^{-\imath\sigma} & 81e^{+\imath\sigma} & 0 \end{vmatrix},$$

$$\texttt{A211C} = \frac{\Delta_x}{120}\begin{vmatrix} 296-13(e^{+3\imath\sigma}+e^{-3\imath\sigma}) & 54e^{+2\imath\sigma}-189e^{-\imath\sigma} & -189e^{+\imath\sigma}+54e^{-2\imath\sigma} \\ -189e^{+\imath\sigma}+54e^{-2\imath\sigma} & 432 & -297e^{-\imath\sigma} \\ 54e^{+2\imath\sigma}-189e^{-\imath\sigma} & -297e^{+\imath\sigma} & 432 \end{vmatrix}. \tag{5.28}$$

The phase velocity was calculated using Macsyma®, but the calculated form is some hundreds of lines of output and is not shown. The phase velocity was calculated at discrete values of C and σ, with an example calculation shown in Appendix V.

5.4 Analyses applied to two-dimensional methods

5.4.1 Optimization of two-dimensional methods by modified equation

With Courant numbers C_x and C_y, the two-dimensional explicit Lax-Wendroff fi-

nite difference method is, expanding (1.13) to two dimensions,

$$\begin{aligned}\mathcal{L}(\{Q\}) = &\left|\begin{matrix}0&0&0\\0&1&0\\0&0&0\end{matrix}\right|(\{Q\}^{n+1}-\{Q\}^n)\\ &+\frac{C_x}{2}\left|\begin{matrix}0&0&0\\-1&0&1\\0&0&0\end{matrix}\right|\{Q\}^n+\frac{C_y}{2}\left|\begin{matrix}0&1&0\\0&0&0\\0&-1&0\end{matrix}\right|\{Q\}^n\\ &-\frac{C_x^2}{2}\left|\begin{matrix}0&0&0\\1&-2&1\\0&0&0\end{matrix}\right|\{Q\}^n-\frac{C_y^2}{2}\left|\begin{matrix}0&1&0\\0&-2&0\\0&1&0\end{matrix}\right|\{Q\}^n.\end{aligned} \tag{5.29}$$

Each discrete value is replaced by a Taylor series (3.25) about the central value, and substituted into (3.34) to yield the modified equation

$$\begin{aligned}T+C_xX+C_yY = &-C_xC_yXY-\frac{C_yC_x^2}{2}X^2Y-\frac{C_xC_y^2}{2}Y^2X\\ &+\frac{C_x(C_x^2-1)}{6}X^3+\frac{C_y(C_y^2-1)}{6}Y^3\\ &+\frac{C_x^2(3C_x^2-4)}{24}X^4+\frac{C_y^2(3C_y^2-4)}{24}Y^4\\ &-\frac{C_xC_y}{6}Y^3X-\frac{C_yC_x}{6}X^3Y-\frac{3C_y^2C_x^2}{4}X^2Y^2+\cdots.\end{aligned} \tag{5.30}$$

The leading even-order error is the mixed derivative of order two with negative sign, implying possible instability. Fourth order derivatives are negative, hence stable for $C < (\frac{4}{3})^{\frac{1}{2}}$.

A common Galerkin finite element method is the two-dimensional $\theta-$ implicit bilinear construction,

$$\mathtt{B200}\,(\{Q\}^{n+1}-\{Q\}^n)+(C_x\,\mathtt{B201}+C_y\,\mathtt{B202})\Big(\theta\{Q\}^{n+1}+(1-\theta)\{Q\}^n\Big)=0. \tag{5.31}$$

Its modified equation is

$$\begin{aligned} T + C_x X + C_y Y &= 2(2\theta - 1)(C_x^2 X^2 + 2C_x C_y XY + C_y^2 Y^2) \\ &+ 8(-3\theta^2 + 3\theta - 1)(C_x^3 X^3 + 3C_x^2 C_y X^2 Y + 3C_x C_y^2 Y^2 X + C_y^3 Y^3) \\ &+ 4(2\theta - 1)(2\theta^2 - 2\theta + 1)(C_x^4 X^4 + 4C_x^3 C_y X^3 Y \\ &+ 6C_x^2 C_y^2 X^2 Y^2 + 4C_x C_y^3 Y^3 X + C_y^4 Y^4) + \cdots . \end{aligned} \tag{5.32}$$

Stability properties are less than optimum, as the second order error term requires $\theta \geq \frac{1}{2}$ for positivity, which makes the fourth order term also positive or zero. Both second and fourth order terms are completely eliminated by $\theta = \frac{1}{2}$. The third order term cannot be eliminated by a real value of θ.

These properties can be improved by repeating the modified equation analysis for a TWS finite element method with arbitrary directional artificial diffusion coefficients β_{ij}, γ_{ij} to be determined, beginning with

$$\begin{aligned} &(\,\mathrm{B200} + \gamma_{xx} C_x{}^2\,\mathrm{B211} + 2\gamma_{xy} C_x C_y\,\mathrm{B212} \\ &+ \gamma_{yy} C_y{}^2\,\mathrm{B222})(\{Q\}^{n+1} - \{Q\}^n) + (C_x\,\mathrm{B201} + C_y\,\mathrm{B202} + \beta_{xx} C_x{}^2\,\mathrm{B211} \\ &+ 2\beta_{xy} C_x C_y\,\mathrm{B212} + \beta_{yy} C_y{}^2\,\mathrm{B222})\left(\theta\{Q\}^{n+1} + (1-\theta)\{Q\}^n\right) = \{0\}. \end{aligned} \tag{5.33}$$

The modified equation of the TWS form, using (3.34) and (5.33) is,

$$\begin{aligned} T + C_x X + C_y Y &= \\ &(\theta - \frac{1}{2})(C_x^2 X^2 + 2C_x C_y XY + C_y^2 Y^2) \\ &+ (\beta_{xx} C_x^2 X^2 + \beta_{xy} C_x C_y XY + \beta_{yy} C_y^2 Y^2) \\ &- C_x^3\left((\theta - 1)(\theta + \beta_{xx}) + \gamma_{xx} + \frac{1}{3}\right) X^3 \\ &- C_x^2 C_y\left((\theta - 1)(3\theta + \beta_{xx} + \beta_{xy}) + \gamma_{xx} + \gamma_{xy} + 1\right) X^2 Y \\ &- C_y^2 C_x\left((\theta - 1)(3\theta + \beta_{yy} + \beta_{xy}) + \gamma_{yy} + \gamma_{xy} + 1\right) XY^2 \\ &- C_y^3\left((\theta - 1)(\theta + \beta_{yy}) + \gamma_{yy} + \frac{1}{3}\right) Y^3 + \cdots . \end{aligned} \tag{5.34}$$

It should represent a method optimized for stability. The modified equation is

$$\begin{aligned} T + C_x X + C_y Y &= \varepsilon (C_x^2 X^2 + 2 C_x C_y XY + C_y^2 Y^2) \\ &+ \frac{C_x (C_x - 1)(C_x + 1)}{6} X^3 + \frac{C_y (C_y - 1)(C_y + 1)}{6} Y^3 \\ &+ \frac{C_y (3C_x^2 - 1)}{6} X^2 Y + \frac{C_x (3C_y^2 - 1)}{6} Y^2 X + \cdots , \end{aligned} \tag{5.38}$$

and it should be linearly stable for any Courant number with positive ε.

Chapter 6

Nonlinear equation systems and finite element methods

6.1 Burgers equation

The Burgers equation has been widely used as a simple model for the nonlinearity of the Euler and Navier-Stokes equations. The inviscid Burgers equation in two dimensions is

$$\begin{aligned} \frac{\partial u_x}{\partial t} + u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y} = 0 \,, \\ \frac{\partial u_y}{\partial t} + u_x \frac{\partial u_y}{\partial x} + u_y \frac{\partial u_y}{\partial y} = 0 \,, \end{aligned} \tag{6.1}$$

and viscous terms may be added.

6.1.1 Finite element method for Burgers equation

A finite element method is developed for (6.1), following (2.14) for the advection-diffusion equation, and yielding

$$\begin{aligned} \mathcal{W}^h(\{U_x\}) = \texttt{M200} \frac{1}{\Delta_t} (\{U_x\}^{n+1} - \{U_x\}^n) + (\{U_k\}^T \, \texttt{M300K} \, \{U_x\})^{n+\theta} = \{0\} \,, \\ \mathcal{W}^h(\{U_y\}) = \texttt{M200} \frac{1}{\Delta_t} (\{U_y\}^{n+1} - \{U_y\}^n) + (\{U_k\}^T \, \texttt{M300K} \, \{U_y\})^{n+\theta} = \{0\} \,. \end{aligned} \tag{6.2}$$

Because of its nonlinearity, (6.2) is not directly solvable and some form of iteration or linearization procedure is necessary. Here, the discrete form (6.2) is linearized using a Newton method, while in other applications the equation system (6.1) may be linearized instead. A Newton method for a matrix equation system $\mathcal{W}^h(\{Q\}) = \{0\}$ is [8]

$$\frac{\partial \mathcal{W}^h(\{Q\}_p^{n+1})}{\partial \{Q\}_p^{n+1}} (\{Q\}_{p+1}^{n+1} - \{Q\}_p^{n+1}) = -\mathcal{W}^h(\{Q\}_p^{n+1}) \,, \tag{6.3}$$

where p is an iteration index. The Newton method will be quadratically convergent if the Jacobian derivatives can be obtained analytically, as is the case for Burgers and most Navier-Stokes equation systems. A procedure is required for taking chain-rule derivatives

of a discrete nonlinear matrix; following [8] the derivative with respect to $\{U\}$ of the matrix product $(\{U\}^T \texttt{M300X}\{U\})$ is $\{U\}^T \texttt{M300X}[I]$ with respect to the right-hand $\{U\}$. The matrix product is equivalently $(\{U\}^T \texttt{M3X00}\{U\})$, hence the derivative is $\{U\}^T \texttt{M3X00}[I]$ and the chain-rule sum is $\{U\}^T(\texttt{M3X00} + \texttt{M300X})[I]$. Applying the method to (6.2) with $\mathcal{W}^h = (\mathcal{W}^h(\{U_x\}), \mathcal{W}^h(\{U_y\}))$ and $\{Q\} = (\{U_x\}^{n+1}, \{U_y\}^{n+1})$ gives the entries of the Jacobian matrix $\frac{\partial \mathcal{W}^h(\{Q\}_p^{n+1})}{\partial \{Q\}_p^{n+1}}$,

$$
\begin{aligned}
\frac{\partial \mathcal{W}_1^h}{\partial \{Q_1\}} &= \frac{\partial \mathcal{W}^h(\{U_x\})}{\partial \{U_x\}^{n+1}} = \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U_k\}^T \texttt{M300K} + \{U_x\}^T \texttt{M3X00}\Big)\,, \\
\frac{\partial \mathcal{W}_1^h}{\partial \{Q_2\}} &= \frac{\partial \mathcal{W}^h(\{U_x\})}{\partial \{U_y\}^{n+1}} = \theta\{U_y\}^T \texttt{M3Y00}\,, \\
\frac{\partial \mathcal{W}_2^h}{\partial \{Q_1\}} &= \frac{\partial \mathcal{W}^h(\{U_y\})}{\partial \{U_x\}^{n+1}} = \theta\{U_x\}^T \texttt{M3X00}\,, \\
\frac{\partial \mathcal{W}_2^h}{\partial \{Q_2\}} &= \frac{\partial \mathcal{W}^h(\{U_y\})}{\partial \{U_y\}^{n+1}} = \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U_k\}^T \texttt{M300K} + \{U_y\}^T \texttt{M3Y00}\Big)\,.
\end{aligned}
\tag{6.4}
$$

A disadvantage of the "full" Newton method using (6.4) is that the eventual solution matrix bandwidth expands by a factor n^2 for a n-equation system compared to a scalar equation. Various approximations to that system, or quasi-Newton methods, are generally used in practice, at a loss of the quadratic convergence. In this research, the n equations were solved sequentially in a segregated manner using only the diagonal equations in the Jacobian matrix, i.e. the first and last entries of (6.4) for the Burgers equation.

6.1.2 TWS method for Burgers equation

The TWS second time derivatives of Burgers equation in the x direction are

$$\begin{aligned}
\frac{\partial^2 u_x}{\partial t^2} &= -\frac{\partial}{\partial t}(u_x\frac{\partial u_x}{\partial x}) - \frac{\partial}{\partial t}(u_y\frac{\partial u_x}{\partial y}) \\
&= -u_x\frac{\partial}{\partial t}\frac{\partial u_x}{\partial x} - \frac{\partial u_x}{\partial t}\frac{\partial u_x}{\partial x} - u_y\frac{\partial}{\partial t}\frac{\partial u_x}{\partial y} - \frac{\partial u_y}{\partial t}\frac{\partial u_x}{\partial y} - \frac{\partial u_y}{\partial y}\frac{\partial u_x}{\partial t} + \frac{\partial u_y}{\partial y}\frac{\partial u_x}{\partial t} \\
&= -\frac{\partial}{\partial x}(u_x\frac{\partial u_x}{\partial t}) - \frac{\partial}{\partial y}(u_y\frac{\partial u_x}{\partial t}) - \frac{\partial u_x}{\partial y}\frac{\partial u_y}{\partial t} + \frac{\partial u_y}{\partial y}\frac{\partial u_x}{\partial t} \\
&= +\frac{\partial}{\partial x}\left[u_x(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y})\right] + \frac{\partial}{\partial y}\left[u_y(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y})\right] \\
&\quad + \frac{\partial u_x}{\partial y}(u_y\frac{\partial u_y}{\partial y} + u_x\frac{\partial u_y}{\partial x}) - \frac{\partial u_y}{\partial y}(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y}) \\
&= \frac{\partial}{\partial x}\left[u_x(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y})\right] + \frac{\partial}{\partial y}\left[u_y(u_x\frac{\partial u_x}{\partial x} + u_y\frac{\partial u_x}{\partial y})\right] + \frac{\partial u_y}{\partial x}u_x\frac{\partial u_x}{\partial y} - \frac{\partial u_y}{\partial y}u_x\frac{\partial u_x}{\partial x}.
\end{aligned} \tag{6.5}$$

The discrete form of the TWS terms, neglecting boundary integrals as before, is

$$\begin{aligned}
&\int_{\Omega^h}\{\chi\}\frac{\partial^2 u}{\partial t^2}\,d\Omega^h \approx \\
&\int_{\Omega^h}\{\chi\}\Bigg\{\frac{\partial}{\partial x}\left[\{\psi\}^T\{U_x\}\left(\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\} + \{\psi\}^T\{U_y\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\}\right)\right] \\
&+ \frac{\partial}{\partial y}\left[\{\psi\}^T\{U_y\}\left(\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\} + \{\psi\}^T\{U_y\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\}\right)\right] \\
&+ \frac{\partial\{\psi\}^T}{\partial x}\{U_y\}\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\} - \frac{\partial\{\psi\}^T}{\partial y}\{U_y\}\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\}\Bigg\}\,d\Omega^h.
\end{aligned} \tag{6.6}$$

Integrating by parts the terms in brackets yields

$$
\begin{aligned}
&\int_{\Omega^h}\{\chi\}\frac{\partial^2 u}{\partial t^2}\,d\Omega^h \approx \\
&-\int_{\Omega^h}\frac{\partial\{\chi\}}{\partial x}\left(\{\psi\}^T\{U_x\}\left(\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\}+\{\psi\}^T\{U_y\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\}\right)\right)d\Omega^h \\
&-\int_{\Omega^h}\frac{\partial\{\chi\}}{\partial y}\left(\{\psi\}^T\{U_y\}\left(\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\}+\{\psi\}^T\{U_y\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\}\right)\right)d\Omega^h \\
&-\int_{\Omega^h}\{\chi\}\frac{\partial\{\psi\}^T}{\partial x}\{U_y\}\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\} \\
&+\int_{\Omega^h}\{\chi\}\frac{\partial\{\psi\}^T}{\partial y}\{U_y\}\{\psi\}^T\{U_x\}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\} \\
=&-\{U_x\}^T\{U_x\}^T\int_{\Omega^h}\{\psi\}\{\psi\}\frac{\partial\{\chi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\}\,d\Omega^h \\
&-\{U_x\}^T\{U_x\}^T\int_{\Omega^h}\{\psi\}\{\psi\}\frac{\partial\{\chi\}}{\partial x}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\}\,d\Omega^h \\
&-\{U_x\}^T\{U_y\}^T\int_{\Omega^h}\{\psi\}\{\psi\}\frac{\partial\{\chi\}}{\partial y}\frac{\partial\{\psi\}^T}{\partial x}\{U_x\}\,d\Omega^h \\
&-\{U_y\}^T\{U_y\}^T\int_{\Omega^h}\{\psi\}\{\psi\}\frac{\partial\{\chi\}}{\partial y}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\}\,d\Omega^h \\
&-\{U_y\}^T\{U_x\}^T\int_{\Omega^h}\{\psi\}\frac{\partial\{\psi\}}{\partial x}\{\chi\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\} \\
&+\{U_y\}^T\{U_x\}^T\int_{\Omega^h}\{\psi\}\frac{\partial\{\psi\}}{\partial y}\{\chi\}\frac{\partial\{\psi\}^T}{\partial y}\{U_x\} \\
=&-\Big[\{U_x\}^T\{U_x\}^T\,\mathtt{M400XX}+\{U_y\}^T\{U_x\}^T\,\mathtt{M400XY} \\
&+\{U_x\}^T\{U_y\}^T\,\mathtt{M400YX}+\{U_y\}^T\{U_y\}^T\,\mathtt{M400YY} \\
&+\{U_y\}^T\{U_x\}^T\,\mathtt{M40X0Y}-\{U_y\}^T\{U_x\}^T\,\mathtt{M40Y0X}\Big]\{U_x\} \\
=&-\{U_j\}^T\{U_k\}^T\,\mathtt{M400KJ}\{U_x\}-\{U_y\}^T\{U_x\}^T(\,\mathtt{M40X0Y}-\,\mathtt{M40Y0X})\{U_x\} \\
\approx&-(\{U_j\}\{U_k\})^T\,\mathtt{M30KJ}\{U_x\}+\bar{U}_x\{U_y\}^T(\,\mathtt{M3X0Y}-\,\mathtt{M3Y0X})\{U_x\} \\
=&-(\{U_j\}\{U_k\})^T\,\mathtt{M30KJ}\{U_i\}+\bar{U}_i\{U_j\}^T(\,\mathtt{M3I0J}-\,\mathtt{M3J0I})\{U_i\}\,.
\end{aligned}
\tag{6.7}
$$

The first term with matrix M30KJ is essentially identical to the TWS term for the linear

advection-diffusion equation (4.14), and the second term with matrix (M3I0J − M3J0I) results from the nonlinearity. In a simplifying assumption, only the first term was used for computations in this research.

6.2 Shallow water equations

One of the simplest nonlinear systems is the one-dimensional shallow-water equations [40], a vertically integrated form of the Navier-Stokes equations. Neglecting viscous terms, the system is

$$\begin{aligned} &\frac{\partial h}{\partial t} + \frac{\partial}{\partial x}(uh) = 0 \\ &\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2} + \frac{h}{\mathrm{Fr}^2}\right) = 0 . \end{aligned} \tag{6.8}$$

6.2.1 Finite element method for shallow-water equations

A (nonconservative) weak statement form for (6.8) follows similarly to Burgers equation, i.e.,

$$\begin{aligned} \mathcal{W}^h(\{H\}) = \mathtt{M200}\frac{\{H\}^{n+1} - \{H\}^n}{\Delta_t} \\ + (\{H\}^T \mathtt{M300X}\{U\})^{n+\theta} + (\{U\}^T \mathtt{M300X}\{H\})^{n+\theta} = \{0\} , \\ \mathcal{W}^h(\{U\}) = \mathtt{M200}\frac{\{U\}^{n+1} - \{U\}^n}{\Delta_t} \\ + (\{U\}^T \mathtt{M300X}\{U\})^{n+\theta} + \frac{1}{\mathrm{Fr}^2}\,\mathtt{M20X}\{H\}^{n+\theta} = \{0\} . \end{aligned} \tag{6.9}$$

The entries in the Jacobian matrix are

$$\begin{aligned}
\frac{\partial \mathcal{W}^h(\{H\})}{\partial \{H\}^{n+1}} &= \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U\}^T\,\texttt{M300X} + \{H\}^T\,\texttt{M3X00}\Big)\,, \\
\frac{\partial \mathcal{W}^h(\{H\})}{\partial \{U\}^{n+1}} &= \theta\,\{U\}^T\,\texttt{M3X00}\,, \\
\frac{\partial \mathcal{W}^h(\{U\})}{\partial \{H\}^{n+1}} &= \frac{\theta}{\mathrm{Fr}^2}\{H\}^T\,\texttt{M3X00}\,, \\
\frac{\partial \mathcal{W}^h(\{U\})}{\partial \{U\}^{n+1}} &= \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U\}^T\,\texttt{M300X} + \{U\}^T\,\texttt{M3X00}\Big)\,.
\end{aligned} \tag{6.10}$$

6.2.2 TWS method for shallow-water equations

The second time derivatives required for the TWS correction to (6.10) are

$$\begin{aligned}
\frac{\partial^2 h}{\partial t^2} &= -\frac{\partial}{\partial t}\frac{\partial}{\partial x}(uh) = -\frac{\partial}{\partial x}(u\frac{\partial h}{\partial t} + h\frac{\partial u}{\partial t}) \\
\frac{\partial^2 h}{\partial t^2} &= \frac{\partial}{\partial x}\Big[u\frac{\partial}{\partial x}(uh) + h\frac{\partial}{\partial x}(\frac{u^2}{2} + \frac{h}{\mathrm{Fr}^2})\Big] \\
\frac{\partial^2 h}{\partial t^2} &= \frac{\partial}{\partial x}\Big[(u^2 + \frac{h}{\mathrm{Fr}^2})\frac{\partial h}{\partial x} + 2uh\frac{\partial u}{\partial x}\Big]\,, \\
\frac{\partial^2 u}{\partial t^2} &= -\frac{\partial}{\partial t}\frac{\partial}{\partial x}(\frac{u^2}{2} + \frac{h}{\mathrm{Fr}^2}) = -\frac{\partial}{\partial x}(u\frac{\partial u}{\partial t} + \frac{1}{\mathrm{Fr}^2}\frac{\partial h}{\partial t}) \\
\frac{\partial^2 u}{\partial t^2} &= \frac{\partial}{\partial x}\Big[u\frac{\partial}{\partial x}(\frac{u^2}{2} + \frac{h}{\mathrm{Fr}^2}) + \frac{1}{\mathrm{Fr}^2}\frac{\partial}{\partial x}(uh)\Big] \\
\frac{\partial^2 u}{\partial t^2} &= \frac{\partial}{\partial x}\Big[(u^2 + \frac{h}{\mathrm{Fr}^2})\frac{\partial u}{\partial x} + \frac{2u}{\mathrm{Fr}^2}\frac{\partial h}{\partial x}\Big]\,.
\end{aligned} \tag{6.11}$$

The preceding may be more easily developed using the flux-vector form of (6.8),

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} = 0\,, \tag{6.12}$$

with $\mathbf{q} = (h, u)^T$ and $\mathbf{F} = (uh, \frac{u^2}{2} + \frac{h}{\mathrm{Fr}^2})^T$. $\mathbf{F}$ is assumed to be a function of $\mathbf{q}$ only such that

$$\begin{aligned}
\frac{\partial}{\partial x}[\mathbf{F}(\mathbf{q})] &= \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial x} \\
\frac{\partial}{\partial t}\left(\frac{\partial \mathbf{q}}{\partial t}\right) &= -\frac{\partial}{\partial t}\left(\frac{\partial \mathbf{F}}{\partial x}\right) = -\frac{\partial}{\partial x}\left(\frac{\partial \mathbf{F}}{\partial t}\right) \\
&= -\frac{\partial}{\partial x}\left(\frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial t}\right) = \frac{\partial}{\partial x}\left(\frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{q}}{\partial x}\right)
\end{aligned} \tag{6.13}$$

For the equation set (6.13), the flux-vector Jacobian matrices are

$$\begin{aligned}
\frac{\partial \mathbf{F}}{\partial \mathbf{q}} &= \begin{bmatrix} u & h \\ \frac{1}{\mathrm{Fr}^2} & u \end{bmatrix} \\
\frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} &= \begin{bmatrix} u^2 + \frac{h}{\mathrm{Fr}^2} & 2uh \\ 2\frac{u}{\mathrm{Fr}^2} & u^2 + \frac{h}{\mathrm{Fr}^2} \end{bmatrix},
\end{aligned} \tag{6.14}$$

which is a condensed form of (6.11). A discrete finite element approximation of the TWS terms (6.11) is

$$\begin{aligned}
\int_\Omega \{\chi\} \frac{\partial^2 \mathbf{q}}{\partial t^2} d\Omega^h &\approx \int_{\Omega^h} \{\chi\} \frac{\partial}{\partial x}\left[\{\psi\}^T \left(\frac{\partial \mathbf{F}}{\partial \{Q\}} \frac{\partial \mathbf{F}}{\partial \{Q\}}\right) \frac{\partial \{\psi\}^T}{\partial x} \{Q\}\right] d\Omega^h \\
&= -\int_{\Omega^h} \frac{\partial \{\chi\}}{\partial x}\left[\{\psi\}^T \left(\frac{\partial \mathbf{F}}{\partial \{Q\}} \frac{\partial \mathbf{F}}{\partial \{Q\}}\right) \frac{\partial \{\psi\}^T}{\partial x} \{Q\}\right] d\Omega^h \\
&= -\left\{\frac{\partial \mathbf{F}}{\partial \{Q\}} \frac{\partial \mathbf{F}}{\partial \{Q\}}\right\}^T \texttt{M30XX} \{Q\}\,.
\end{aligned} \tag{6.15}$$

In such hyperbolic conservation law systems resembling the Euler equations, the conservation form is often preferred [67], as it is found to produce more accurate approximations of flows with shocks by better enforcing mass conservation. In the conservation form, the equation set is rearranged so that, in the mass conservation equation (here the equation for h), the object of the space derivative (uh) is made a primary variable (m), yielding

$$\begin{aligned}
\frac{\partial h}{\partial t} + \frac{\partial}{\partial x}(m) &= 0\,, \\
\frac{\partial m}{\partial t} + \frac{\partial}{\partial x}\left(\frac{m^2}{h} + \frac{h^2}{2\mathrm{Fr}^2}\right) &= 0\,,
\end{aligned} \tag{6.16}$$

and the first equation of the conservation law set shows that m will have a small (and thus more accurately approximated) gradient as the solution approaches steady state. The

flux vector form is $\mathbf{q} = (h,m)^T$, $\mathbf{F} = (m, \frac{m^2}{h} + \frac{h^2}{2\mathrm{Fr}^2})^T$, and the replacement for (6.14) is

$$\begin{aligned} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} &= \begin{bmatrix} 0 & 1 \\ \frac{h}{\mathrm{Fr}^2} - \frac{m^2}{h^2} & \frac{2m}{h} \end{bmatrix}, \\ \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} &= \begin{bmatrix} \frac{h}{\mathrm{Fr}^2} - \frac{m^2}{h^2} & \frac{2m}{h} \\ \frac{2m}{h}(\frac{h}{\mathrm{Fr}^2} - \frac{m^2}{h^2}) & \frac{h}{\mathrm{Fr}^2} + \frac{3m^2}{h^2} \end{bmatrix}. \end{aligned} \tag{6.17}$$

Optionally defining $u = \frac{m}{h}$ for calculation yields

$$\frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} = \begin{bmatrix} \frac{h}{\mathrm{Fr}^2} - u^2 & 2u \\ 2u(\frac{h}{\mathrm{Fr}^2} - u^2) & \frac{h}{\mathrm{Fr}^2} + 3u^2 \end{bmatrix}. \tag{6.18}$$

Two potential problems are apparent with this TWS formulation. The term $u = \frac{m}{h}$ may become very large at small h, and three of the terms in (6.18) may become negative, hence destabilizing. One approximate TWS formulation is to neglect the off-diagonal and possibly negative terms of (6.18), yielding a diagonal and uniform positive TWS contribution as

$$\frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \approx \begin{bmatrix} \frac{h}{\mathrm{Fr}^2} & 0 \\ 0 & \frac{h}{\mathrm{Fr}^2} + 3u^2 \end{bmatrix}. \tag{6.19}$$

Another approximate TWS term for the non-conservation form (6.14) would be

$$\frac{\partial \mathbf{F}}{\partial \mathbf{q}} \frac{\partial \mathbf{F}}{\partial \mathbf{q}} \approx \begin{bmatrix} u^2 + \frac{h}{\mathrm{Fr}^2} & 0 \\ 0 & u^2 + \frac{h}{\mathrm{Fr}^2} \end{bmatrix}. \tag{6.20}$$

6.3 Navier-Stokes equations and pressure approximation

The non-dimensional incompressible laminar flow Navier-Stokes equations in two dimensions [85] are

$$\begin{aligned} &\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} = 0, \\ &\frac{\partial u_x}{\partial t} + u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y} + \frac{\partial p}{\partial x} - \frac{1}{\mathrm{Re}}(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2}) = 0, \\ &\frac{\partial u_y}{\partial t} + u_x \frac{\partial u_y}{\partial y} + u_y \frac{\partial u_y}{\partial x} + \frac{\partial p}{\partial y} - \frac{1}{\mathrm{Re}}(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2}) + \frac{\mathrm{Gr}}{\mathrm{Re}^2}\Theta = 0, \\ &\frac{\partial \Theta}{\partial t} + u_x \frac{\partial \Theta}{\partial x} + u_y \frac{\partial \Theta}{\partial y} - \frac{1}{\mathrm{Pe}}(\frac{\partial^2 \Theta}{\partial x^2} + \frac{\partial^2 \Theta}{\partial y^2}) = 0. \end{aligned} \tag{6.21}$$

The continuity equation presents a computational difficulty because it constrains the pressure field to that which will satisfy continuity, but does not provide a explicit equation for calculating that pressure field. Here, a projection or pressure correction method [20], [43] is selected to approximately enforce the continuity constraint.

If a divergence-free velocity field can be found, calculation of the pressure field is straightforward. Taking the divergence of the momentum equations,

$$\begin{aligned}&\frac{\partial}{\partial x}\Big(\frac{\partial u_x}{\partial t}+u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}+\frac{\partial p}{\partial x}-\frac{1}{\mathrm{Re}}(\frac{\partial^2 u_x}{\partial x^2}+\frac{\partial^2 u_x}{\partial y^2})\Big)\\&+\frac{\partial}{\partial y}\Big(\frac{\partial u_y}{\partial t}+u_y\frac{\partial u_y}{\partial y}+u_x\frac{\partial u_y}{\partial x}+\frac{\partial p}{\partial y}-\frac{1}{\mathrm{Re}}(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2})+\frac{\mathrm{Gr}}{\mathrm{Re}^2}\Theta\Big)=0\,,\end{aligned}\tag{6.22}$$

and assuming that mixed derivatives are interchangeable,

$$\begin{aligned}&\frac{\partial}{\partial t}\Big(\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}\Big)-\frac{1}{\mathrm{Re}}\nabla\cdot\nabla\Big(\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}\Big)+\frac{\mathrm{Gr}}{\mathrm{Re}^2}\frac{\partial\Theta}{\partial y}\\&+\frac{\partial}{\partial x}\Big(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\Big)+\frac{\partial}{\partial y}\Big(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\Big)+\nabla\cdot\nabla p=0\,.\end{aligned}\tag{6.23}$$

If continuity is satisfied in (6.23), the first two terms are zero, and the resulting "pressure Poisson equation" is

$$\nabla\cdot\nabla p+\frac{\partial}{\partial x}\Big(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\Big)+\frac{\partial}{\partial y}\Big(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\Big)+\frac{\mathrm{Gr}}{\mathrm{Re}^2}\frac{\partial\Theta}{\partial y}=0\,.\tag{6.24}$$

The Neumann boundary condition is obtained from the dot product of the momentum equation with the normal direction [43]. At Dirichlet boundaries for velocity (walls and inlets in this study) only the viscous and pressure terms remain, hence

$$\vec{\mathbf{n}}\cdot\nabla(p^{n+1})=\frac{1}{\mathrm{Re}}\nabla\cdot\nabla(\vec{u}\cdot\hat{n})\,.\tag{6.25}$$

The velocity second derivatives on the right-hand side of (6.25) are approximated here by finite differences. In this research, outlet boundaries were assigned homogeneous Neumann conditions for velocities and a Dirichlet condition for pressure.

A discrete form of the divergence of momentum (6.23) may be used to generate a procedure for determining a pressure-like field which approximately enforces continuity. Assuming a known divergence-free initial condition at time level n, the momentum equa-

tions are used to generate velocity solutions at time level $n+1$ using the pressure from time level n. This velocity field, denoted here as $(\hat{u}_x, \hat{u}_y)^{n+1}$ and in general not satisfying continuity, is evaluated by substituting a semi-discrete approximation for the time term in (6.23). Assuming for brevity that $\theta = 1$,

$$\begin{aligned}
&\frac{1}{\Delta_t}\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1}-\frac{1}{\Delta_t}\left(\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}\right)^{n}+\frac{\partial}{\partial x}\left(\hat{u}_x\frac{\partial \hat{u}_x}{\partial x}+u_y\frac{\partial \hat{u}_x}{\partial y}\right)^{n+1}\\
&+\frac{\partial}{\partial y}\left(\hat{u}_x\frac{\partial \hat{u}_y}{\partial x}+\hat{u}_y\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1}+\nabla\cdot\nabla\hat{p}^{n+1} \qquad (6.26)\\
&+\frac{\mathrm{Gr}}{\mathrm{Re}^2}\frac{\partial \hat{\Theta}}{\partial y}^{n+1}-\frac{1}{\mathrm{Re}}\nabla\cdot\nabla\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1}=0\,.
\end{aligned}$$

Assuming the existence of an unknown velocity field (u_x^{n+1}, u_y^{n+1}) which does satisfy continuity, formulating a similar semidiscrete approximation, and subtracting (6.26) from it yields

$$\begin{aligned}
&\frac{1}{\Delta_t}\left(\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}\right)^{n+1}-\frac{1}{\Delta_t}\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1}\\
&-\frac{1}{\mathrm{Re}}\nabla\cdot\nabla\left(\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}\right)^{n+1}+\frac{1}{\mathrm{Re}}\nabla\cdot\nabla\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1}\\
&+\frac{\partial}{\partial x}\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)^{n+1}+\frac{\partial}{\partial y}\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)^{n+1} \qquad (6.27)\\
&-\frac{\partial}{\partial x}\left(\hat{u}_x\frac{\partial \hat{u}_x}{\partial x}+\hat{u}_y\frac{\partial \hat{u}_x}{\partial y}\right)^{n+1}-\frac{\partial}{\partial y}\left(\hat{u}_x\frac{\partial \hat{u}_y}{\partial x}+\hat{u}_y\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1}\\
&+\nabla\cdot\nabla p^{n+1}-\nabla\cdot\nabla\hat{p}^{n+1}+\frac{\mathrm{Gr}}{\mathrm{Re}^2}\left(\frac{\partial \Theta}{\partial y}^{n+1}-\frac{\partial \hat{\Theta}}{\partial y}^{n+1}\right)=0\,.
\end{aligned}$$

Recalling that the divergence of the unknown velocity field (u_x^{n+1}, u_y^{n+1}) is assumed negligible at time level $n+1$, and that the intermediate pressure field $\hat{p}^{n+1} \equiv p^n$,

(6.27) simplifies to

$$\begin{aligned}
&\nabla\cdot\nabla(p^{n+1}-p^n) = \frac{1}{\Delta_t}\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1} - \frac{1}{\mathrm{Re}}\nabla\cdot\nabla\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1} \\
&-\frac{\partial}{\partial x}\left(u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\right)^{n+1} + \frac{\partial}{\partial x}\left(\hat{u}_x\frac{\partial \hat{u}_x}{\partial x}+\hat{u}_y\frac{\partial \hat{u}_x}{\partial y}\right)^{n+1} \\
&-\frac{\partial}{\partial y}\left(u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\right)^{n+1} + \frac{\partial}{\partial y}\left(\hat{u}_x\frac{\partial \hat{u}_y}{\partial x}+\hat{u}_y\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1} \\
&-\frac{\mathrm{Gr}}{\mathrm{Re}^2}\left(\frac{\partial \Theta^{n+1}}{\partial y} - \frac{\partial \hat{\Theta}^{n+1}}{\partial y}\right) = 0\,.
\end{aligned} \tag{6.28}$$

Equation (6.28) as shown is not solvable, since the unknowns $(u_x^{n+1}, u_y^{n+1}, \Theta^{n+1})$ appear on the right-hand side. In practice, arguments are developed to retain only the first one or two terms on the right-hand side [86]; in particular, a Helmholtz decomposition of the velocity may be used to rigorously eliminate some of the other terms, but requires some simplifying assumptions on the form of the momentum equations. Here an order-of-magnitude approach will be followed to eliminate most of the terms. Values for a nondimensionalization are assumed such that $O(u) = O(\hat{u}) = O(t) = 1$, $O(\frac{1}{\mathrm{Re}}) = 0.01$, and $O(\frac{\mathrm{Gr}}{\mathrm{Re}^2}) = 1$. Typical computational constants are $O(\Delta_t) = O(\Delta_x) = 0.01$. If the $\hat{u}$ approximation is reasonable such that $O(u-\hat{u}) = 0.1$, then the first term on the right-hand side of (6.28) with coefficient $\frac{1}{\Delta_t}$, will be $O(10^3)$. The second term with coefficient $\frac{1}{\mathrm{Re}}$ will be $O(10^2)$. The last term with coefficient $\frac{\mathrm{Gr}}{\mathrm{Re}^2}$ will be $O(10)$, and the remaining velocity derivatives terms will be $O(1)$.

Therefore, retaining only the first Δ_t term is a reasonable approximation. The order-of-magnitude analysis also illustrates a disadvantage of the formulation, i.e. that Δ_t is limited in practice to a small value. The resultant "pressure correction equation" is

$$\nabla\cdot\nabla(\tilde{p}^{n+1}-p^n) - \frac{1}{\Delta_t}\left(\frac{\partial \hat{u}_x}{\partial x}+\frac{\partial \hat{u}_y}{\partial y}\right)^{n+1} \approx 0\,, \tag{6.29}$$

where the approximation to p^{n+1} is designated $\tilde{p}^{n+1}$. If (6.29) is developed using the Helmholtz decomposition, a companion velocity correction is available,

$$(\tilde{u}_i^{n+1} - \hat{u}_i^{n+1}) \approx -\Delta_t \frac{\partial}{\partial x_i}(\tilde{p}^{n+1} - p^n)\,. \tag{6.30}$$

The velocity correction was not used for computation in this study, but is convenient in determining boundary conditions for (6.29). At nodes which have Dirichlet boundary conditions for velocity, typically inlets and no-slip walls, the dot product of (6.30) in the normal direction yields a homogeneous Neumann boundary condition for pressure correction. At nodes which have Dirichlet boundary conditions for pressure, typically an inlet or outlet boundary, pressure is assumed correct, hence a zero Dirichlet boundary condition for pressure correction is appropriate.

6.3.1 Finite element method for Navier-Stokes equations

A finite element weak statement algorithm for the momentum and energy equations is developed as an extension of the developments for the advection-diffusion equation and the Burgers equation, i.e.,

$$
\begin{aligned}
\mathcal{W}^h(\{U_x\}) &= \mathtt{M200}\frac{\{U_x\}^{n+1}-\{U_x\}^n}{\Delta_t} + \left(\{U_k\}^T\,\mathtt{M300K}\,\{U_x\}\right)^{n+\theta} \\
&\quad + \mathtt{M20X}\{P\} + \frac{1}{\mathrm{Re}}\,\mathtt{M2KK}\,\{U_x\}^{n+\theta} = \{0\}\,, \\
\mathcal{W}^h(\{U_y\}) &= \mathtt{M200}\frac{\{U_y\}^{n+1}-\{U_y\}^n}{\Delta_t} + \left(\{U_k\}^T\,\mathtt{M300K}\,\{U_y\}\right)^{n+\theta} \\
&\quad + \mathtt{M20Y}\{P\} + \frac{1}{\mathrm{Re}}\,\mathtt{M2KK}\,\{U_y\}^{n+\theta} + \frac{\mathrm{Gr}}{\mathrm{Re}^2}\,\mathtt{M200}\{\Theta\} = \{0\}\,, \\
\mathcal{W}^h(\{\Theta\}) &= \mathtt{M200}\frac{\{\Theta\}^{n+1}-\{\Theta\}^n}{\Delta_t} + \left(\{U_k\}^T\,\mathtt{M300K}\,\{\Theta\}\right)^{n+\theta} \\
&\quad + \frac{1}{\mathrm{Pe}}\,\mathtt{M2KK}\,\{\Theta\}^{n+\theta} = \{0\}\,.
\end{aligned}
\tag{6.31}
$$

The entries of the Jacobian matrix are, with all terms evaluated at time level $n+1$,

$$
\begin{aligned}
\frac{\partial \mathcal{W}^h(\{U_x\}^{n+1})}{\partial \{U_x\}^{n+1}} &= \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U_k\}^T\,\texttt{M300K} + \{U_x\}^T\,\texttt{M3X00} + \frac{1}{\mathrm{Re}}\,\texttt{M2KK}\Big)\,,\\
\frac{\partial \mathcal{W}^h(\{U_x\}^{n+1})}{\partial \{U_y\}^{n+1}} &= \theta\{U_y\}^T\,\texttt{M3Y00}\,,\\
\frac{\partial \mathcal{W}^h(\{U_x\}^{n+1})}{\partial \{\Theta\}^{n+1}} &= 0\,,\\
\frac{\partial \mathcal{W}^h(\{U_y\}^{n+1})}{\partial \{U_x\}^{n+1}} &= \theta\{U_x\}^T\,\texttt{M3X00}\,,\\
\frac{\partial \mathcal{W}^h(\{U_y\}^{n+1})}{\partial \{U_y\}^{n+1}} &= \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U_k\}^T\,\texttt{M300K} + \{U_y\}^T\,\texttt{M3Y00} + \frac{1}{\mathrm{Re}}\,\texttt{M2KK}\Big)\,,\\
\frac{\partial \mathcal{W}^h(\{U_y\}^{n+1})}{\partial \{\Theta\}^{n+1}} &= \frac{\mathrm{Gr}}{\mathrm{Re}^2}\theta\,\texttt{M200}^{n+1}\,,\\
\frac{\partial \mathcal{W}^h(\{\Theta\}^{n+1})}{\partial \{U_x\}^{n+1}} &= \theta\{U_x\}^T\,\texttt{M3Y00}\,,\\
\frac{\partial \mathcal{W}^h(\{\Theta\}^{n+1})}{\partial \{U_y\}^{n+1}} &= \theta\{U_y\}^T\,\texttt{M3Y00}\,,\\
\frac{\partial \mathcal{W}^h(\{\Theta\}^{n+1})}{\partial \{\Theta\}^{n+1}} &= \frac{\texttt{M200}}{\Delta_t} + \theta\Big(\{U_k\}^T\,\texttt{M300K} + \{\Theta\}^T\,\texttt{M3X00} + \frac{1}{\mathrm{Pe}}\,\texttt{M2KK}\Big)\,.
\end{aligned}
\tag{6.32}
$$

Because the complete Jacobian produces a very large bandwidth, the code developed for this research solves the equations sequentially using only the diagonal Jacobian entries, i.e. $\frac{\partial \mathcal{W}^h(\{U_x\}^{n+1})}{\partial \{U_x\}^{n+1}}$, $\frac{\partial \mathcal{W}^h(\{U_y\}^{n+1})}{\partial \{U_y\}^{n+1}}$, $\frac{\partial \mathcal{W}^h(\{\Theta\}^{n+1})}{\partial \{\Theta\}^{n+1}}$ from (6.32).

The finite element algorithm for the Poisson equation for pressure correction (6.29) has the discrete approximation

$$
\texttt{M2KK}\,(\tilde{P}^{n+1} - \hat{P}) - \frac{1}{\Delta_t}\,\texttt{M20K}\{U_k\} = \{0\}\,,
\tag{6.33}
$$

and the pressure Poisson equation (6.24) has the discrete approximation

$$
\begin{aligned}
&\texttt{M2KK}\,P^{n+1} + \frac{\mathrm{Gr}}{\mathrm{Re}^2}\,\texttt{M20Y}\{\Theta\}^{n+1} - \Big(\{U_x\}^T\,\texttt{M3X0X}\{U_x\} + \{U_y\}^T\,\texttt{M3X0Y}\{U_x\}\\
&+ \{U_x\}^T\,\texttt{M3Y0X}\{U_y\} + \{U_y\}^T\,\texttt{M3Y0Y}\{U_y\}\Big)^{n+1} = \{0\}\,.
\end{aligned}
\tag{6.34}
$$

6.3.2 TWS method for the Navier-Stokes equations

Previous developments of the TWS correction terms for the incompressible Navier-Stokes equations [28] have neglected pressure terms, since explicit time derivatives of pressure are not available. The TWS correction terms for velocity and temperature were assumed in this research to be identical to those for the advection-diffusion equation, neglecting the additional terms shown for the Burgers equation in the continuum (6.5) and discrete (6.7) forms, and also neglecting TWS corrections resulting from source and viscous terms. Incorporating those TWS corrections into (6.31) and using directional γ and β coefficients yields

$$\begin{aligned}
\mathcal{W}^h(\{U_i\}) =& \left(\mathtt{M200} + \gamma_{kj}\bar{U}_k\bar{U}_j\Delta_t\,\mathtt{M2KJ}\right)\left(\{U_i\}^{n+1} - \{U_i\}^n\right) \\
&+\Delta_t\left[\left(\{U_k\}^T\,\mathtt{M300K}\,\{U_i\}\right)^{n+\theta} + \delta_{ik}\,\mathtt{M20K}\{P\} + \frac{1}{\mathrm{Re}}\,\mathtt{M2KK}\,\{U_i\}\right]^{n+\theta} \\
&+\frac{\mathrm{Gr}}{\mathrm{Re}^2}\,\mathtt{M200}\{\Theta\}^{n+\theta} + \beta_{kj}\bar{U}_k\bar{U}_j\Delta_t^*\,\mathtt{M2KJ}\,\{U_i\}^{n+\theta}\Big] = \{0\}\,, \\
\mathcal{W}^h(\{\Theta\}) =& \left(\mathtt{M200} + \gamma_{kj}\bar{U}_k\bar{U}_j\Delta_t\,\mathtt{M2KJ}\right)\left(\{\Theta\}^{n+1} - \{\Theta\}^n\right) \\
&+\Delta_t\left[\left(\{U_k\}^T\,\mathtt{M300K}\,\{\Theta\}\right)^{n+\theta} + \frac{1}{\mathrm{Pe}}\,\mathtt{M2KK}\,\{\Theta\}\right]^{n+\theta} \\
&+\beta_{kj}\bar{U}_k\bar{U}_j\Delta_t^*\,\mathtt{M2KJ}\,\{\Theta\}^{n+\theta}\Big] = \{0\}\,.
\end{aligned} \tag{6.35}$$

The Jacobian terms in (6.32) are then modified to add the TWS terms, assuming derivatives with respect to the leading coefficients $\bar{U}_k\bar{U}_j$ are negligible. Then (6.33-6.35) with boundary conditions define a discrete approximation for the Navier-Stokes equations.

A computer program was written in Fortran to implement the developed range of approximations. A Gaussian elimination direct solver was used when at least one Dirichlet boundary condition was available. The discrete finite element form of the Poisson equations (6.33) and (6.34) with only Neumann conditions produces a positive semi-definite solution matrix which is unsuitable for Gaussian elimination, so a bi-conjugate-gradient solver [62] was coded and used in those cases.

A pressure correction solution was implemented after the first of four to five iterations at each nondimensional timestep, of magnitude 0.1 to 0.005, depending on the problem and not changed during the simulation. The pressure correction solution was ac-

cumulated until a pressure Poisson solution was implemented after each 5 to 25 timesteps. At that point the pressure Poisson solution replaced the previous pressure, and the cumulative pressure correction was zeroed. True steady-state solutions in parameters such as energy norms were difficult to achieve at higher Reynolds numbers, so steady-state was considered to have been reached when centerline and wall profiles of velocities remained constant to five significant figures.

Chapter 7

Results and discussion

A list of the tested algorithms is shown in Table 7.1. The selected test problems

Table 7.1: Summary of algorithms tested

Symbol	finite element algorithms
GWS	1D/2D/3D transient/steady Galerkin weak statement
TWS-γ	1D/2D/3D transient TWS time-accurate
TWS-β	1D/2D/3D transient/steady TWS artificial diffusion
SUPG	2D transient/steady artificial diffusion SUPG
SGM	2D transient/steady artificial diffusion SGM
MPG-1	2D transient/steady artificial diffusion first monotone method
MPG-2	2D transient/steady artificial diffusion second monotone method
EPG	2D transient/steady artificial diffusion exponential PG
BPG	2D transient/steady artificial diffusion bubble function PG

include verification problems for which known analytical solutions exist, and benchmark problems for which numerical solutions which have been generated on sufficiently fine meshes and reported in the literature.

7.1 Verification simulations for the advection-diffusion equations

Many analytical solutions are available for the advection-diffusion equation. With zero diffusion, the transient advection equation (1.7) exactly propagates any initial condition in the direction of the velocity. With diffusion included, several steady-state exact solutions are available with select source or boundary conditions necessary to generate a nonzero solution.

7.1.1 Translating wave one-dimensional verification simulation

A square wave with considerable high-frequency content has often been used to assess phase accuracy [60],[48] for the one-dimensional advection equation. A snapshot of the analytical solution is graphed in Figure 7.1 at times $t = 0.2$ and $t = 0.8$. The one-

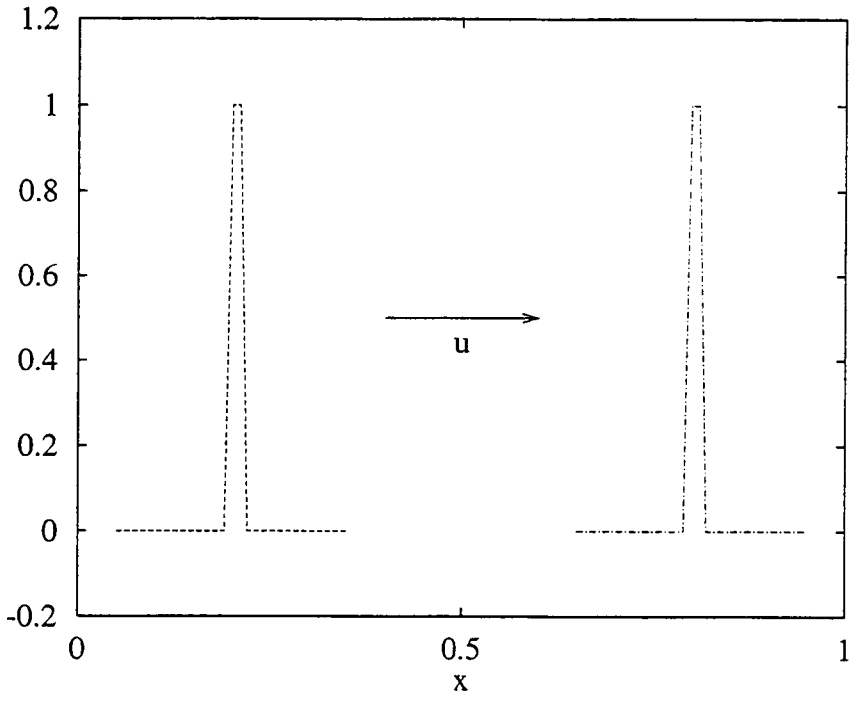

Figure 7.1: Translating square wave initial and analytical solution

dimensional advection-diffusion equation with constant coefficients u and Pe is

$$\frac{\partial q}{\partial t} + u\frac{\partial q}{\partial x} - \frac{1}{\mathrm{Pe}}\frac{\partial^2 q}{\partial x^2} = 0\,, \tag{7.1}$$

with the analytical solution from Carey and Oden [15],

$$q(x,t) = \frac{1}{2}\exp(x\mathrm{Pe})\left[1-\mathrm{erf}\left(\left(\frac{\mathrm{Pe}}{t}\right)^{\frac{1}{2}}\frac{x+t}{2}\right)\right] + \left[1-\mathrm{erf}\left(\left(\frac{\mathrm{Pe}}{t}\right)^{\frac{1}{2}}\frac{x-t}{2}\right)\right]\,. \tag{7.2}$$

For simulation purposes, initial and boundary conditions are generated from the analytical solution. A snapshot of the analytical solution is graphed in Figure 7.2 at times $t = 0.2$ and $t = 0.8$ for $\mathrm{Pe} = 800$, the value used The steepness of the front increases with Peclet number and decreases with time.

A non-diffusive translating square wave and a diffusive translating wave with solution (7.2) were simulated using a variety of finite element and finite difference algorithms at several Courant numbers. Since time accuracy is the primary requirement for this class of problems, the trapezoidal rule ($\theta = \frac{1}{2}$) was used with each algorithm. As a rough quantitative comparative measure of accuracy, the simulation value at the center of the diffusive

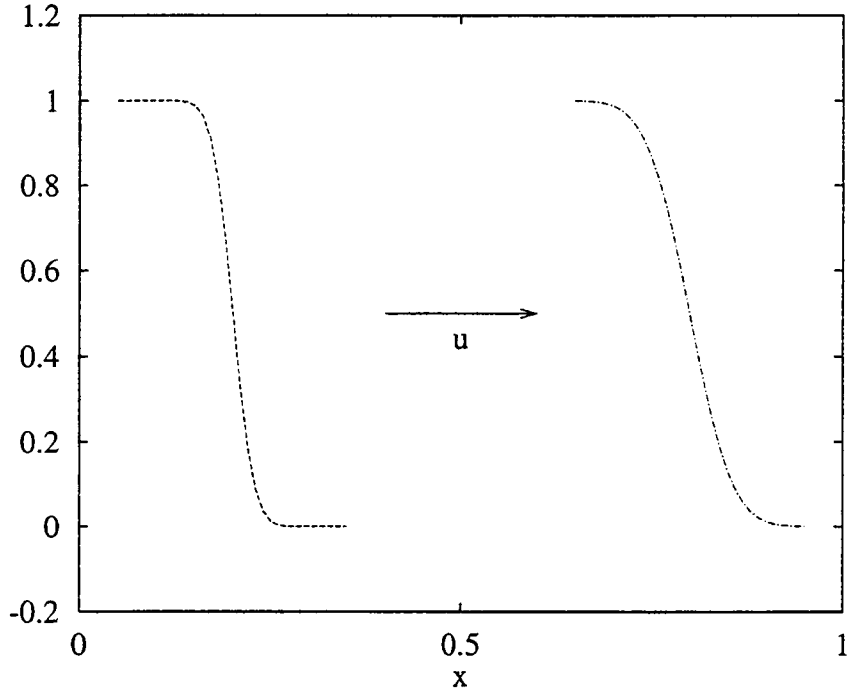

Figure 7.2: Translating diffusive wave initial and analytical solution

wave with analytical value $q = 0.5$ at $x = 0.8$ is shown in Table 7.2 for each algorithm and each Courant number.

Table 7.2: Summary of one-dimensional diffusive wave predicted center value

Algorithm	$C = \frac{1}{4}$	$C = \frac{1}{2}$	$C = \frac{3}{4}$	$C = 1$
Optimized Taylor linear FEM	0.50001	0.50098	0.49067	0.50302
Taylor quadratic FEM	0.50685	0.50881	0.51214	0.45354
Taylor linear FEM	0.48565	0.48756	0.49147	0.50297
Galerkin cubic FEM	0.50034	0.45431	0.41680	0.39432
Galerkin quadratic FEM	0.49086	0.45012	0.41341	0.39065
Galerkin linear FEM	0.47282	0.44303	0.41225	0.38968
QUICK-5 FVM	0.38116	0.39588	0.39379	0.39588
QUICK-3 FVM	0.38402	0.39404	0.40108	0.40395
Crank-Nicolson FDM	0.37204	0.36923	0.36524	0.36070

Table 7.2 shows that each Crank-Nicolson finite difference solution missed the analytical solution by about -0.13, each QUICK-3 finite volume solution missed the value by about -0.12, and each QUICK-5 finite volume solution missed the value by about -0.11, with each FDM/FVM accuracy relatively independent of Courant number. The

Galerkin finite element solutions were more accurate than the FVM at small Courant number, error decreasing with FEM basis degree from -0.03 to $+0.003$. The Galerkin solutions each deteriorated with increasing Courant number to an error of about -0.11 at $C = 1$. The 12 Taylor finite element solution errors were within ± 0.015 with the exception of one point (TWS quadratic at $C = 1$) at -0.05.

The complete simulation data for $C = \frac{1}{4}$ are graphed in Figure 7.3 with exact solutions shown as dotted lines and simulated solutions as symbols. The simulation data for $C = \frac{1}{2}$ are graphed in Figure 7.4, the simulation data for $C = \frac{3}{4}$ are graphed in Figure 7.5, and the simulation data for $C = 1$ are graphed in Figure 7.6. The relative performance trends shown in Table 7.2 carry over to both diffusive and non-diffusive wave solutions, although the diffusive wave solutions are more accurate in each case. An error measure for the non-diffusive traveling square wave is the lag of the simulated peak value from the exact solution. Each Crank-Nicolson square wave solution peak lags 3 to 5 nodes behind the analytical peak, while each QUICK solution peak lags about 3 nodes. The linear basis Galerkin peaks lag about one node at small Courant number to two nodes at $C = 1$. The quadratic and cubic basis Galerkin peaks lag zero nodes at small Courant number to two nodes at $C = 1$. The Taylor linear and quadratic basis peaks are within one node or less at each Courant number, and the optimized Taylor linear basis peak lags zero nodes at each Courant number. In summary, only the optimized Taylor linear basis algorithm is satisfactory for both test problems over the tested range of Courant number.

7.1.2 Rotating cone transient two-dimensional verification simulation

A useful and informative verification problem for inviscid transient advection in two dimensions is a "rotating cone" half-wave cosine hill initial condition convected in the $x - y$ plane by a solid body rotation velocity field. The trapezoidal rule is used for time accuracy, and a time-split matrix factorization algorithm [8] was used to factor both Jacobian and residual terms, producing a one-dimensional alternating direction solution process.

Comparative results are graphed in Figure 7.7 for $C = \frac{1}{2}$, with Figure 7.7a showing the initial condition and the (identical) analytical solution after one rotation. The rank of solution quality by algorithm follows the one-dimensional presentation, as determined by the comparative phase accuracy of the algorithms in Figures 5.1-5.3. The Crank-Nicolson

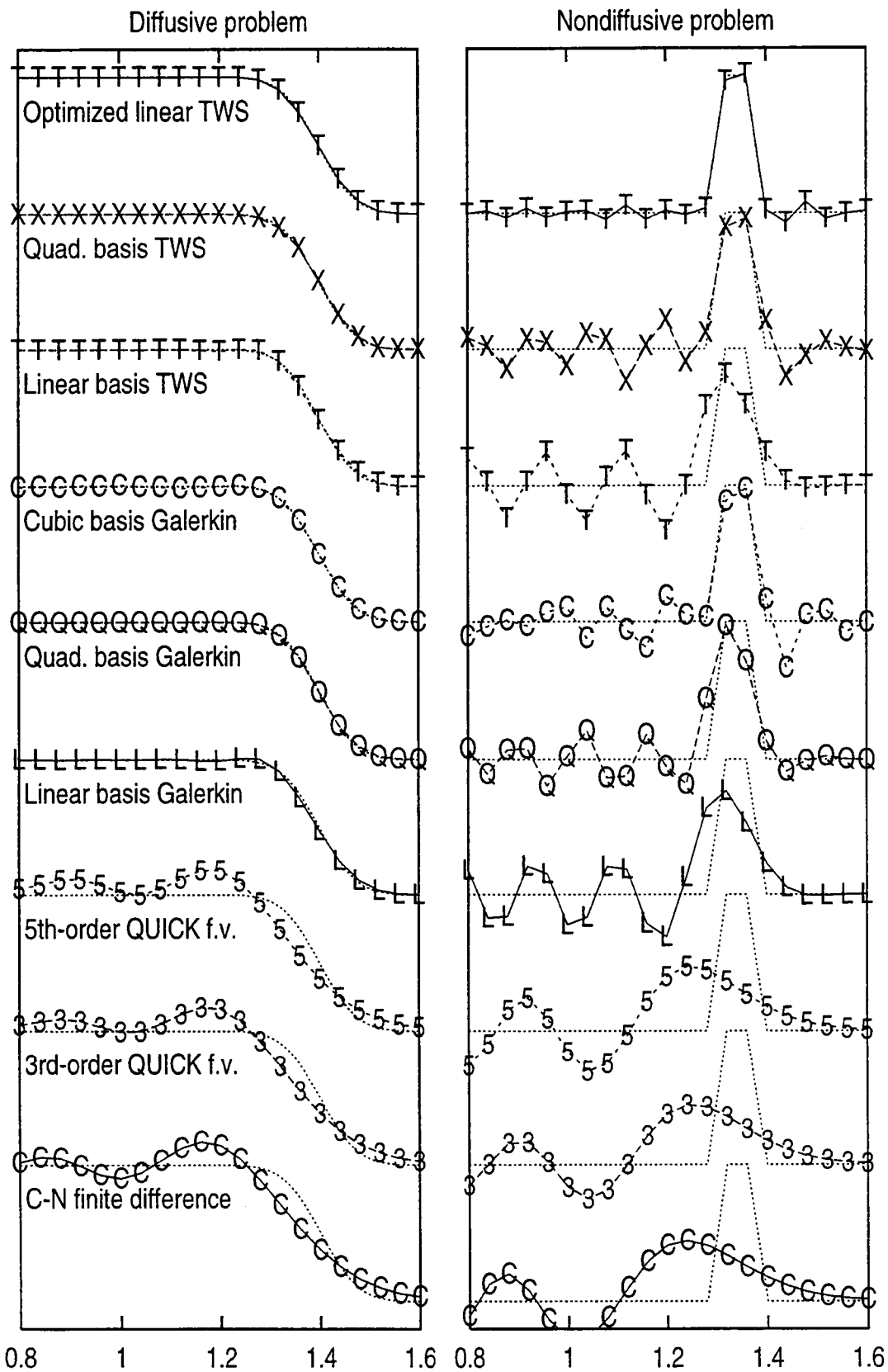

Figure 7.3: One-dimensional simulations at $C = \frac{1}{4}$

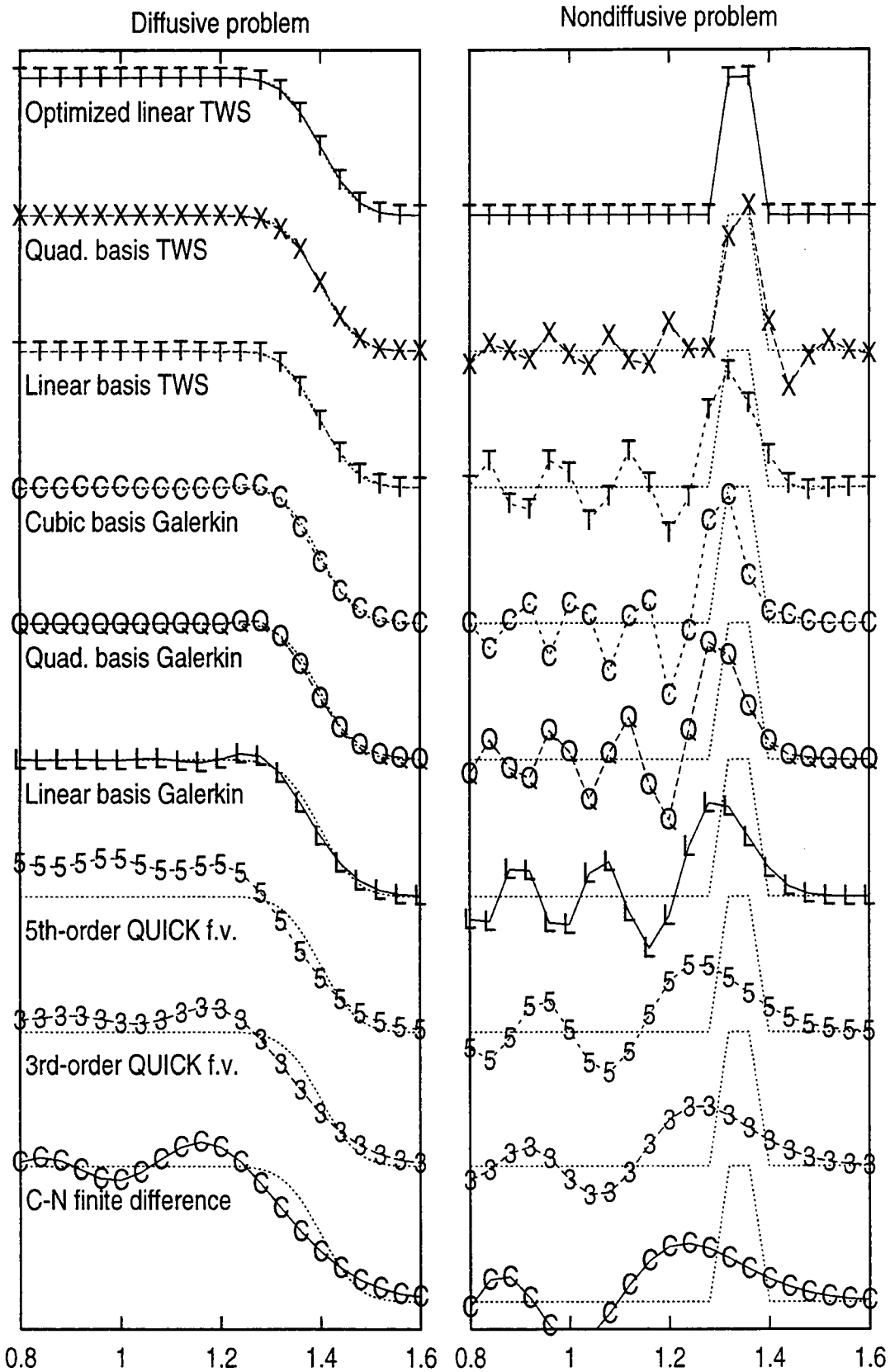

Figure 7.4: One-dimensional simulations at $C = \frac{1}{2}$

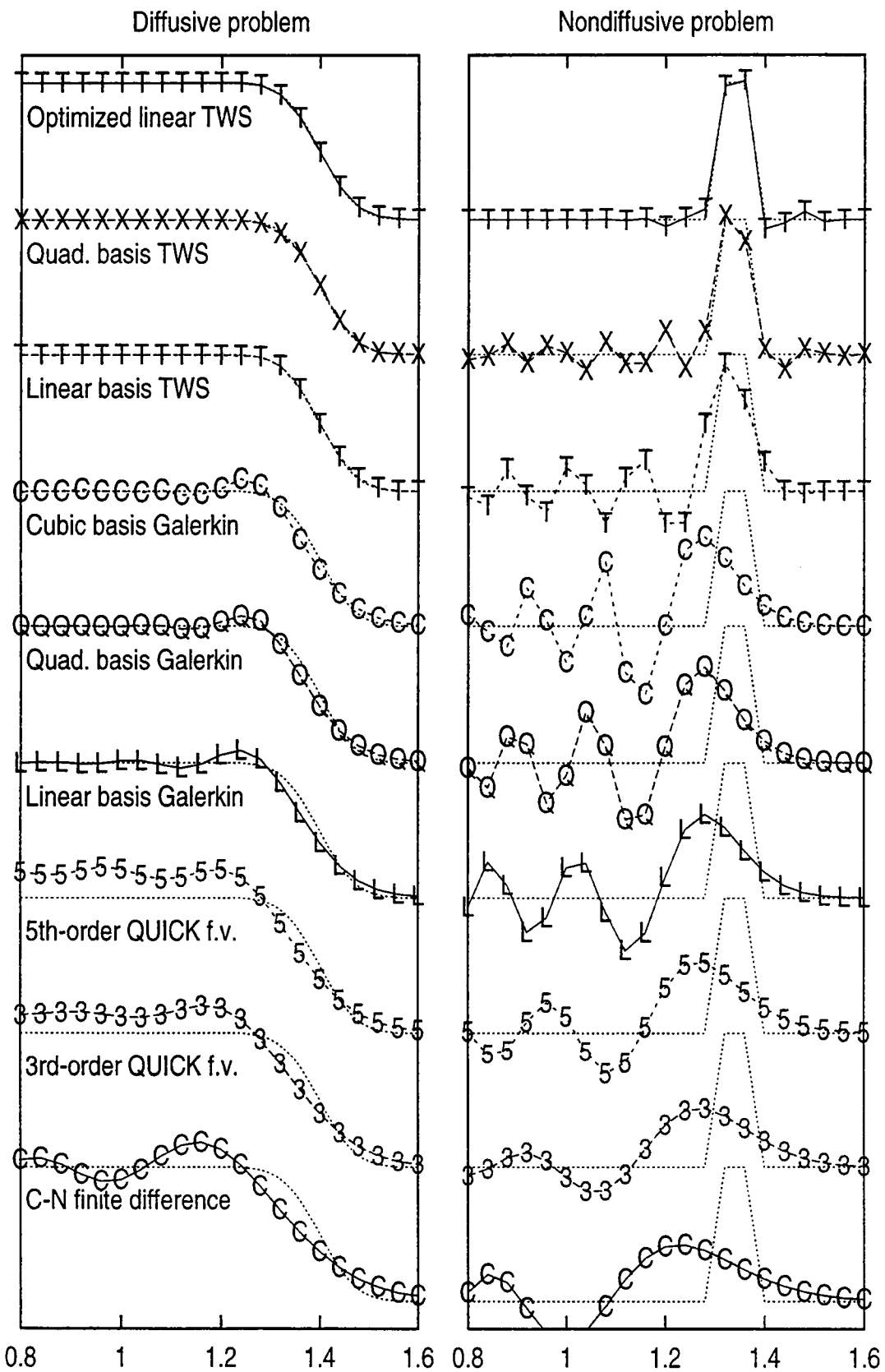

Figure 7.5: One-dimensional simulations at $C = \frac{3}{4}$

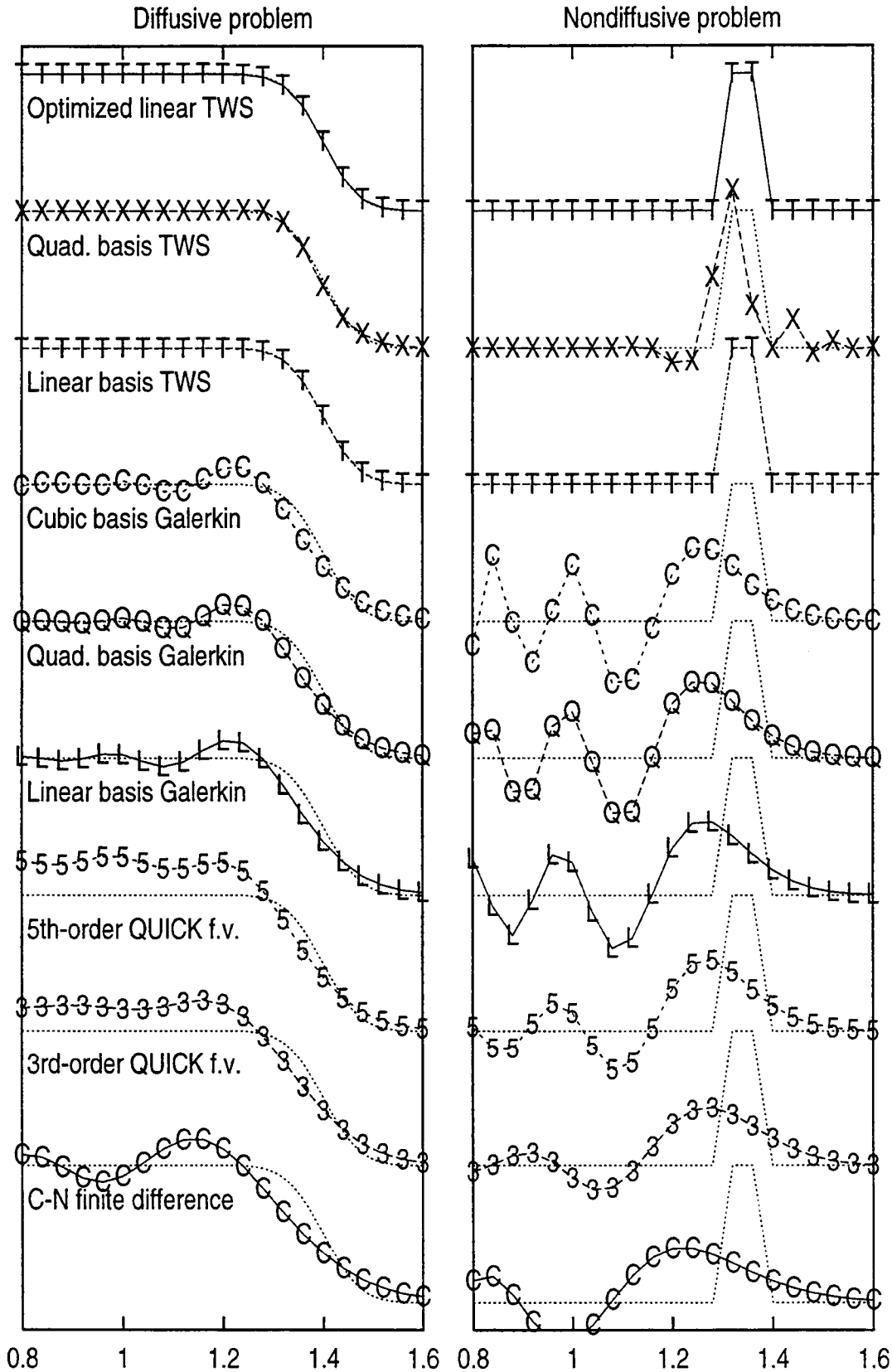

Figure 7.6: One-dimensional simulations at $C = 1$

finite difference algorithm is the least phase accurate and the solution is shown in Figure 7.7b. The QUICK solutions are shown in Figures 7.8a and 7.8b. The linear basis

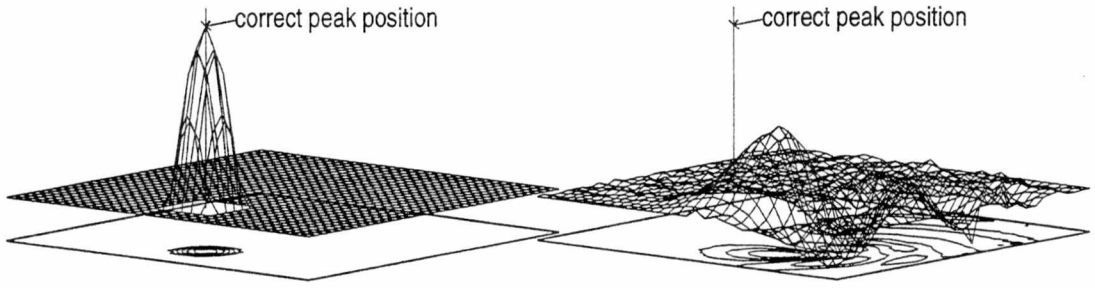

Figure 7.7: (a) Rotating cone analytical solution, (b) Crank-Nicolson solution

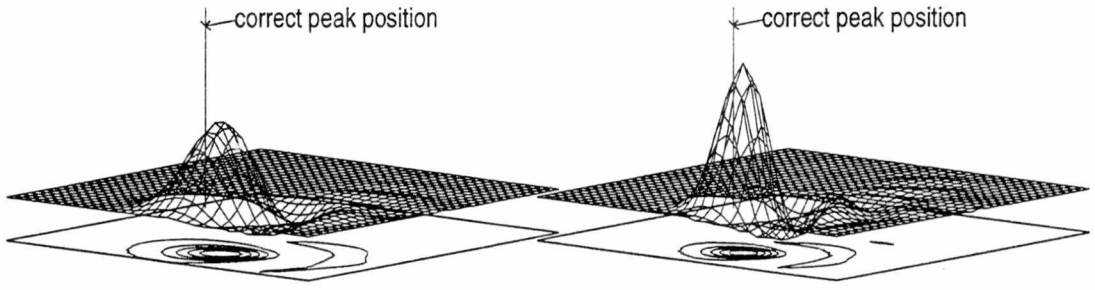

Figure 7.8: (a) 3rd order QUICK solution, (b) 5th order QUICK solution

Galerkin solutions, linear and quadratic basis, are shown in Figure 7.9a and Figure 7.9b. The quadratic and linear basis TWS-γ solutions are shown in Figures 7.10a and 7.10b. The

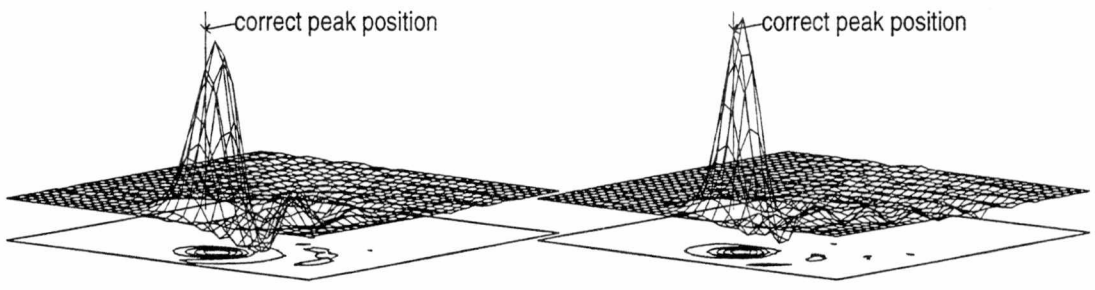

Figure 7.9: (a) Linear basis Galerkin solution, (b) Quadratic basis Galerkin solution

fifth-order optimized linear basis TWS-γ solutions are shown in Figures 7.11a and 7.11b.

The largest relative error in the minimum value, or magnitude of the largest negative error wave, is shown for each algorithm in Figure 7.12. The best performance in this measure is for third-order QUICK, which is highly diffusive, thus damping negative errors, and for the optimized TWS algorithm.

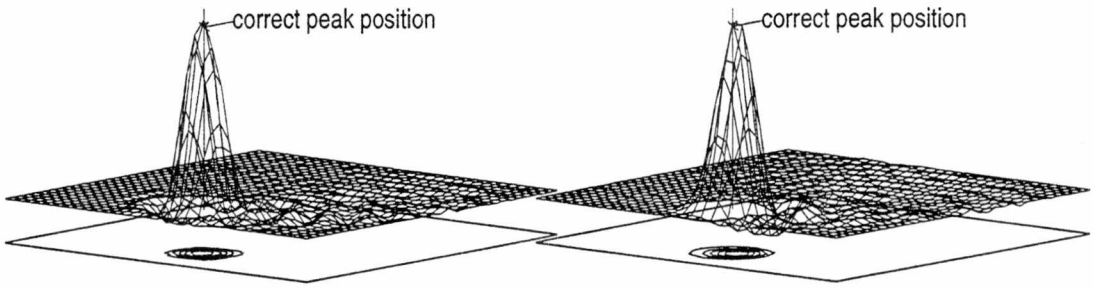

Figure 7.10: (a) Quadratic basis TWS-γ solution, (b) Linear basis TWS-γ solution

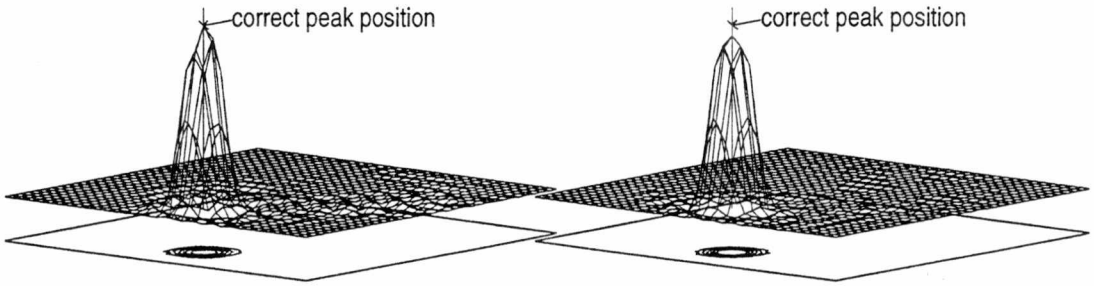

Figure 7.11: (a) Fifth-order TWS-γ solution, (b) Optimized TWS-γ solution

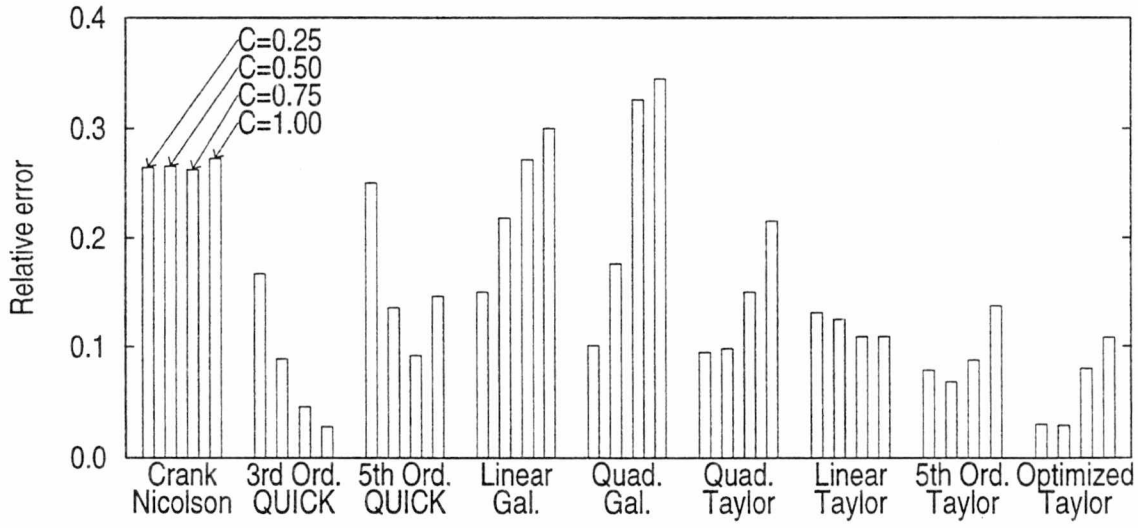

Figure 7.12: Error in rotating cone minimum value by algorithm

The greatest relative error in the maximum value, or deviation of the peak value from one, is summarized in Figure 7.13. The best performance in this measure is for optimized TWS, with the worst Crank-Nicolson and third-order QUICK. The diffusive nature of the QUICK algorithms tends to increase error in this measure. The RMS error or dis-

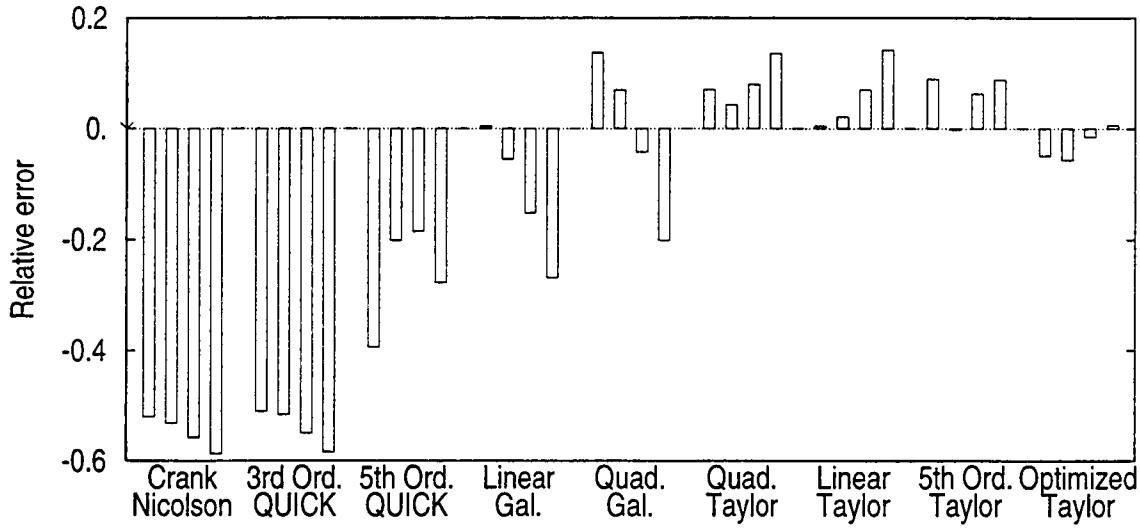

Figure 7.13: Error in rotating cone maximum value by algorithm

crete L^2 error is shown in Figure 7.14. The Taylor algorithms are clearly an improvement in this measure over the others, with Crank-Nicolson significantly worst.

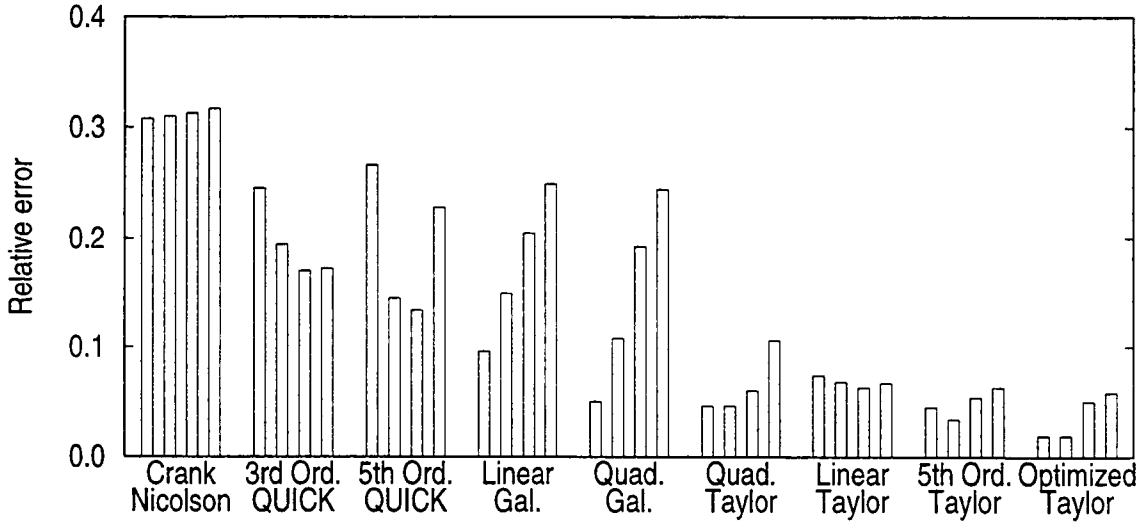

Figure 7.14: RMS error in rotating cone solution by algorithm

GWS and TWS-γ bilinear two-dimensional algorithms were also tested in simula-

tions of the rotating cone problem at $C = \frac{1}{2}$. Both algorithms performed poorly for the non-diffusive test problem, with error waves growing with time. The algorithms were replaced by TWS-β and TWS-γ, β algorithms respectively, both using $\beta = \frac{1}{400}$. The TWS-β solution is shown in Figure 7.15a, with peak reduction of 45% scale and error waves of 18% scale. The TWS-γ, β solution, shown in Figure 7.15b, has peak reduction of 37% scale and error waves of 13% scale. This is clearly an improvement over the bilinear TWS-β but much inferior to the time-split algorithm solutions.

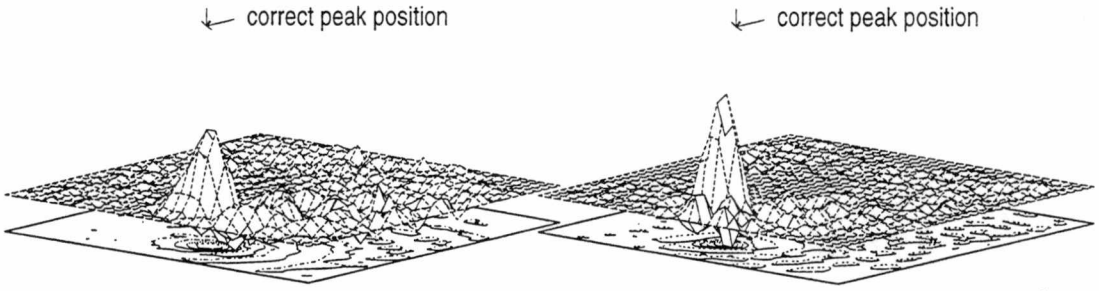

Figure 7.15: (a) TWS-β bilinear solution, (b) TWS-γ,β bilinear solution, $C = \frac{1}{2}$

7.1.3 Steady two-dimensional advection-diffusion verification problem

A boundary value steady-state diffusive solution is known for the advection-diffusion equation, with product forms in one to three dimensions. In this research, the two-dimensional problem is selected. The analytical solution for a constant velocity field $u_x = u_y = 1$ on a unit square is

$$q(x,y) = \frac{(e^{x\mathrm{Pe}} - 1)(e^{y\mathrm{Pe}} - 1)}{(e^{\mathrm{Pe}} - 1)(e^{\mathrm{Pe}} - 1)}\,, \tag{7.3}$$

with boundary conditions zero Dirichlet on the two inlet planes and homogeneous Neumann on the two outlet boundaries, except Dirichlet $q(x,y) = 1$ at the outflow corner $x = y = 1$. The exact nodal solution is graphed for $\mathrm{Pe} = 10^6$ in Figure 7.16; at large Peclet number the solution is essentially zero everywhere except the outlet corner node.

Approximate solutions were calculated using various algorithms for $\mathrm{Pe} = 10^6$ on a 10×10 element (11×11) nodal mesh. The GWS nodal solution is shown in Table 7.3; dispersion error waves of wavelength $2\Delta_x$ and $2\Delta_y$ are evident.

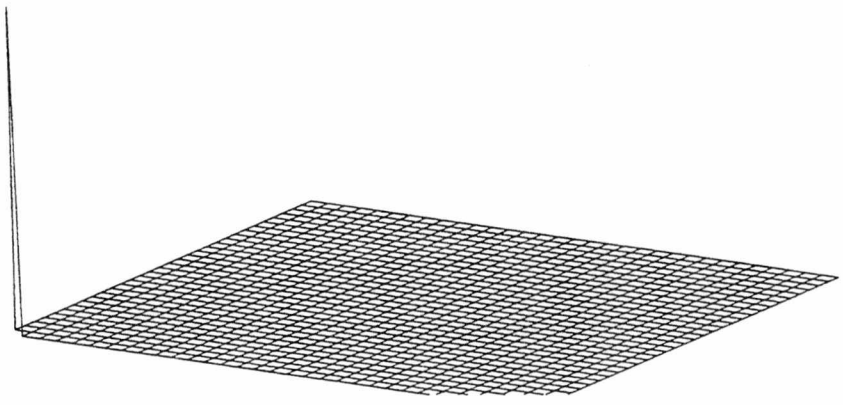

Figure 7.16: Steady-state advection-diffusion analytical solution, flow to left

Table 7.3: GWS steady advection-diffusion results (x 1000)

0	0	1	-1	0	0	4	-19	74	-270	1000
0	0	0	-1	1	2	-12	41	-144	534	-270
0	0	-1	1	3	-12	39	-134	499	-144	74
0	-1	1	3	-13	39	-133	495	-134	41	-19
0	0	3	-13	39	-132	494	-133	39	-12	4
0	3	-12	38	-132	493	-132	39	-12	2	0
0	-10	38	-132	493	-132	39	-13	3	1	0
0	34	-132	494	-132	38	-13	3	1	-1	-1
0	-123	492	-132	38	-12	3	1	-1	0	1
0	461	-123	34	-10	3	0	-1	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Each of the artificial diffusion methods for this and subsequent problems was formulated as a correction matrix, i.e. an assembled modified matrix minus the corresponding assembled GWS matrix. This correction matrix was multiplied in each case by a to-be-determined scalar to achieve the best results for the method, with the scalar multiplier being unity for the default version of the method. In the TWS-β method, this multiplier is the constant β, but the other methods do not explicitly include a (linearly) adjustable artificial diffusion constant. For this advection-diffusion problem, the optimal solution was determined somewhat arbitrarily as that with zeroes in the (1,10) and (2,11) nodal locations on the outlet boundaries adjacent to the corner boundary condition.

The MPG-1 nodal solution is shown in Table 7.4 obtained with multiplier 0.154. The MPG-2 solution is shown in Table 7.5 obtained with multiplier 0.000000743, and with

Table 7.4: MPG-1 steady advection-diffusion results (x 1000)

0	0	0	0	0	0	0	1	7	0	1000
0	0	0	0	0	0	0	0	2	21	0
0	0	0	0	0	0	0	0	1	2	7
0	0	0	0	0	0	0	0	0	0	1
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	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	0	0	0	0	0	0	0

the lower six rows, all zero, omitted. The TWS-β solution is shown in Table 7.6 obtained using multiplier $\beta = 1.125$, with the lower six zero rows omitted. The SGM solution is shown in Table 7.7 obtained using multiplier 0.0664, with the lower six zero rows omitted.

The EPG and BPG methods could not produce a solution satisfying the selected criteria, in spite of the EPG algorithm apparently being derived to fit the solution of this verification problem [18]. The MPG-1 solution was the best by the measure of the interior corner (2,10) entry, analytically almost zero, and ranging from -0.042 to 0.077 for the four algorithms which could produce a solution.

7.2 Verification simulations for Burgers equation

The unsteady viscous Burgers equation in one dimension is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \frac{\partial}{\partial x}\left(\frac{1}{\mathrm{Re}}\frac{\partial u}{\partial x}\right) = 0\,. \tag{7.4}$$

Table 7.5: MPG-2 steady advection-diffusion results (x 1000)

0	0	0	0	0	0	0	1	-5	0	1000
0	0	0	0	0	0	0	0	2	35	0
0	0	0	0	0	0	0	0	2	2	-5
0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0

Table 7.6: TWS-β steady advection-diffusion results (x 1000)

0	0	0	0	0	0	1	2	19	0	1000
0	0	0	0	0	0	0	3	4	77	0
0	0	0	0	0	0	0	1	7	4	19
0	0	0	0	0	0	0	1	1	3	2
0	0	0	0	0	0	0	0	0	0	1

Table 7.7: SGM steady advection-diffusion results (x 1000)

0	0	0	0	0	0	0	1	-4	0	1000
0	0	0	0	0	0	0	0	-4	-42	0
0	0	0	0	0	0	0	0	1	-4	-4
0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0

A steady-state analytical solution is

$$u = \tanh\left(\frac{-x\,\mathrm{Re}}{2}\right), \tag{7.5}$$

with boundary conditions $u(x=-1)=1$ and $u(x=1)=-1$ on the domain $-1 \leq x \leq 1$. The steady GWS approximate solution with $n=20$, $\mathrm{Re}=10^5$ is shown in Figure 7.17 for the middle of the domain. Most of the tested artificial diffusion methods were able to

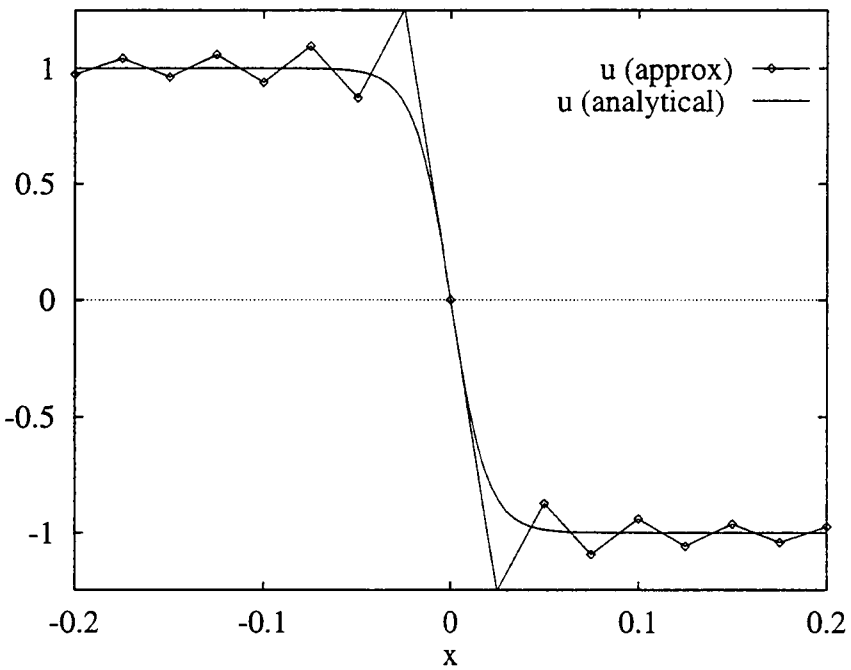

Figure 7.17: Burgers equation GWS solution

dampen the oscillations in a steady simulation. Again, the EPG and BPG methods did not produce a reasonable solution. A steady TWS-β solution is shown in Figure 7.18 using $\beta = 1$. A steady MPG-1 solution is shown in Figure 7.19 using multiplier 0.0833. A steady MPG-2 solution is shown in Figure 7.20 using multiplier 0.000025. A steady SGM solution is shown in Figure 7.21 using multiplier 40. As an example of excess artificial diffusion, a TWS-β solution is shown in Figure 7.22 using multiplier $\beta = 10$. The diffusive effect on the solution is obvious.

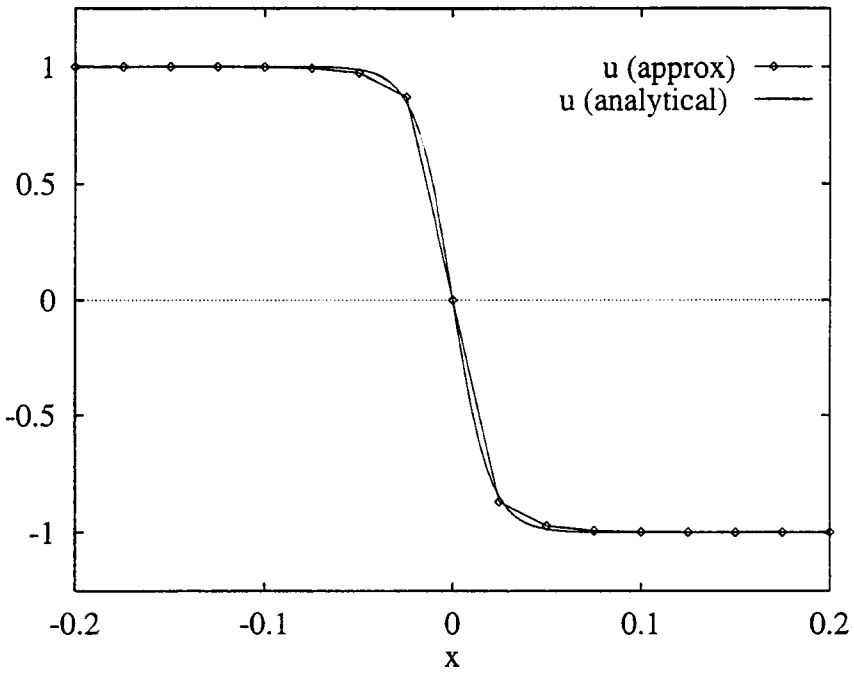

Figure 7.18: Burgers equation TWS-β solution

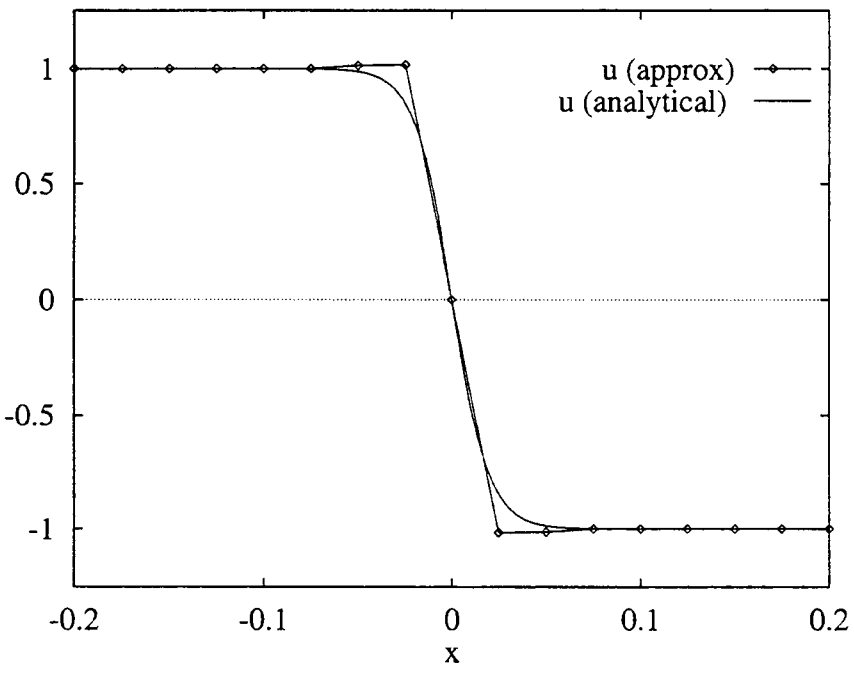

Figure 7.19: Burgers equation MPG-1 solution

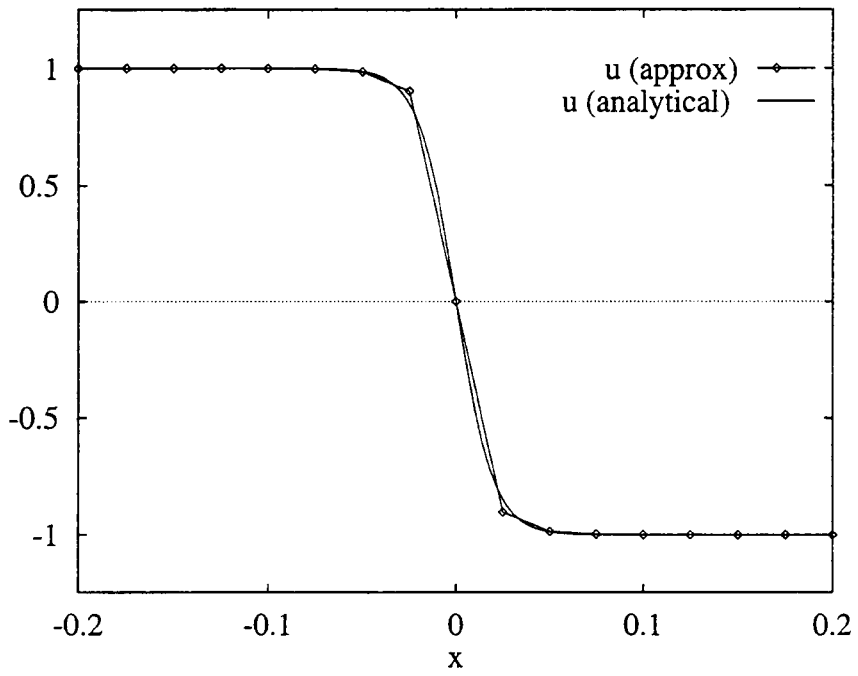

Figure 7.20: Burgers equation MPG-2 solution

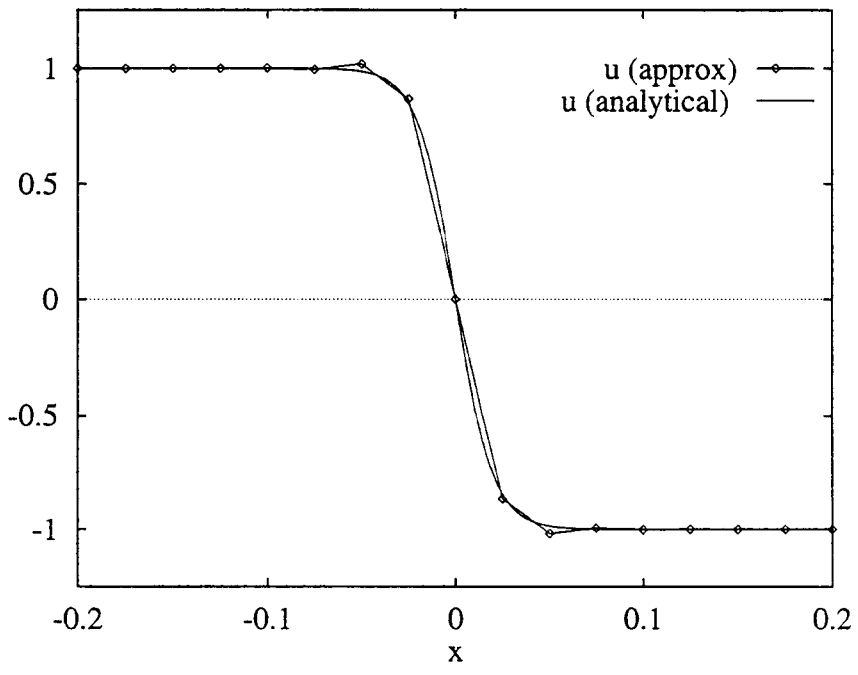

Figure 7.21: Burgers equation SGM solution

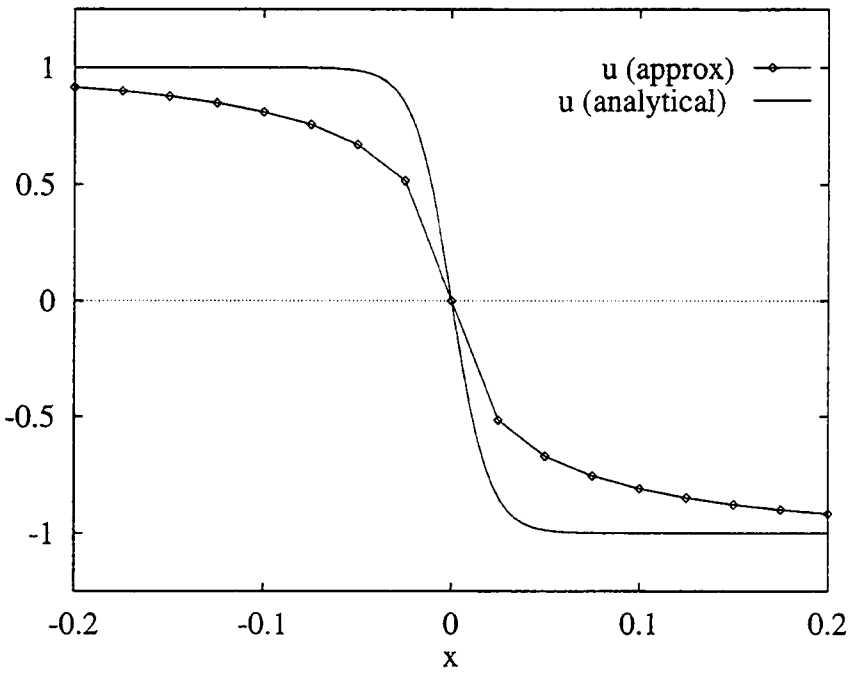

Figure 7.22: Burgers equation TWS-β solution

7.2.1 Steady two-dimensional advection-diffusion with source verification simulation

A steady-state two- or three-dimensional verification solution with a source is known as the Gaussian plume, a solution of

$$u\frac{\partial q}{\partial x} - \frac{\partial}{\partial y}(\nu\frac{\partial q}{\partial y}) - \frac{\partial}{\partial z}(\nu\frac{\partial q}{\partial z}) - s = 0\,, \tag{7.6}$$

with no diffusion in the x-direction and with delta function source s at $x = 0$. Heines and Peters [47] develop an analytical steady-state two-dimensional solution for a uniform x-velocity, uniform y-diffusion k, and a line source at $(x = 0, y = 0)$ as

$$q(x,y) = \frac{s}{\sqrt{\pi\nu ux}}\exp\left(\frac{-uy^2}{4\nu x}\right)\,. \tag{7.7}$$

A corresponding three-dimensional solution is known; in this case the x-coordinate conditions are identical, while zero velocity with constant diffusion is assigned along y and z

coordinates, with solution

$$q(x,y,z) = \frac{s}{\pi \nu u x} \exp\left(\frac{-u(y^2+z^2)}{4\nu x}\right) . \tag{7.8}$$

The Gaussian plume analytical solution is used in air-pollution analysis of multiple sources by superposition [27]. Since the plume solution requires (physically incorrect) assumptions of uniform one-dimensional velocity and uniform diffusion, an approximate method such as those developed here, not requiring those assumptions, could potentially provide a more accurate analysis than the simplified analytical solution. However, the assumptions of uniform velocity and diffusion were made for the two-dimensional verification simulation.

Steady-state GWS approximations were made to (7.7) with arbitrary data of $s = 0.00375$, $\nu = 0.01$ and $u = 1$ on a unit square $(0 \leq x \leq 1, -0.5 \leq y \leq 0.5)$. The entire simulated source was located at the second node on the x-axis. A 10×10 uniform-mesh GWS solution is shown in Figure 7.23. The two upper curves are the analytical (solid line) and simulated (symbols) profile of concentration along the x-axis, with the two lower curves the analytical (solid line) and simulated (symbols) transverse profile of concentration on the outlet boundary. On this coarse mesh, the simulated concentration at the first interior node in the x-direction is 29% of the analytical, while at the outlet node on the x-axis it is 85% of analytical. A 80×80 uniform-mesh GWS approximation is shown in Figure 7.24. Here the simulated concentration at the first interior x-node is 70% of analytical, and the simulated concentration at the last x-axis node is 99.95% of analytical, and mesh resolution near the source is shown to be mandatory.

7.3 Verification and benchmark simulations for the Navier-Stokes equations

Exact solutions for the Navier-Stokes equations are for few and trivial situations. Therefore, comparison of approximations is usually based on accepted accurate or "benchmark" solutions. However, one exact solution was simulated to verify proper operation of the code.

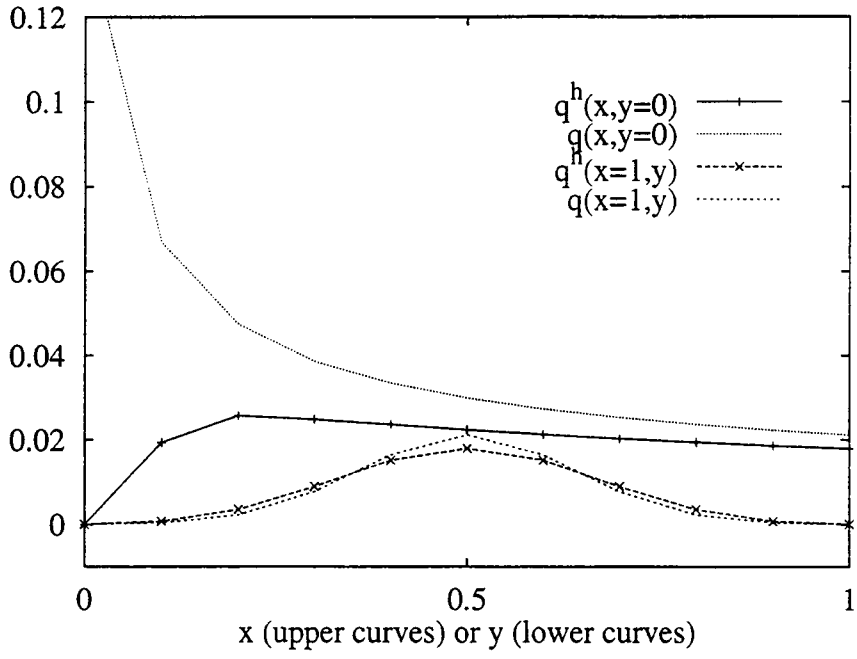

Figure 7.23: Gaussian plume, GWS 10×10

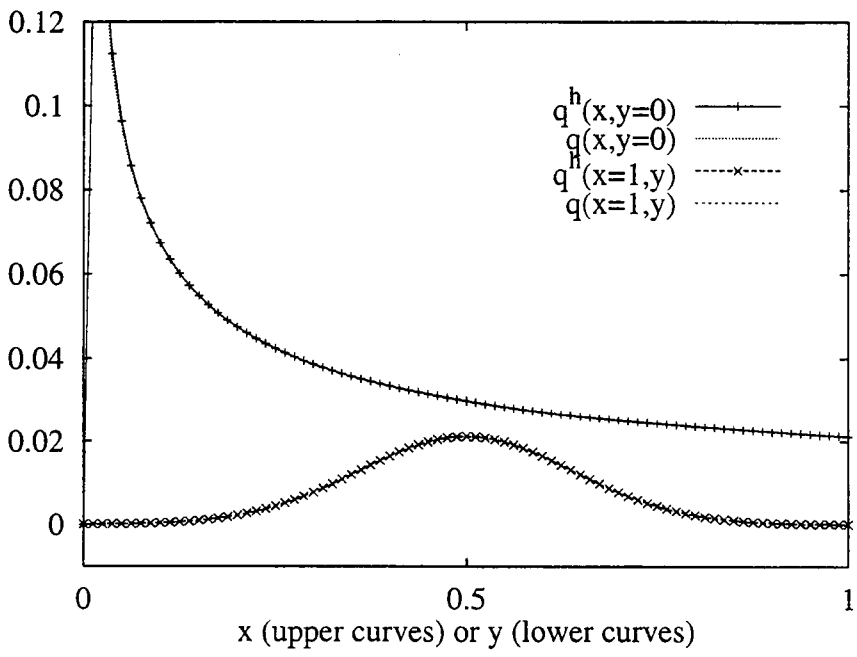

Figure 7.24: Gaussian plume, GWS 80×80

7.3.1 Duct flow verification simulation

A developing two-dimensional duct flow with Re = 10 and slug flow input $u = 1$ was simulated on a unit width duct with length 10. The flow should be fully developed at $x > \frac{0.1}{\mathrm{Re}}$ [86], easily satisfied with these data. The fully developed channel flow solution is

$$\begin{aligned} u(y) &= 6y(1-y)\,, \\ \frac{\partial}{\partial x}p(x) &= -\frac{12}{\mathrm{Re}}\,. \end{aligned} \tag{7.9}$$

For this simulation, the pressure level and slug flow profile were defined on the inlet boundary. The GWS solution on a 32×16 non-uniform mesh is graphed for inlet and outlet boundaries in Figure 7.25. The inlet velocity level was multiplied by a factor of 1.0625 to correct the total mass flow for the no-slip wall. The resulting outlet velocity is a parabolic profile with maximum 1.507 compared to the analytical value of 1.5. Outlet y-velocity and pressure distributions (relative to the boundary mean) are shown to have maximum values of $O(10^{-4})$ compared to the analytical zero.

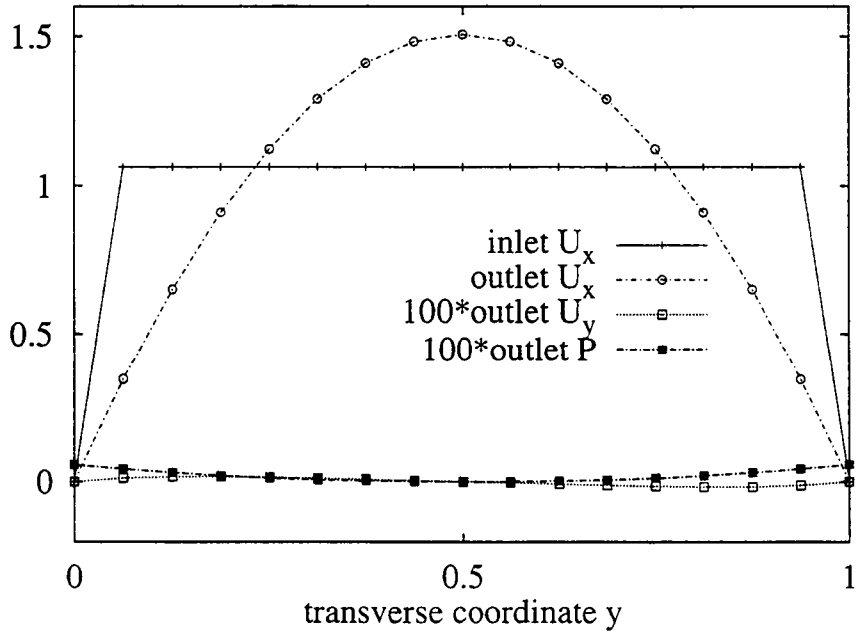

Figure 7.25: Developing duct flow transverse velocity/pressure profiles , Re=10

7.3.2 Driven cavity benchmark simulation

A widely-used steady benchmark solution for viscous flow is the two-dimensional driven cavity, a square cavity with a sliding lid translating at constant velocity $u_x(x, y = 1) = 1$. The sliding lid induces a rotational flow in the cavity with considerable vorticity at larger Reynolds numbers. Fine-mesh solutions have been published by Ghia et al. [39] and others using stream function-vorticity algorithms. Relatively few solutions have been published using pressure-based algorithms ([7], [66]).

Evaluation the x-momentum equation along the sliding lid ($u_x = 1, u_y = 0$) at the cavity-lid corners yields

$$\frac{\partial p}{\partial x} = \frac{1}{\mathrm{Re}} \frac{\partial^2 u_x}{\partial y^2} , \tag{7.10}$$

which is infinite for the analytical problem statement. This relation has apparently been replaced by some difference approximation for the few pressure-based solutions reported in the literature, but details of those approximations were not discussed. Selected finite difference relations for (7.10) were not successful in this study.

GWS simulations were performed using $\frac{\partial^2 u_x}{\partial y^2} \approx 150$ in (7.10) regardless of mesh, chosen to approximate the pressure fields reported by Aubert and Deville [7] at $\mathrm{Re} = 100$. Resulting velocity fields are compared with the results of Ghia et al. [39] in Figure 7.26. The x-velocity profiles of Ghia et al. along the vertical centerline, and y-velocity profiles along the horizontal centerline, are compared with results of this study on 24×24 and 72×72 uniform meshes. Comparing the largest negative x-velocity, the 24×24 mesh produced about -0.158, the 72×72 mesh produced about -0.190, and Ghia et al. had about -0.211, thus convergence to the Ghia et al. solution is demonstrated.

Similar velocity profiles at $\mathrm{Re} = 400$ are shown for a GWS simulation in Figure 7.27. Comparing the largest negative x-velocity, the 24×24 mesh produced about -0.274, the 72×72 mesh produced about -0.301, and Ghia et al. had about -0.328. Again convergence to the Ghia et al. solution is demonstrated. However, because of the difficulties with boundary conditions and the lack of experimental data, this verification problem was not further considered in this study.

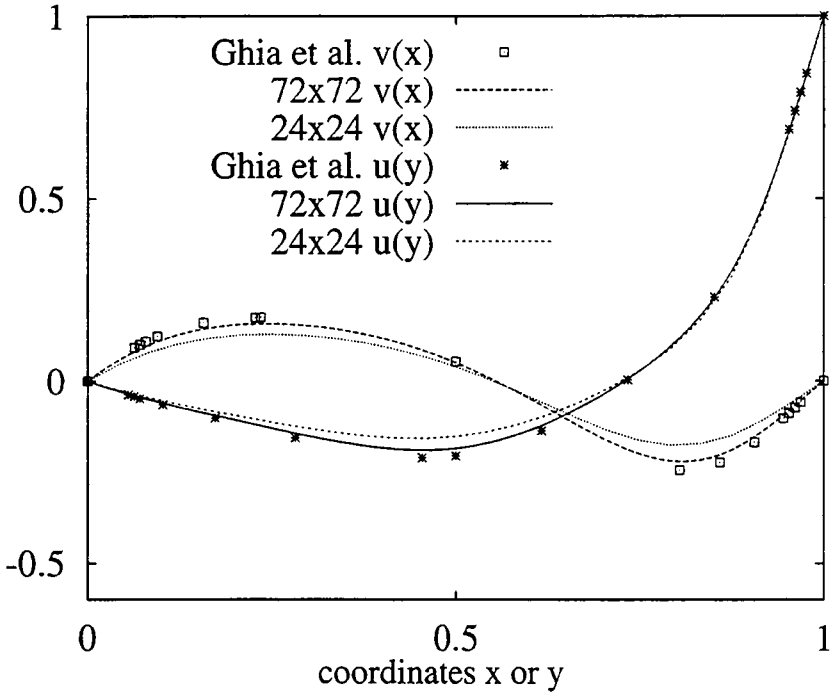

Figure 7.26: Driven cavity velocity profiles, Re=100, vs. Ghia et al.

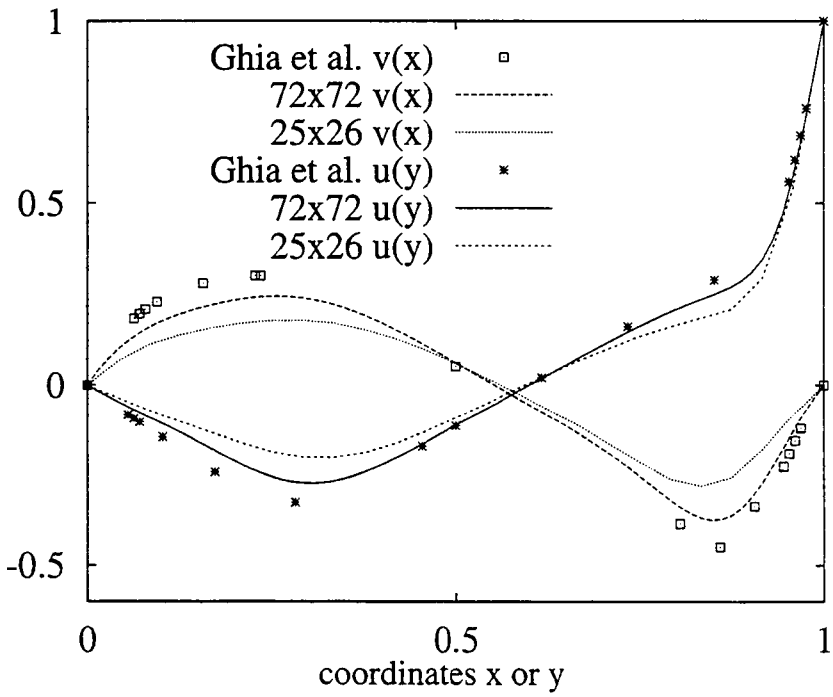

Figure 7.27: Driven cavity velocity profiles, Re=400, vs. Ghia et al.

7.3.3 Thermal cavity benchmark simulation

A widely used steady-flow benchmark is the two-dimensional thermal cavity, a square cavity with natural convection induced by one heated ($\theta = 1$) vertical wall and one cooled vertical wall ($\theta = 0$) with horizontal walls insulated. The problem is parameterized by the Rayleigh and Prandtl numbers, and fine-mesh solutions have been published by de Vahl Davis [25], Le Quéré [63], and Comini et al. [23]. Features of the flow at high Rayleigh number are illustrated in Figure 7.28.

The thermal cavity benchmark problem was simulated for Rayleigh numbers of 10^3 to 10^8 on non-uniform meshes of 24×24 to 72×72 bilinear elements. Results were compared to the benchmark results of de Vahl Davis [25], Le Quéré [63], and Comini et al. [23], whose reported velocities were rescaled for this study since the various authors used different nondimensionalizations. Specifically, de Vahl Davis used $\mathrm{Re} = 1/\mathrm{Pr}$ and $\mathrm{Gr}/\mathrm{Re}^2 = \mathrm{RaPr}$, Le Quéré and Comini et al. used $\mathrm{Re} = \mathrm{Ra}^{0.5}/\mathrm{Pr}$ and $\mathrm{Gr}/\mathrm{Re}^2 = \mathrm{Pr}$, and this study used $\mathrm{Re} = \mathrm{Ra}/\mathrm{Pr}^{0.5}$ and $\mathrm{Gr}/\mathrm{Re}^2 = 1$. Each author used nondimensional temperature of 0 to 1 and $\mathrm{Pr} = 0.71$, thus the various results only differ by the scale factor of velocity. This was confirmed by simulations (not reported) using the other two nondimensionalizations.

de Vahl Davis, Le Quéré, and Comini et al. each used vorticity-stream function algorithms, the first two finite difference and Comini et al. finite element. de Vahl Davis and Le Quéré reported results including the largest magnitude negative x-velocity along the vertical centerline, largest negative y-velocity along the horizontal centerline, maximum and centerline $y = 0.5$ Nusselt numbers along the heated side, and several vorticity measures not computed in this study. Comini et al. reported only the maximum and centerline Nusselt numbers. Energy norms were computed in this study, which are not available in the known benchmark studies.

Results of this study are summarized and compared with the benchmarks in Table 7.8. Reported results include the mentioned velocities and Nusselt numbers along with summed velocity-temperature energy norm $||q^h||_E$, pressure correction norm $||\phi^h||_E$, and pressure Poisson norm $||p^h||_E$.

Table 7.8 shows that the simulations of this study produced centerline Nusselt numbers 1% to 7% higher than de Vahl Davis at Rayleigh numbers up to 10^5 on the finest grid, and produced centerline Nusselt numbers 5% to 12% higher than Le Quéré at larger

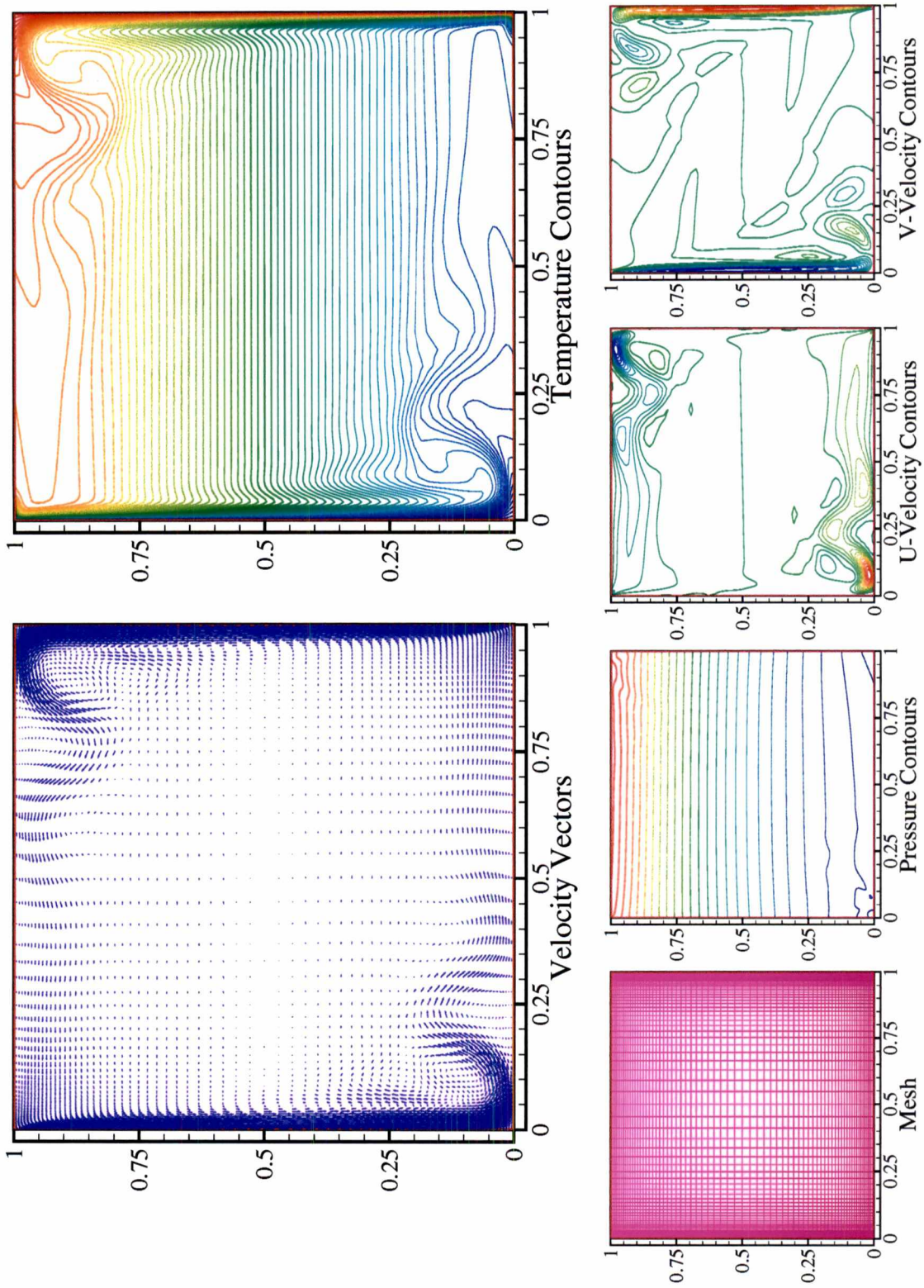

Figure 7.28: Thermal cavity simulation results, Ra=10^8, 72x72 mesh

Table 7.8: Summary of GWS non-uniform mesh thermal cavity results

Ra	mesh	Ux_{min}	Uy_{min}	Nu_{max}	$Nu_{0.5}$	$\|\|q^h\|\|_E$	$\|\|\phi^h\|\|_E$
10^3	24x24	0.14091	0.14031	1.505	1.1277	5.51151 e-2	2.556 e-7
10^3	42x42	0.13745	0.13898	1.506	1.1305	5.52253 e-2	2.306 e-7
10^3	54x54	0.13750	0.13898	1.506	1.1307	5.52398 e-2	2.273 e-7
10^3	64x64	0.13760	0.13923	1.506	1.1308	5.52456 e-2	2.261 e-7
10^3	72x72	0.13758	0.13916	1.506	1.1309	5.52485 e-2	2.256 e-7
10^3	Davis	0.13690	0.13870	1.505	1.1180	n/a	n/a
10^4	24x24	0.19844	0.23663	3.533	2.3953	3.92422 e-2	3.691 e-7
10^4	42x42	0.19327	0.23349	3.541	2.4013	3.92826 e-2	2.654 e-7
10^4	54x54	0.19298	0.23334	3.542	2.4014	3.93004 e-2	2.582 e-7
10^4	64x64	0.19283	0.23307	3.541	2.4020	3.93056 e-2	2.559 e-7
10^4	72x72	0.19275	0.23333	3.543	2.4018	3.93079 e-2	2.545 e-7
10^4	Davis	0.19200	0.23280	3.528	2.2430	n/a	n/a
10^5	24x24	0.13010	0.26253	7.621	4.6067	2.40049 e-2	1.433 e-7
10^5	42x42	0.13079	0.25686	7.731	4.6373	2.41939 e-2	9.405 e-8
10^5	54x54	0.13100	0.25875	7.741	4.6395	2.42173 e-2	8.875 e-8
10^5	64x64	0.13090	0.25808	7.745	4.6400	2.42248 e-2	8.710 e-8
10^5	72x72	0.13089	0.25837	7.746	4.6404	2.42277 e-2	8.635 e-8
10^5	Davis	0.13030	0.25600	7.717	4.5190	n/a	n/a
10^5	Comini et al.	n/a	n/a	7.704	4.5030	n/a	n/a
10^6	24x24	0.07418	0.27008	16.148	8.1423	1.42109 e-2	1.927 e-7
10^6	42x42	0.07688	0.26439	17.400	8.3283	1.45394 e-2	1.032 e-7
10^6	54x54	0.07679	0.26337	17.515	8.3448	1.45687 e-2	9.785 e-8
10^6	64x64	0.07684	0.26283	17.518	8.3489	1.45881 e-2	9.675 e-8
10^6	72x72	0.07685	0.26248	17.537	8.3504	1.45920 e-2	9.644 e-8
10^6	Davis	0.07670	0.26030	17.930	8.8170	n/a	n/a
10^6	Le Quéré	0.06483	0.22060	17.536	8.8252	n/a	n/a
10^6	Comini et al.	n/a	n/a	17.400	8.8220	n/a	n/a
10^7	24x24	0.05404	0.29396	27.466	13.696	7.81349 e-3	4.158 e-7
10^7	42x42	0.05616	0.26196	36.753	14.772	8.38158 e-3	2.166 e-7
10^7	54x54	0.05647	0.26341	37.938	14.873	8.44949 e-3	2.153 e-7
10^7	64x64	0.05650	0.26280	38.270	14.897	8.46717 e-2	2.204 e-7
10^7	72x72	0.05642	0.26358	38.336	14.904	8.47409 e-3	2.238 e-7
10^7	Le Quéré	0.04690	0.22120	39.350	16.560	n/a	n/a
10^7	Comini et al.	n/a	n/a	38.920	16.528	n/a	n/a
10^8	24x24	0.04344	0.34531	31.778	20.176	4.02743 e-3	1.597 e-6
10^8	42x42	0.04484	0.26058	61.558	25.677	4.61221 e-3	4.216 e-8
10^8	54x54	0.04354	0.26724	69.752	26.361	4.75060 e-3	5.184 e-8
10^8	64x64	0.04502	0.26735	71.469	26.539	4.78339 e-3	6.142 e-8
10^8	72x72	0.04558	0.26592	71.562	26.604	4.79444 e-3	6.917 e-8
10^8	Le Quéré	0.03200	0.22220	87.210	30.220	n/a	n/a
10^7	Comini et al.	n/a	n/a	84.380	30.161	n/a	n/a

Rayleigh numbers. The three reference benchmark solutions differ in Nusselt number from each other by up to 1% at the common point of $Ra = 10^6$.

The centerline Nusselt numbers of the thermal cavity simulations are graphed as a function of mesh in Figure 7.29 with ordinate of benchmark Nusselt number minus simulation Nusselt number, divided by the benchmark. The present simulation Nusselt numbers are shown to approach grid-independent values, and the relative offset from the x-axis indicates a difference from the benchmark values.

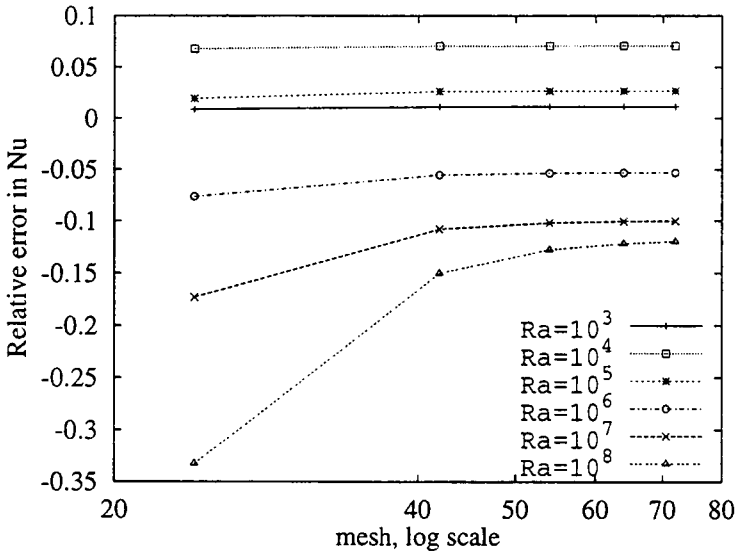

Figure 7.29: Thermal cavity Nusselt numbers, relative error vs. benchmarks

Horizontal and vertical velocity and Nusselt number profiles along the cavity centerlines for a 72×72 mesh are shown with arrows indicating de Vahl Davis' published maximums for Rayleigh numbers of 10^5 in Figure 7.30. The agreement is graphically exact.

Horizontal and vertical velocity and Nusselt number profiles along the cavity centerlines for a 72×72 mesh are shown with arrows indicating Le Quéré's published maximums for Rayleigh numbers of 10^8 in Figure 7.31. Obvious disagreement occurs on the left wall. The very thin boundary layer at this Rayleigh number may indicate an inadequate resolution for the current simulations, which contributes to the comparatively low calculated Nusselt numbers. Further, the utilized first-order finite difference formula used only the first off-wall nodal temperature, which will induce a substantial error in Nusselt number if the node is insufficiently refined near the wall, apparently the case.

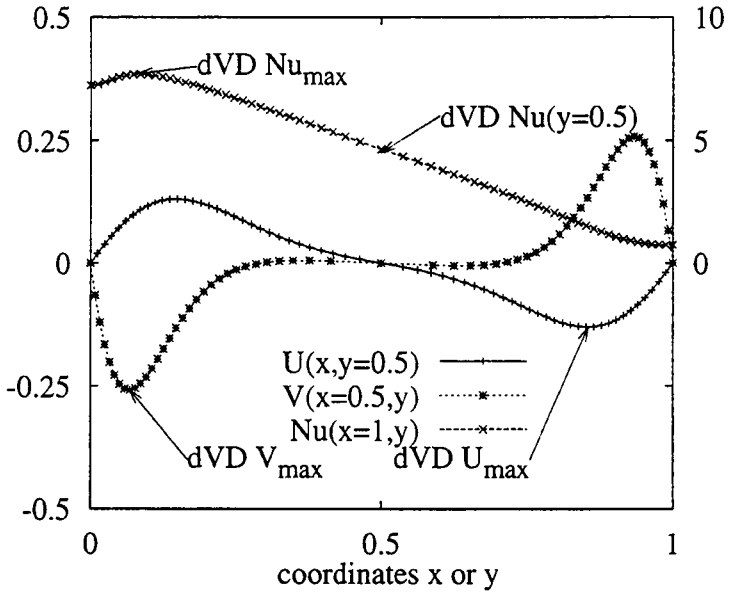

Figure 7.30: Thermal cavity velocity/Nu profiles, Ra = 10^5, vs. de Vahl Davis

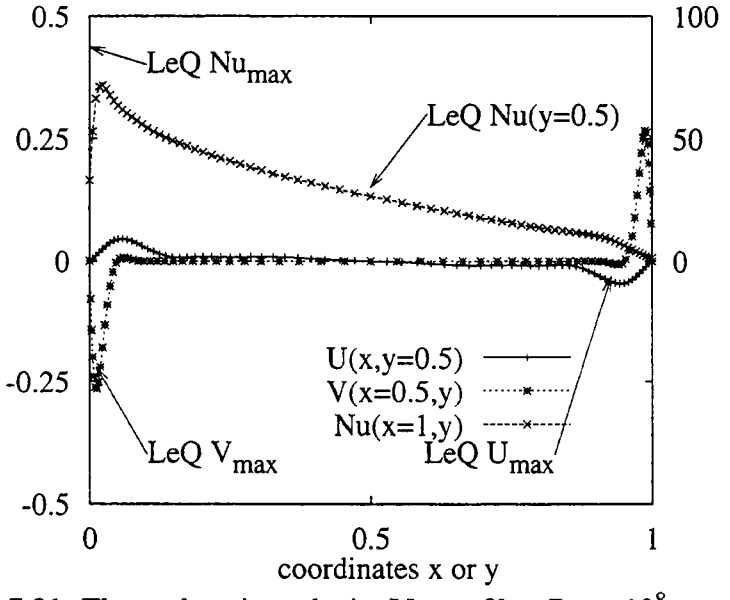

Figure 7.31: Thermal cavity velocity/Nu profiles, Ra = 10^8, vs. Le Quéré

Error measured in the energy norm of the thermal cavity simulations is graphed in Figure 7.32 as estimated infinite-mesh energy norm $||q||_E$ minus approximation energy norm $||q^h||_E$ according to (3.52). The infinite-mesh energy norm $||q||_E$ is estimated using a slope of $n = -2$ for the two finest grid solutions. As shown in Figure 7.32, the energy norm deviations (points and dashed lines) correctly approach the $n = -2$ slope (solid lines) as the mesh is refined. A useful accuracy measure is then obtained which may be used on any simulation without requiring reference to possibly unavailable benchmark solution point data.

A coarse-grid high-Rayleigh cavity ($\mathrm{Ra} = 10^8$ on a 42×42 grid) was simulated with each of the artificial diffusion algorithms of Table 6.1. None of the forms of artificial diffusion appeared to have any beneficial (or detrimental) effect on thermal cavity simulations. The reason is that natural convection velocities are relatively small, thus velocity-based artificial diffusion methods are expected to be ineffective.

7.3.4 Backward-facing step benchmark simulation

The backward-facing step [38],[86], or duct sudden expansion is a steady-state two- or three-dimensional benchmark problem which has through-flow boundary conditions, much more typical of actual flows than the closed cavity problems. In this research, the step geometry was chosen to match the experimental data of Armaly et al. [6]. The results of Williams [86] show perceptible differences for that geometry between two-dimensional and mid-plane three-dimensional simulations beginning at $\mathrm{Re} \approx 400$.

The two-dimensional backward-facing step was simulated for Reynolds numbers (based on step height) of 50 to 1000 using several non-uniform meshes. Results were compared to the two-dimensional benchmark results of Gartling [38], Roy and Baker [70], and Freitas [37], and the two- and three-dimensional results of Williams [86], as well as the experimental results of Armaly et al. [6]. The usual measure of accuracy comparison is the primary vortex re-attachment length (designated X_1 by Armaly). Some authors also reported re-attachment points of several secondary vortices. At Reynolds numbers of 500 to 1000, other relevant comparison points are X_4, the initiation point of a vortex on the upper wall and X_5, the termination point of that vortex.

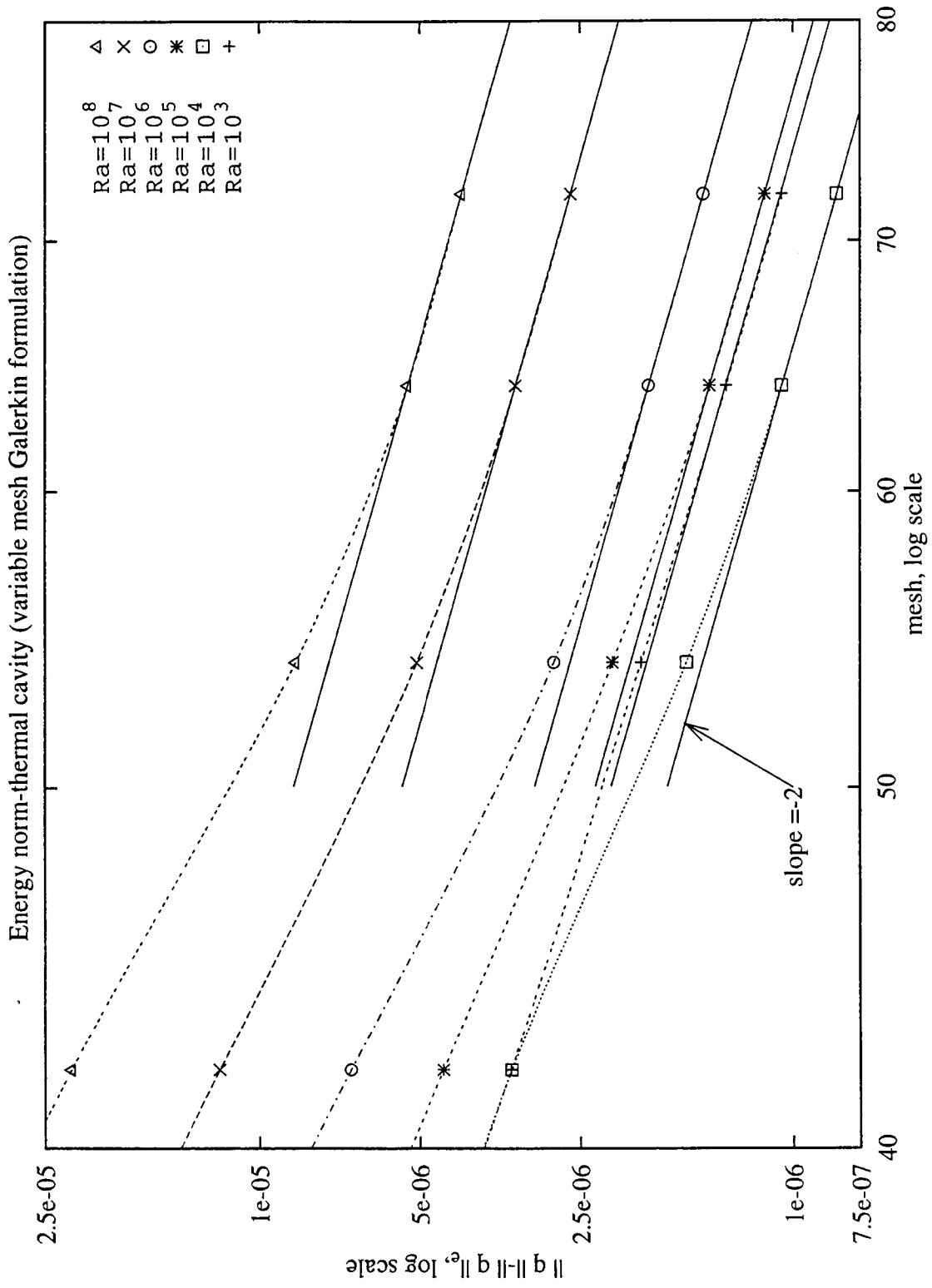

Figure 7.32: Thermal cavity energy norm deviation

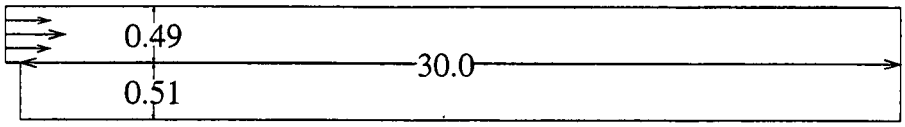

Figure 7.33: Backward-facing step geometry

Geometry of the backward-facing step is shown in Figure 7.33. Features of the two-dimensional flow at Re = 1000 are compared in Figures 7.34 (31×200 mesh GWS) and 7.35 (31×74 mesh TWS-β). Both upper and lower recirculation regions are predicted, with the lower region of length about 12 step heights. Computing a two-dimensional flow at this Reynolds number is strictly a numerical test, since the physical flow is significantly three-dimensional. However, the multiple measurable vortices, susceptible to excess numerical diffusion effects, make this flow an attractive benchmark problem reported by the several authors cited.

Results of the simulations are summarized and compared with literature results in Table 7.9. The data shown are the three re-attachment lengths X_1, X_4, and X_5, summed solution velocity norm $||q^h||_E$, the pressure correction norm $||\phi^h||_E$, and the pressure Poisson norm $||p^h||_E$.

A GWS algorithm with a 100-element (in the x-direction) nonuniform bilinear mesh produced stable solutions up to Re = 800. GWS methods using 200 and 300-element bilinear meshes produced stable solutions up to Re = 1000. The finer meshes with the GWS method produced primary re-attachment length X_1 0.2% lower than Gartling [38] (-2, finer mesh) at Re = 800. The upper re-attachment initial length X_4 was over-predicted by 8% and final length X_5 under-predicted by 4% compared to Gartling.

The 31×300 Re = 800 GWS solution was projected onto 31×50 meshes using Tecplot® to provide the initial condition for artificial diffusion simulations by the TWS-β/SUPG SGM, MPG-1, MPG-2, EPG, and BPG algorithms. Each method was able to produce a stable solution with the multiplier (shown x0.2, etc. in the table) on the artificial diffusion adjusted for the range of results. The best of the coarse-grid artificial diffusion solutions over-predicted X_1 by 1% and the worst by 7%. Each artificial diffusion solution overestimated X_1 and X_4 and underestimated X_5 compared to Gartling.

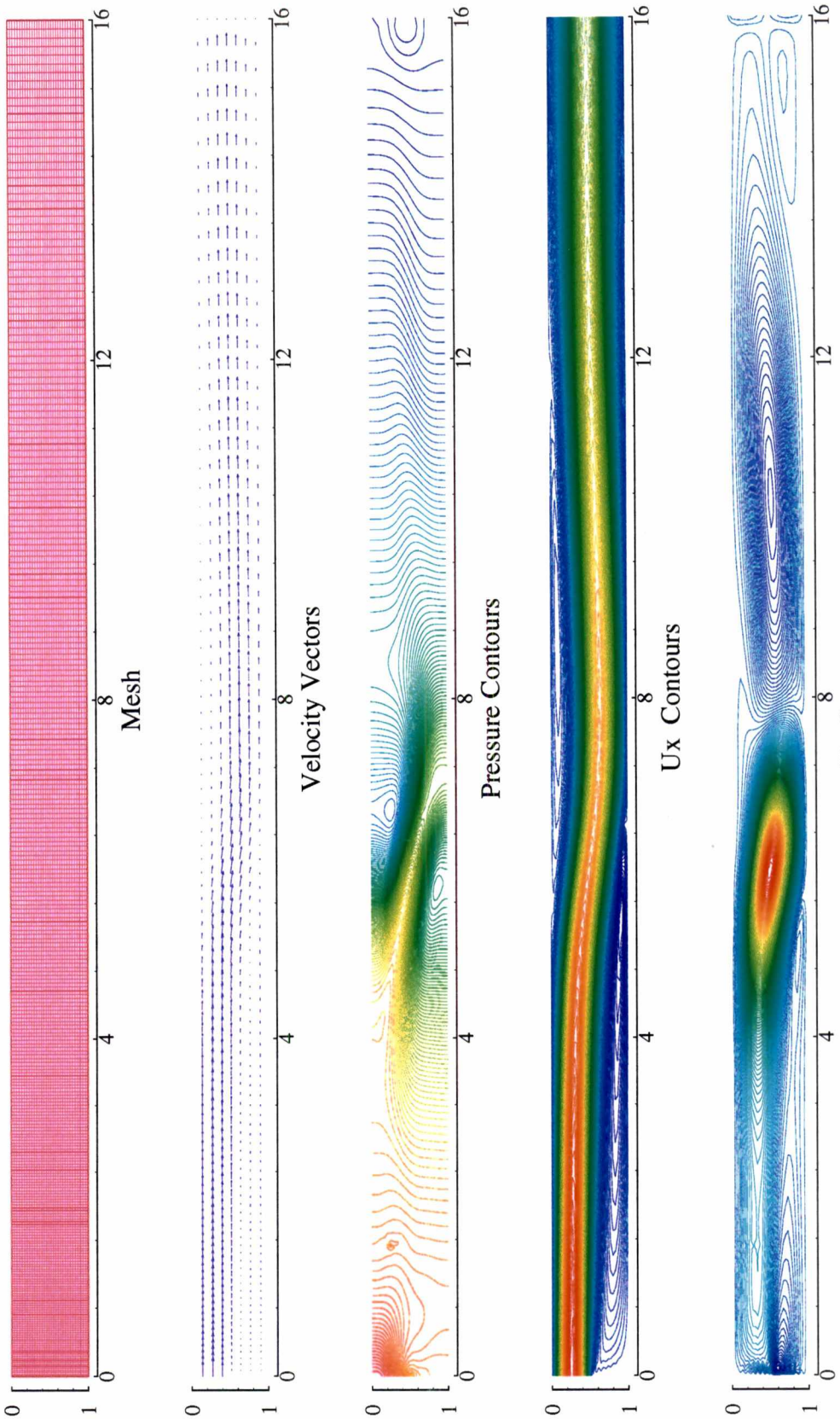

Figure 7.34: Backward-facing step, Re=1000, 31x300 mesh, β =0

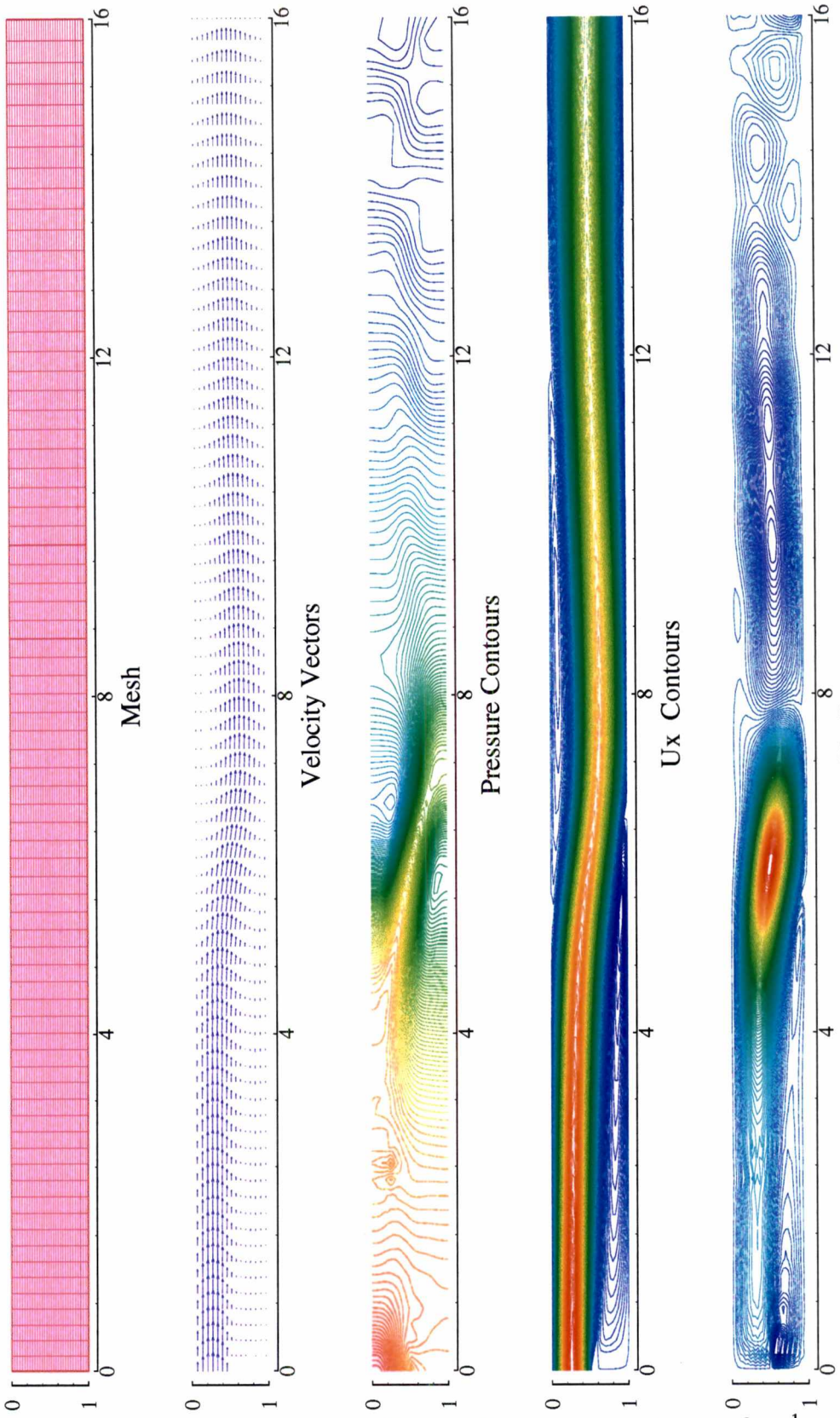

Figure 7.35: Backward-facing step, Re=1000, 31x74 mesh, $\beta = \frac{1}{2}$

Table 7.9: Summary of backward-facing step benchmark data

Re	mesh	algorithm	X_1	X_4	X_5	$\|\|q^h\|\|_E$	$\|\|\phi^h\|\|_E$	$\|\|p^h\|\|_E$
50	31x 50	GWS	02.05	00.00	00.00	1.198770	0.255e-6	1.431
400	31x200	GWS	08.53	00.00	00.00	0.226189	0.161e-8	0.107
400	31x 50	GWS	08.87	00.00	00.00	0.234427	0.155e-7	0.099
600	31x300	GWS	10.72	9.63	15.46	0.181487	0.994e-9	0.117
600	31x200	GWS	10.72	9.62	15.43	0.181454	0.118e-8	0.116
600	31x100	GWS	10.71	9.60	15.44	0.181697	0.277e-8	0.114
600	31x 50	GWS	11.04	10.00	16.04	0.185144	0.132e-7	0.106
800	31x300	GWS	12.18	10.44	20.23	0.159482	0.998e-9	0.170
800	31x200	GWS	12.17	10.42	20.21	0.159470	0.113e-8	0.170
800	31x100	GWS	12.37	10.63	20.48	0.160582	0.337e-8	0.175
800	31x 50	TWS-βx2.50	12.34	10.54	21.41	0.162214	0.928e-8	0.191
800	31x 50	TWS-βx0.25	12.35	10.55	20.69	0.161892	0.787e-8	0.175
800	31x 50	TWS-βx0.025	13.06	11.27	20.97	0.162007	0.923e-8	0.137
800	31x 50	SGMx0.30	12.41	10.74	18.85	0.161707	0.801e-8	0.221
800	31x 50	EPGx10.0	12.35	10.55	20.63	0.161893	0.787e-8	0.175
800	31x 50	EPGx4.00	12.35	10.55	20.64	0.161515	0.787e-8	0.175
800	31x 50	EPGx1.00	12.35	10.55	20.64	0.161516	0.787e-8	0.175
800	31x 50	EPGx0.10	12.34	10.54	21.36	0.162217	0.925e-8	0.190
800	31x 50	BPGx4.00	12.36	10.50	20.12	0.162009	0.991e-8	0.253
800	40x800	Gartling-2	12.20	9.70	20.96	n/a	n/a	n/a
800	20x200	Gartling-1	11.62	9.58	20.96	n/a	n/a	n/a
1000	31x300	GWS	13.42	11.35	24.46	0.145492	0.106e-8	0.231
1000	31x200	GWS	13.54	11.45	24.43	0.145463	0.140e-8	0.221
1000	31x 74	TWS-βx10.0	13.55	11.46	24.27	0.146264	0.320e-8	0.234
1000	31x 74	TWS-βx2.00	13.55	11.46	24.24	0.146621	0.320e-8	0.234
1000	31x 74	TWS-βx1.00	13.55	11.46	24.24	0.146621	0.320e-8	0.234
1000	31x 74	TWS-βx0.25	13.55	11.46	24.23	0.146621	0.320e-8	0.234
1000	31x 74	TWS-βx0.05	13.55	11.46	24.23	0.146621	0.320e-8	0.234
1000	31x 74	TWS-βx0.02	13.47	11.37	24.35	0.145921	0.335e-8	0.228
1000	31x 74	TWS-βx0.01	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	MPG-1x4.00	13.69	11.57	24.55	0.145941	0.471e-8	0.408
1000	31x 74	MPG-1x1.00	13.62	11.54	24.29	0.146551	0.321e-8	0.233
1000	31x 74	SGMx0.80	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	SGMx0.40	13.51	11.41	24.05	0.146665	0.328e-8	0.250
1000	31x 74	SGMx0.20	13.54	11.46	24.16	0.145875	0.342e-8	0.237
1000	31x 74	SGMx0.10	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	EPGx10.0	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	EPGx2.00	13.46	11.37	24.35	0.145921	0.335e-8	0.228
1000	31x 74	EPGx1.00	13.46	11.37	24.33	0.145922	0.335e-8	0.228
1000	31x 74	EPGx0.50	13.46	11.37	24.35	0.145921	0.335e-8	0.228
1000	31x 74	EPGx0.10	13.46	11.37	24.35	0.145921	0.335e-8	0.228
1000	31x 74	EPGx0.05	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	BPGx2.00	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	BPGx1.00	00.00	00.00	00.00	diverged	0.000e-8	0.000
1000	31x 74	BPGx0.25	00.00	00.00	00.00	diverged	0.000e-8	0.000

Similarly, the 31×300 Re = 1000 GWS solution was projected onto a 31×74 mesh using Tecplot® to provide an initial condition for testing the artificial diffusion methods. Each method except BPG was able to produce a stable solution. The best of the 31×74 coarse-grid artificial diffusion solutions over-predicted X_1 by 1% and the worst by 7%. Each artificial diffusion solution overestimated X_1 and X_4 and underestimated X_5 in comparison to Gartling. None of the methods were able to produce a stable solution at Re = 1000 with the 31×50 mesh.

Results of several simulations as primary re-attachment length are shown with literature results in Figure 7.36. The solid line indicates experimental data of Armaly et al. [6], which is somewhat approximate at small Reynolds number because of difficulty in interpolation from his figures. Filled boxes indicate the three-dimensional and filled circles the two-dimensional simulation results of Williams [86] using a TWS-β method. The three-dimensional and two-dimensional data begin to separate around Re = 400. The two-dimensional results of Williams show a constant re-attachment length from Re = 600 to Re = 800, indicating excess artificial diffusion, although the three-dimensional results show an excellent agreement with Armaly up to Re = 800.

Asterisks in Figure 7.36 indicate the two-dimensional non-diffusive fine-mesh results of Gartling [38] at Re = 800 using the commercial finite element code FIDAP. Plus-signs and X-marks indicate the results compiled by Freitas [37] using commercial codes on less fine meshes. Apparently the most accurate (in this case longest re-attachment length) of the commercial code results was obtained by the finite volume code FLUENT and the least accurate by the finite element code FLOTRAN. Freitas reported that FLUENT used a bounded QUICK scheme and FLOTRAN used a monotone SUPG scheme. Since FLOTRAN predicted an (erroneously) nearly constant re-attachment length for Re > 400, it appears to exhibit excess artificial diffusion, which limits the effective Reynolds number. Results from other commercial codes compiled by Freitas, not reported, were generally intermediate between FLUENT and FLOTRAN in results. FLUENT used 7401 nodes and FLOTRAN 6191 nodes, comparable with this study's 6633 nodes for GWS, and much finer than this study's 1683 nodes for artificial diffusion simulations at Re = 800.

The two-dimensional results of this study are indicated in Figure 7.36 with open symbols. As a meshing simplification, these simulations assumed fully developed flow directly at the step, hence no entrance channel exists and the initial downwards flow is compromised. This may lead to a over-prediction of the primary recirculation length, ap-

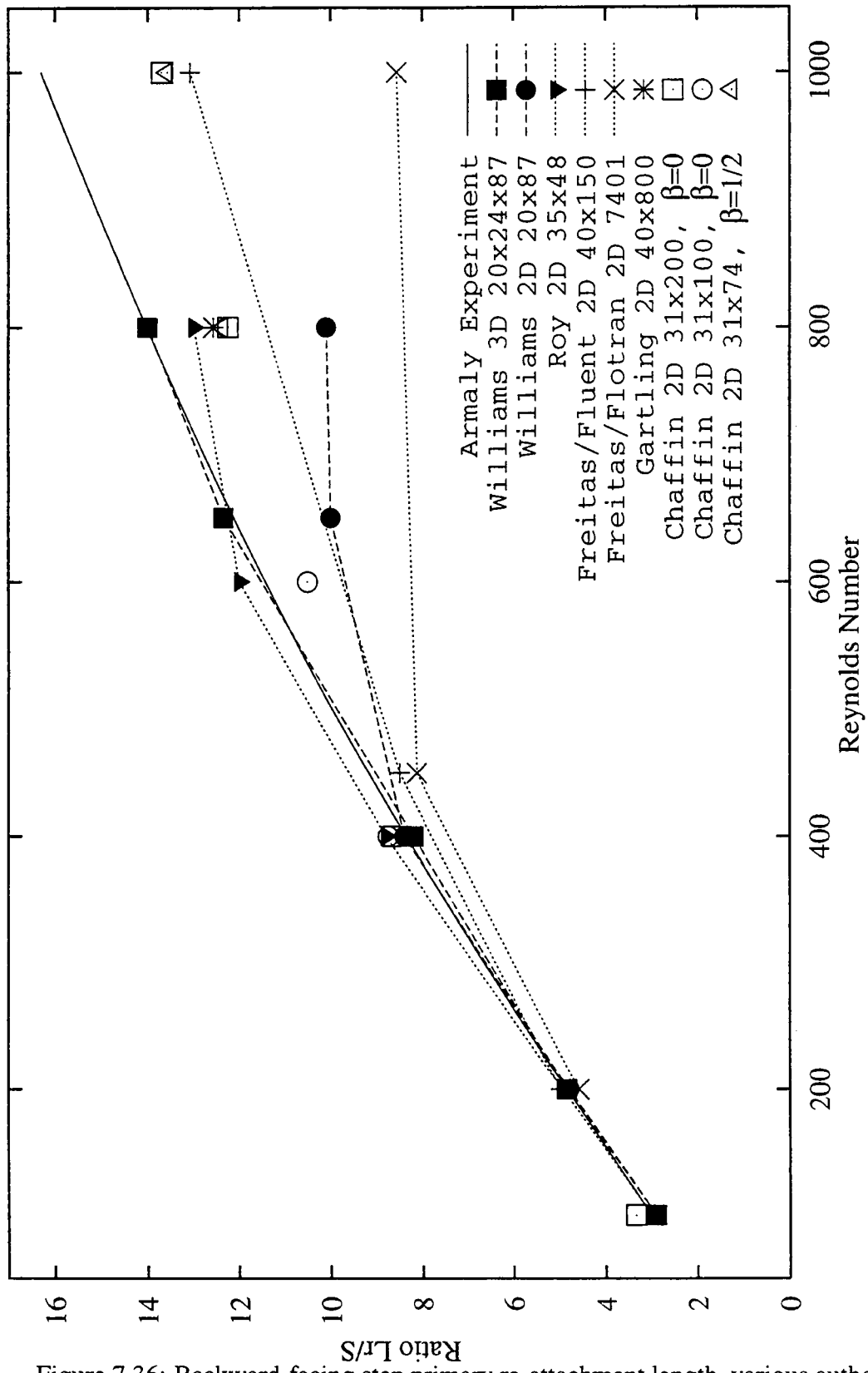

Figure 7.36: Backward-facing step primary re-attachment length, various authors

parently about 0.1 at Re = 50. Such an error would be insignificant at the higher Reynolds numbers simulated.

Comparing the various two-dimensional simulations at Re = 800, the artificial diffusion methods of this study produced re-attachment lengths among the largest, in contrast to the two-dimensional artificial diffusion results of Williams [86] and Freitas-FLOTRAN [37], which appear to show a constant length above Re = 600. The TWS-β method used for this study was essentially similar to that of Williams except for time scale (4.82). Simulations comparing the time scales of this study and of Williams revealed no significant difference for this problem statement. Given the excellent three-dimensional agreement of Williams also using artificial diffusion, the two-dimensional result at Re = 800 may be erroneous, although Williams attributed differences to dimensionality. Also shown in Figure 7.36 are the results of Roy and Baker [70] using one of the artificial diffusion methods (SGM) examined here, on a 35×48 mesh essentially similar to the present (31×50) coarse mesh. Roy and Baker's re-attachment are about about 5% longer than the coarse mesh artificial diffusion lengths of this study.

A graph of some selected distributions from the Re = 800 , 31×300 GWS simulation is shown in Figure 7.37. Profiles of u-velocity profiles at the rows of nodes adjacent to the walls indicate recirculation zones intercepts where crossing the x-axis. The recirculation zone boundaries of the Armaly et al.[6] experiment and the Gartling [38] simulation are indicated by arrows. The two velocity profiles in Figure 7.37, equidistant from opposite walls, become nearly coincident near the end of the domain, as required for the approach to fully developed duct exit flow, and the x-pressure distribution becomes nearly linear.

The comparison Re = 800 , 31×50 TWS-β simulation data is shown in Figure 7.38. Results are essentially indistinguishable to the finer-grid GWS simulation. In both simulations, as discussed, the primary lower recirculation length X_1 is quite accurate compared to Gartling, with the upper recirculation zone beginning X_4 slightly over-predicted, and its end X_5 slightly under-predicted.

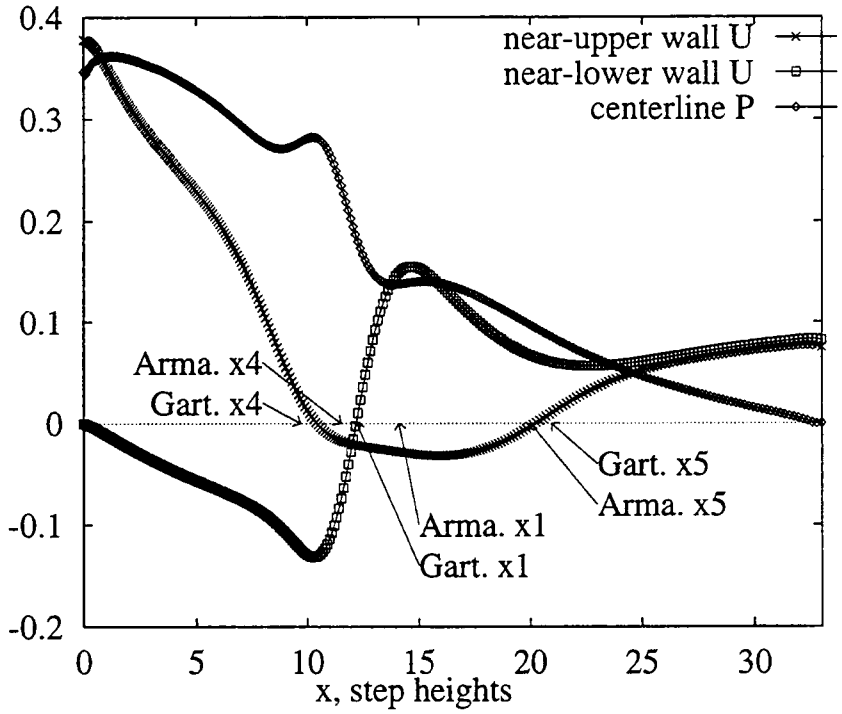

Figure 7.37: Backward-facing step velocity/pressure, Re=800, GWS 31×300

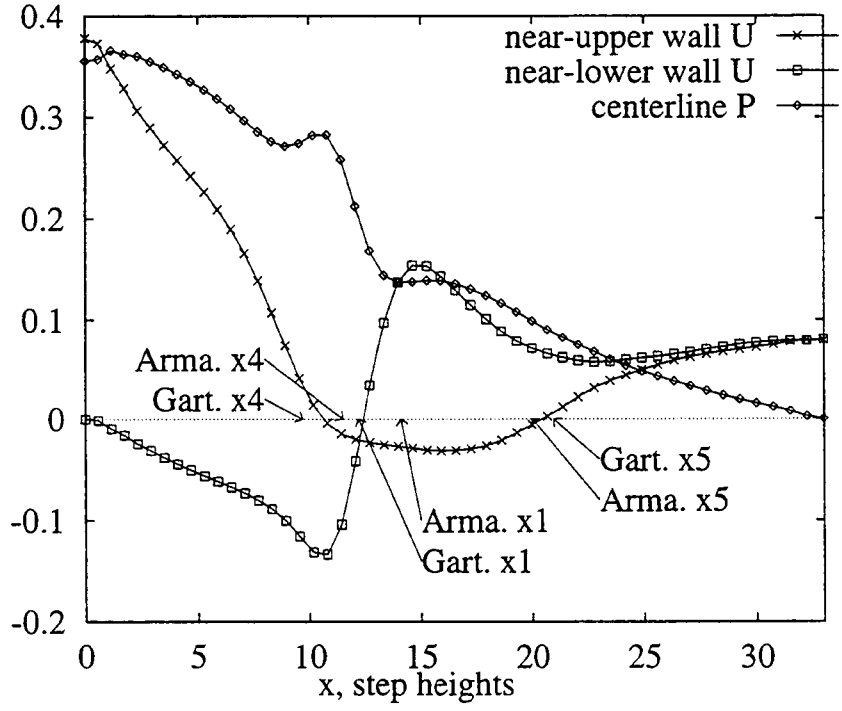

Figure 7.38: Backward-facing step velocity/pressure, Re=800, TWS-β 31×50

The comparison graph for Re = 1000 , 31×300 GWS simulation is shown in Figure 7.39. Armaly's 3D recirculation zone endpoints are indicated by arrows. As occurred for the Re = 800 simulations, X_1 and X_4 are under-predicted and X_5 is over-predicted; much of the difference due to three-dimensional effects. The Re = 1000 , 31×74 TWS-β simulation data is shown in Figure 7.40, with the Re = 1000 , 31×74 EPG simulation

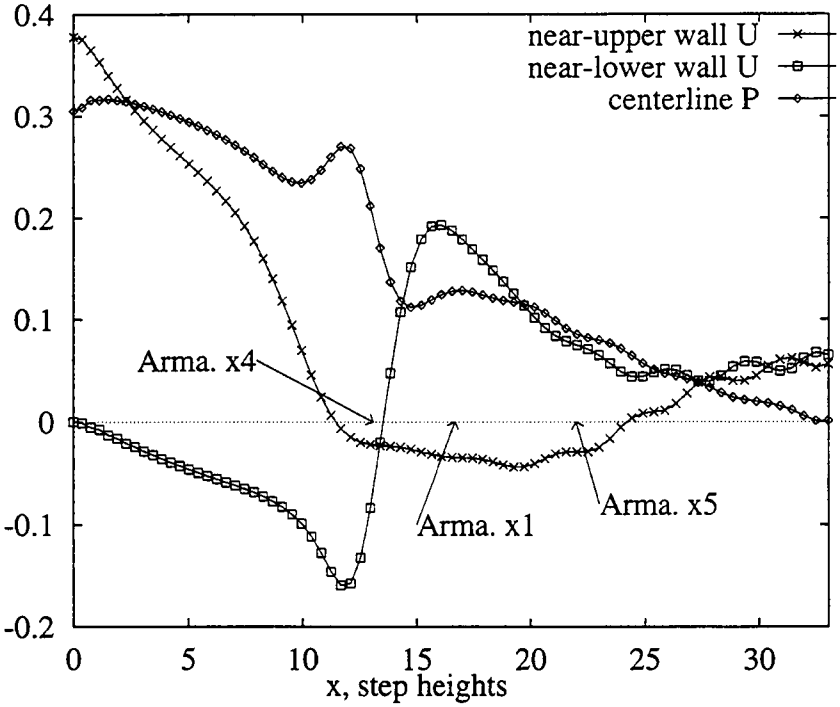

Figure 7.39: Backward-facing step velocity/pressure, Re=1000, TWS-β 31×74

data in Figure 7.41, each with artificial diffusion multiplier of unity.

The error control (as smoothness) of the EPG solution appears slightly better than that of the TWS-β solution. This is shown quantitatively in Table 7.8 as smaller velocity norm $||q^h||_E$ and pressure norm $||p^h||_E$. Comparing the recirculation zone boundaries, the EPG solution data are closer to the fine-mesh GWS solution than are the data of the TWS-β solution, hence the EPG solution is more accurate in that sense.

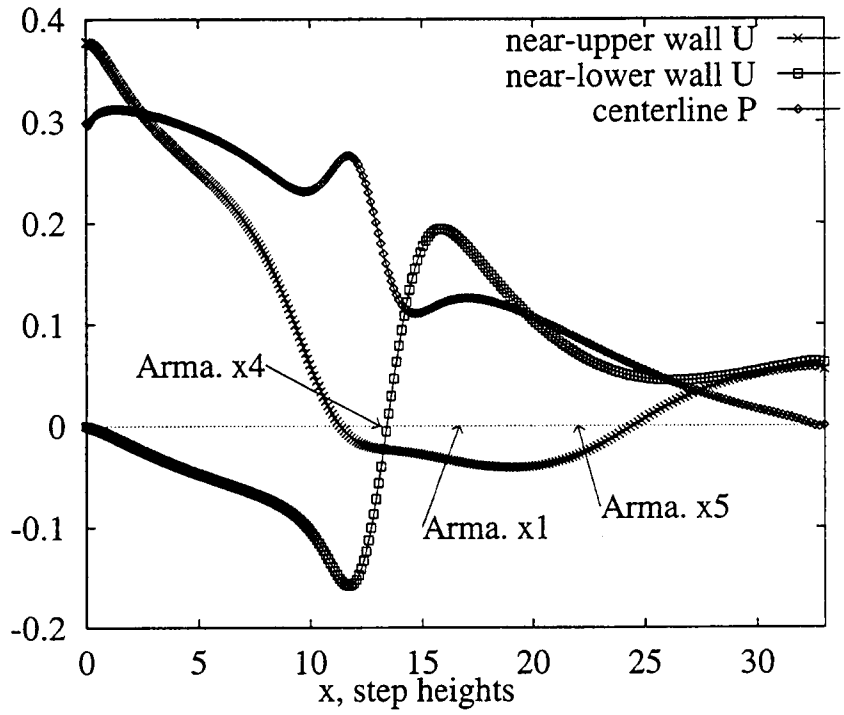

Figure 7.40: Backward-facing step velocity/pressure, Re=1000, GWS 31×300

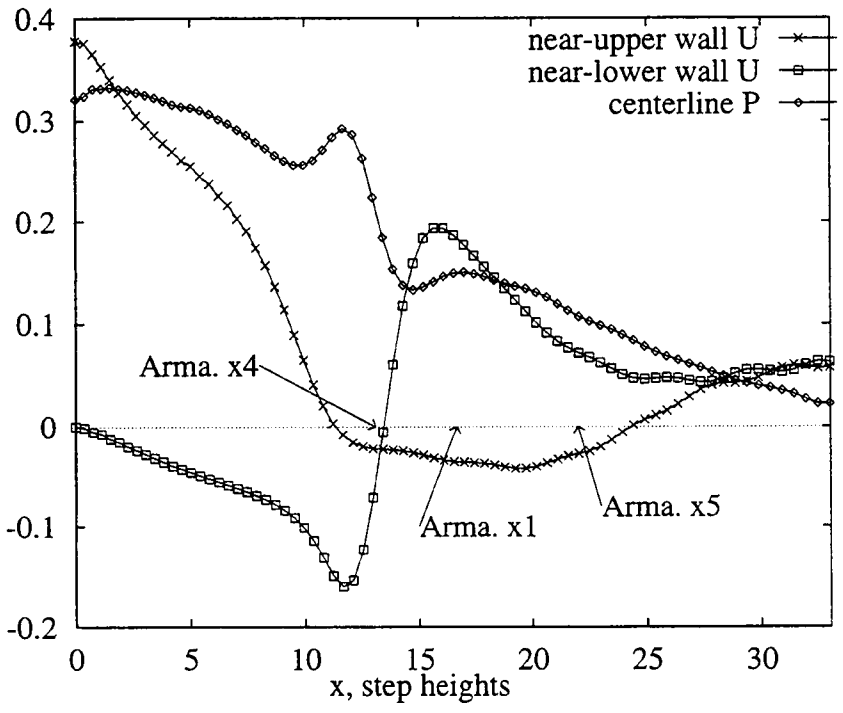

Figure 7.41: Backward-facing step velocity/pressure, Re=1000, EPG 31×74

The best of the Re = 1000 , 31 × 74 TWS-β simulations is summarized in Figure 7.42. Here a smaller artificial diffusion multiplier of 0.02 produces a smoother solution, which is somewhat counterintuitive. The numerical results in Table 7.8 show that this TWS-β simulation is essentially identical in recirculation lengths and norms to the four EPG simulations at this Reynolds number.

A desirable feature of an artificial diffusion method is robustness, i.e. achieving a reasonable solution without having to exactly guess an unknown parameter. The TWS and EPG methods appear to be the best of the tested methods in that measure. The TWS methods produced stable solutions using a range of parameter β from 0.02 to 10.0, while the EPG method produced stable solutions using a range of multiplier from 0.1 to 2.0. The TWS method functioned with a wider range of the parameter, but the EPG method produced a more accurate and constant solution over its range.

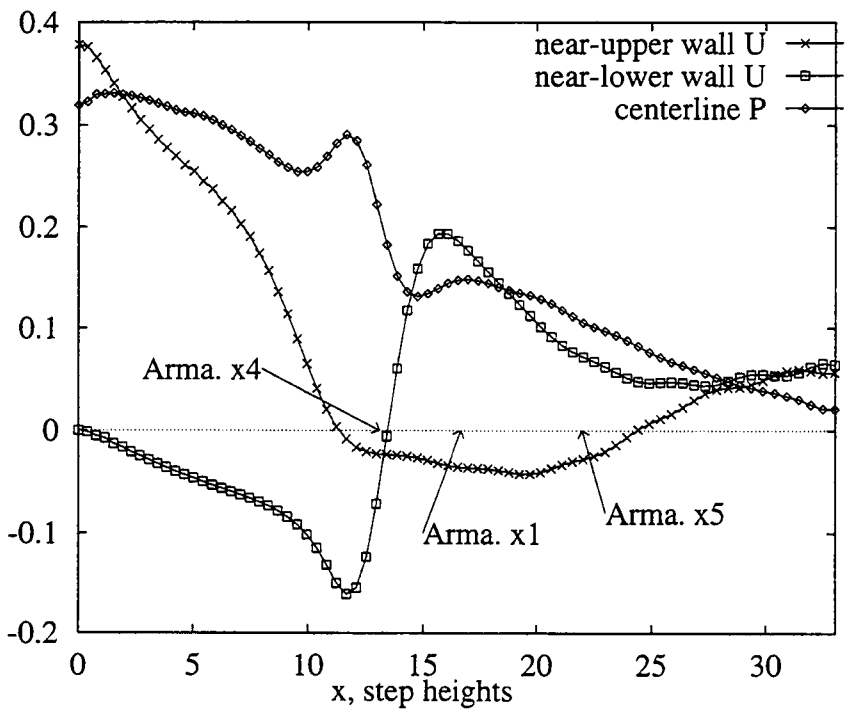

Figure 7.42: Backward-facing step velocity/pressure, Re=1000, TWS-β 31 × 74

7.3.5 Cylinder-in-crossflow benchmark simulation

A steady-state or transient two- or three-dimensional benchmark is a cylinder in crossflow in a duct [71], simulated in this study, or the qualitatively similar cylinder in a freestream [30]. The flow is steady at very low $\mathrm{Re}_D \approx 20$ and exhibits periodic vortex shedding at $\mathrm{Re}_D > 40$. The accuracy of the steady flow may be compared to literature values of the closure length of the vortex behind the cylinder. The accuracy of the periodic flow may be compared to literature values of vortex shedding frequency, nondimensionalized as Strouhal number $St = \frac{fD}{U}$. The geometry of the problem is shown in Figure 7.43, taken from Schäfer and Turek [71], except the clearance on the upper side of the cylinder (0.16 in [71], intended to generate nonzero lift) was modified for this study to 0.15 for simplified mesh generation.

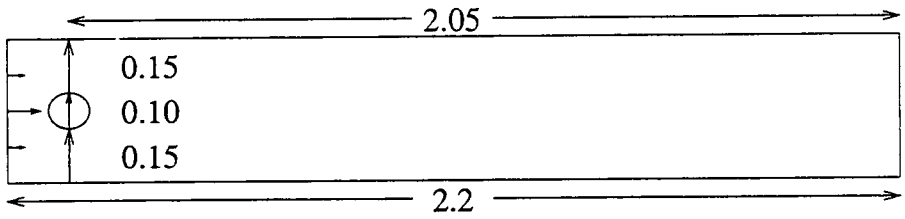

Figure 7.43: Geometry of cylinder in duct

A steady $\mathrm{Re}_D = 20$ simulation is shown in Figure 7.44 for a 56×299 GWS non-uniform mesh. Pressure oscillations are apparent near the cylinder in spite of the relatively fine mesh and small Reynolds number.

Simulated $\mathrm{Re} = 20$ GWS vortex closure lengths were 0.081 on a 56×299 non-uniform mesh, 0.079 on a 36×199 non-uniform mesh, and 0.077 on a 33×99 non-uniform mesh. Schäfer and Turek [71] obtained lengths of 0.066 to 0.078 on meshes of 42,000 to 2,660,000 degrees of freedom using an adaptive upwind FEM algorithm, with details of the upwind method unspecified. Degrees of freedom for the two meshes in this study were 10,000 to 51,500, a much coarser mesh than those of Schäfer and Turek. Other researchers in the compilation [71] used various upwind algorithms to get apparently grid-converged lengths of 0.073 and 0.085. Three researchers using non-diffusive FVM and FEM algorithms each obtained lengths of 0.085, thus the non-diffusive algorithms appeared to mostly, but not uniformly, produce longer vortex closure lengths.

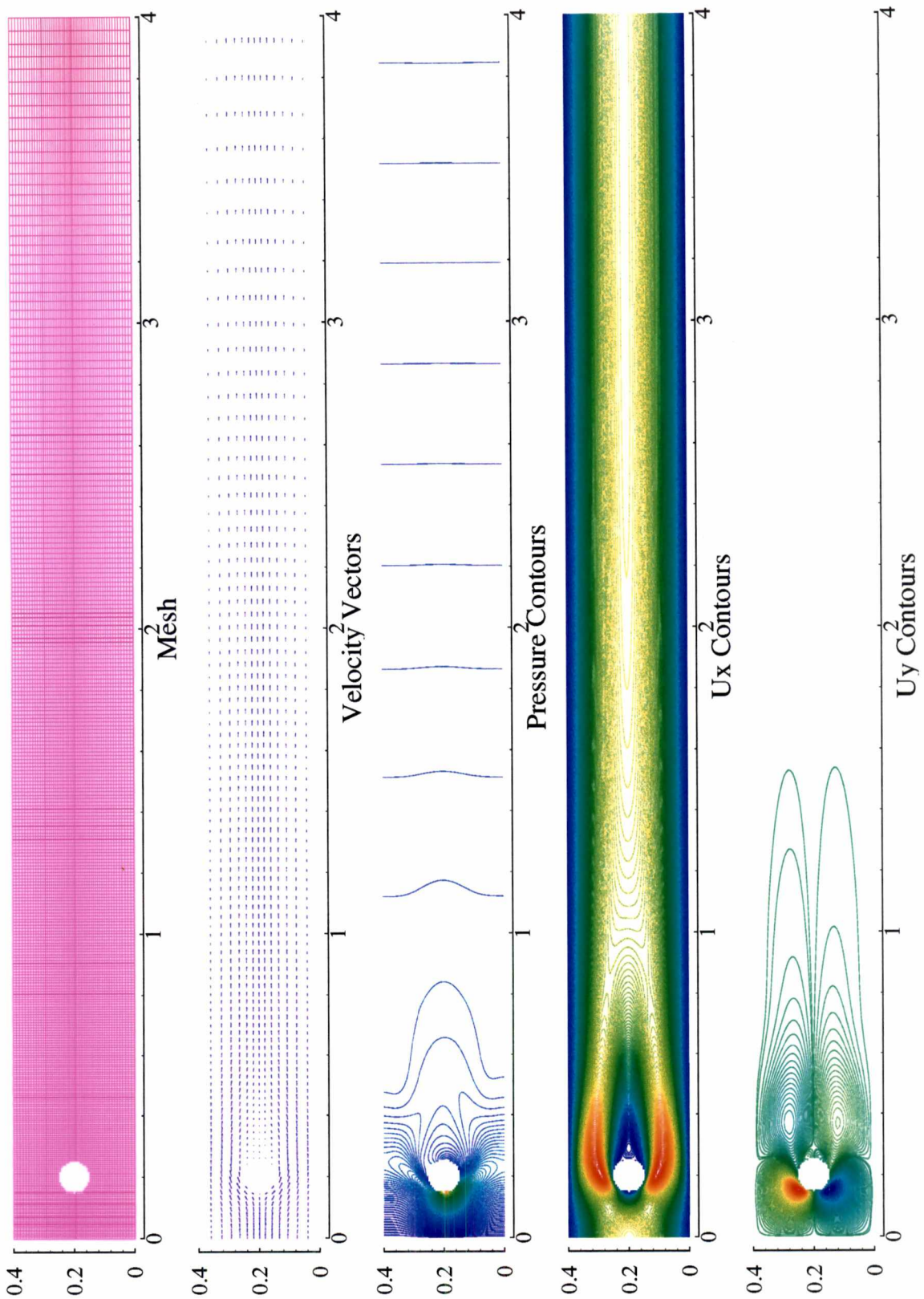

Figure 7.44: Cylinder in crossflow, Re = 20, 56×299 mesh

A snapshot of the unsteady $\mathrm{Re}_D = 100$ simulation data is shown in Figure 7.45 for a 56×299 GWS non-uniform mesh. The vortex street is evident. Oscillation period of the unsteady prediction was measured at the grid point on the centerline nearest $x = 0.475$, where the maximum values of transverse velocity were observed. The period was taken as the time between two sign changes of that nodal velocity, and Strouhal number was calculated using frequency as the inverse of the period with $D = 0.1$ and $\tilde{U} = 1$. The measured period was 0.3294 and Strouhal number 0.3036 for a GWS simulation on a 56×299 nonuniform mesh, The Strouhal number was 0.3147 for a GWS simulation on a 36×200 nonuniform mesh. As with the backward-facing step, the fine-mesh GWS solution was used to generate initial conditions for any artificial diffusion algorithms on a coarse mesh.

Results of the cylinder benchmark simulations are summarized in Table 7.10 as vortex closure length l_r at Re = 20, Strouhal number at Re = 100, and norms as before.

Table 7.10: Summary of cylinder-in-crossflow results

Re	mesh	algorithm	l_r	$\|\|q^h\|\|_E$	$\|\|\phi^h\|\|_E$	$\|\|p^h\|\|_E$
20	56x299	GWS	0.08082	0.8095e-2	0.80e-11	0.153e+0
20	36x199	GWS	0.07935	0.8064e-2	0.36e-10	0.142e+0
20	32x 99	GWS	0.07710	0.7937e-2	0.102e-9	0.139e+0
Re	mesh	algorithm	St	$\|\|q^h\|\|_E$	$\|\|\phi^h\|\|_E$	$\|\|p^h\|\|_E$
100	56x299	GWS	0.3036	0.2943e+0	0.627e-8	0.215e+3
100	36x199	GWS	0.3078	0.2908e+0	0.176e-7	0.233e+3
100	32x 99	TWS-βx10.0	0.2962	0.3005e+0	0.145e-7	0.244e+3
100	32x 99	TWS-βx1.00	0.2974	0.3009e+0	0.145e-7	0.244e+3
100	32x 99	TWS-βx10.0	0.2974	0.3009e+0	0.145e-7	0.244e+3
100	32x 99	MPG-2x0.001	0.2522	0.2904e+0	0.144e-7	0.204e+3
100	32x 99	MPG-2x0.0001	0.2522	0.2904e+0	0.144e-7	0.204e+3
100	32x 99	EPGx10.0	0.2974	0.3009e+0	0.145e-7	0.245e+3
100	32x 99	EPGx1.00	0.2977	0.3009e+0	0.145e-7	0.244e+3
100	32x 99	EPGx0.10	0.2976	0.3010e+0	0.145e-7	0.245e+3
100	32x 99	BPGx1.00	0.2977	0.3009e+0	0.145e-7	0.245e+3

Schäfer and Turek [71] obtained Strouhal numbers of 0.271 to 0.297 on meshes of 42,000 to 670,000 degrees of freedom. Other researchers in their compilation using diffusive algorithms reported St = 0.278, 0.298, and 0.299, and researchers using non-diffusive algorithms reported St = 0.300, 0.302, and 0.302. Although the different algorithms may not be directly comparable, each diffusive algorithm produced a lower

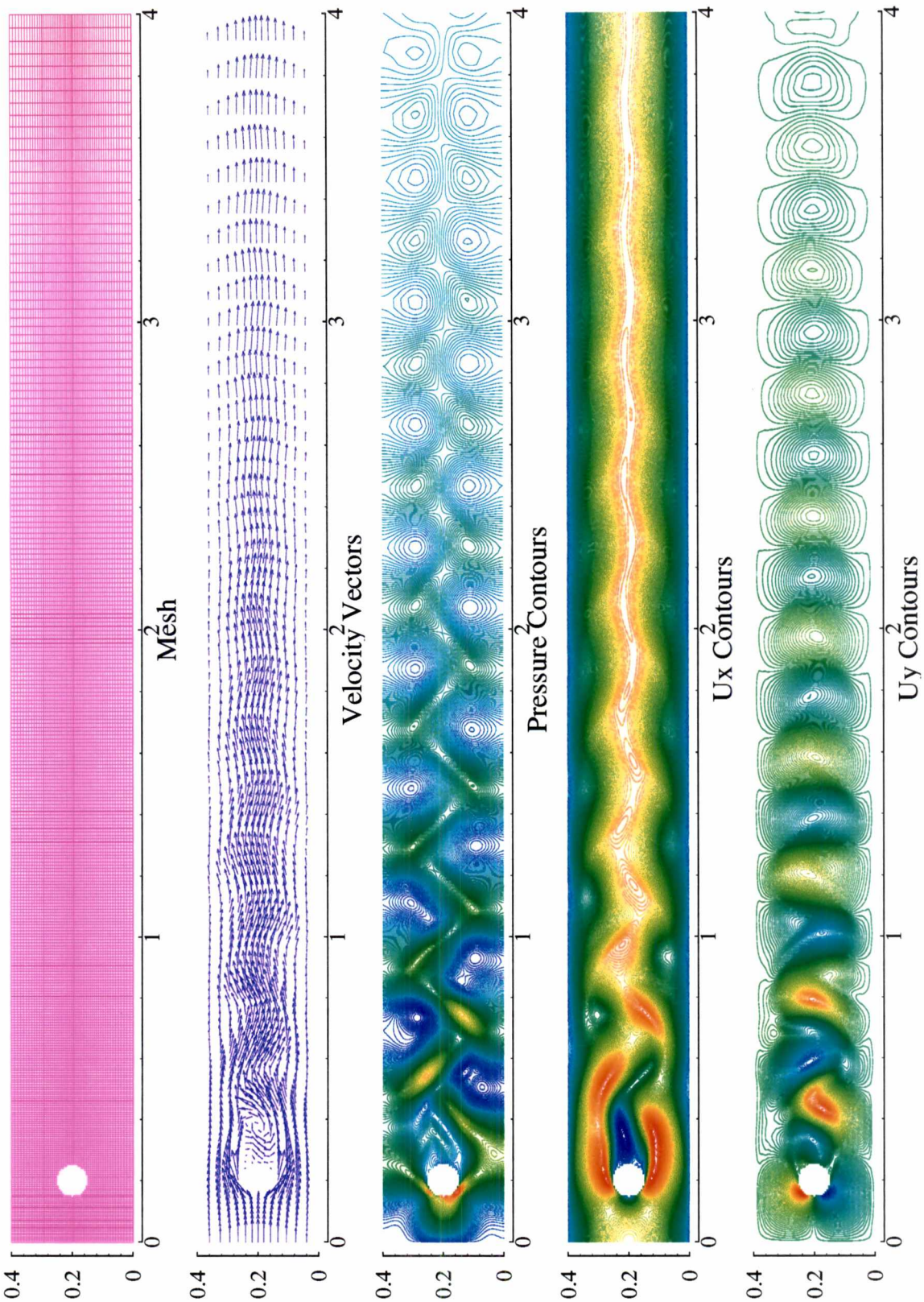

Figure 7.45: Cylinder in crossflow, Re = 100, 56×299 mesh

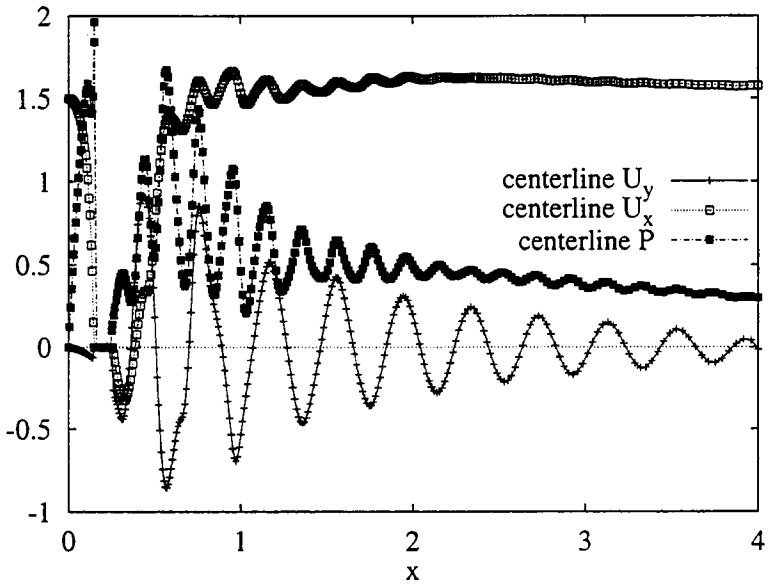

Figure 7.46: Cylinder in crossflow, Re = 100, 56×299 mesh

Strouhal number than each non-diffusive algorithm. Schäfer and Turek also reported an experimental Strouhal number of 0.287 on slightly different geometry, compared with a simulated St = 0.289 for that geometry.

Plots of centerline velocities and pressure are shown in Figure 7.46 for a 56×299 GWS solution and in Figure 7.47 for a 32×99 TWS-β solution, obviously less smooth but capable of resolving primary flow features.

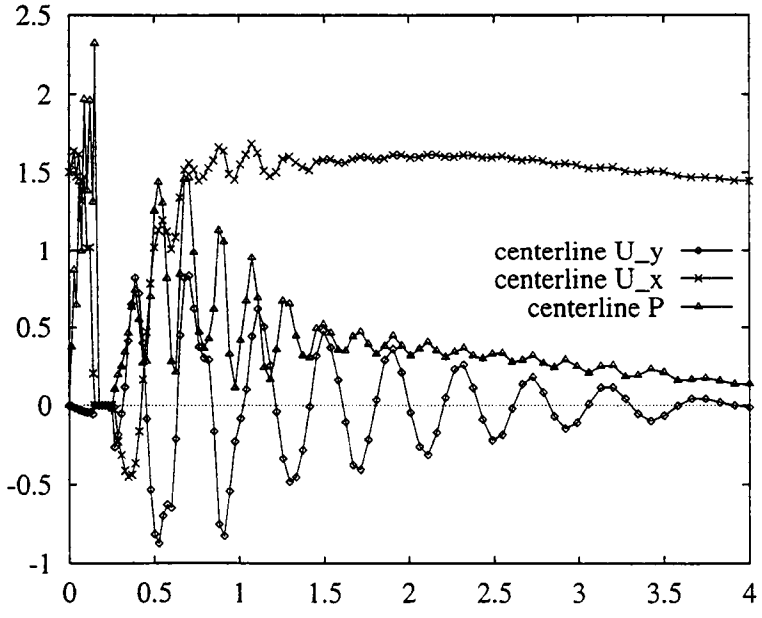

Figure 7.47: Cylinder in crossflow, Re = 100, TWS-β 32×99 mesh

Chapter 8

Summary and Conclusions

Taylor weak statement (TWS) finite element method approximations were formulated for the advection-diffusion equation, the Burgers equation, and the incompressible Navier-Stokes equations. A new TWS approximation was developed for the shallow-water equations. A Fourier analysis was performed to compare phase accuracy and stability for linear, quadratic, and cubic basis Galerkin and TWS one-dimensional finite element methods. New results include the analysis for quadratic and cubic basis TWS finite element methods, and a multiple-step linear basis TWS finite element method optimized for phase accuracy. A new modified equation analysis was developed in a general linear form, and the analysis was used to optimize two-dimensional TWS methods for phase accuracy and for stability.

Artificial diffusion TWS approximations for the incompressible Navier-Stokes equations were systematically compared for the first time to a wide variety of modified finite element methods, including SUPG, exponential Petrov-Galerkin, SGM, hierarchical basis, least-squares, and monotone methods. Each of the modified finite element methods was shown to be reducible to a Galerkin weak statement (GWS) method with some form of added artificial diffusion. The added artificial diffusion in some cases could be equated to a TWS correction form.

GWS, various finite difference and phase-accurate TWS algorithms were tested on verification solutions of the transient advection-diffusion equation. The phase-accurate TWS approximations were found to be effective in maintaining time accuracy for the advection-diffusion solutions, but that time-accuracy property was found to be of limited value on test problems for the Navier-Stokes equations.

GWS and artificial diffusion algorithms were tested on verification solutions of the boundary-value advection-diffusion equation and the viscous Burgers equation. Several of the artificial diffusion algorithms were found to be effective in controlling dispersive error.

GWS and artificial diffusion algorithms were tested on the thermal cavity incompressible Navier-Stokes benchmark problem. The artificial diffusion algorithms were found to have no effect on high-Rayleigh source-induced solution inaccuracy, apparently due to the relatively small induced velocities.

GWS and artificial diffusion algorithms were tested on a backward-facing step two-dimensional laminar incompressible Navier-Stokes benchmark problem at Reynolds numbers of 50 to 1000. A 31×300 GWS approximation at $Re = 800$ was found to predict a primary recirculation length within 0.2% of the 40×800 biquadratic GWS solution of Gartling [38]. The present GWS approximation predicted secondary recirculation lengths within 7% of Gartling. The exponential Petrov-Galerkin and TWS algorithms on a 31×74 mesh were the best of several artificial diffusion algorithms at $Re = 1000$. They were found to predict recirculation lengths within 1% of the 31×300 GWS solution at that Reynolds number. The SGM and monotone algorithms were found to be less accurate, and the bubble-function algorithm did not produce a stable solution on that mesh.

GWS and artificial diffusion algorithms were tested on a cylinder-in-crossflow two-dimensional laminar incompressible Navier-Stokes benchmark problem. A 56×299 GWS approximation at $Re = 100$ was found to predict vortex shedding frequency Strouhal number within 2% of the benchmark results of Schäfer and Turek [71]. Again the exponential Petrov-Galerkin and TWS algorithms on a 31×99 mesh produced the best results of several artificial diffusion algorithms at $Re = 100$. They predicted Strouhal numbers within 2% of the 56×299 GWS approximation at that Reynolds number. In addition, the exponential and TWS/SUPG algorithms were found to be relatively robust, in that accurate guesses were not required of the artificial diffusion multiplier. Again the SGM, monotone, and bubble-function methods were found less accurate.

In summary, the exponential Petrov-Galerkin and TWS/SUPG streamline artificial diffusion methods were found to be generally the most effective of the several alternative algorithms as determined for recirculating through-flow laminar incompressible Navier-Stokes test problems such as the backward-facing step or cylinder-in-crossflow.

Bibliography

[1] M. Ahues and M. Telias. Petrov-Galerkin schemes for the steady state convection-diffusion equation. In K. P. Holz, editor, *Finite elements in water resources : proceedings of the 4th international conference*. S ɔringer-Verlag, 1982.

[2] A. A. Aldama. *Filtering techniques for turbulent flow simulation*. Springer-Verlag, Berlin, 1990.

[3] M. B. Allen, I. Herrera, and G. F. Pinder. *Numerical modeling in science and engineering*. Wiley, New York, 1988.

[4] W. F. Ames. *Numerical methods for partial differential equations*. Academic Press, New York, 2nd edition, 1977.

[5] I. K. Argyros. *Theory and application of iteration methods*. Hemisphere, New York, 1993.

[6] B. F. Armaly, F. Durst, J. C. F. Pereira, and B. Schonung. Experimental and theoretical investigation of backward-facing step flow. *J. fluid mech.*, 172:473–496, 1983.

[7] X. Aubert and M. Deville. Compact differences in natural coordinates. *J. comp. phys.*, 49:505–516, 1983.

[8] A. J. Baker. *Finite element computational fluid dynamics*. Taylor & Francis, Washington, 1983.

[9] A. J. Baker and J. W. Kim. A Taylor weak-statement algorithm for hyperbolic conservation laws. *Int. J. numer. meth. fluids*, 7:489–520, 1987.

[10] E. T. Bouloutas and M. A. Celia. An improved cubic Petrov-Galerkin method for simulation of transient advection-diffusion processes in rectangularly decomposable domains. *Comput. meth. appl. mech. eng.*, 92:289–308, 1991.

[11] F. Brezzi. Mathematical theory of finite elements. In *State-of-the-art surveys of finite element technology*, New York, 1975. ASME.

[12] A. N. Brooks and T. J. R. Hughes. Streamline upwind Petrov-Galerkin methods for advection dominated flows. In *Proc. third int. conf. on finite element meth. in fluid flow*, Banff, Canada, 1980.

[13] M. E. Cantekin, J. J. Westerink, and R. A. Luettich Jr. Low and moderate Reynolds number transient flow simulations using space filtered Navier-Stokes equations. *Numer. meth. partial diff. eq.*, 10:491–524, 1994.

[14] G. F. Carey and J. T. Oden. *Finite elements-an introduction*, volume 1. Prentice-Hall, 1981.

[15] G. F. Carey and J. T. Oden. *Finite elements-computational aspects*, volume 3. Prentice-Hall, 1984.

[16] B. Cathers and B. A. O'Connor. The group velocity of some numerical schemes. *Int. J. numer. meth. fluids*, 5:201–224, 1985.

[17] D. J. Chaffin and A. J. Baker. On Taylor weak statement finite element methods for computational fluid dynamics. *Int. J. numer. meth. fluids*, 21:273–294, 1995.

[18] S. K. Chee and J. H. Kwon. An upwind finite element method using optimal weighting functions for solving compressible Euler equations. *Comput. fluid dynamics J.*, 4:293–301, 1995.

[19] R. C. Y. Chin, G. W. Hedstrom, and K. E. Karlsson. A simplified Galerkin method for hyperbolic equations. *Math. comp.*, 33:647–658, 1979.

[20] A.J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. comp.*, 23:341–353, 1969.

[21] I. Christie, D. F. Griffiths, A. R. Mitchell, and O. C. Zienkiewicz. Finite element methods for second order differential equations with significant first derivative. *Int. J. numer. meth. engrg.*, 10:1389–1396, 1976.

[22] I. Christie and A. R. Mitchell. Upwinding of high order Galerkin methods in conduction-convection problems. *Int. J. numer. meth. engrg.*, 12:1764–1771, 1978.

[23] G. Comini, G. Cortella, and M. Manzan. A streamfunction-vorticity-based finite-element formulation for laminar-convection problems. *Numer. heat transfer Part B*, 28:1–22, 1995.

[24] R. Courant, E. Isaacson, and M. Rees. On the solution of nonlinear hyperbolic differential equations. *Comm. on pure and appl. math.*, 5:243–255, 1952.

[25] G. de Vahl Davis. Natural convection of air in a square cavity. *Int. J. numer. meth. fluids*, 3:249–264, 1983.

[26] E. Dick. Accurate Petrov-Galerkin methods for transient convective diffusion problems. *Int. J. numer. meth. engrg.*, 19:1425–1433, 1983.

[27] R. A. Dobbins. *Atmospheric motion and air pollution: an introduction for students.* Wiley, New York, 1979.

[28] J. Donea. A Taylor-Galerkin method for convective transport problems. *Int. J. numer. meth. engrg.*, 20:101–119, 1984.

[29] J. Donea, L. Quartapelle, and V. Selmin. An analysis of time discretization in the finite element solution of hyperbolic problems. *J. comp. phys.*, 70:463–499, 1987.

[30] M. S. Engelman and M.-A. Jamnia. Transient flow past a circular cylinder: a benchmark solution. *Int. J. numer. meth. fluids*, 11:985–1000, 1990.

[31] J. H. Ferziger. *Numerical methods for engineering application.* Wiley, New York, 1981.

[32] J. H. Ferziger and M. Perić. *Computational methods for fluid dynamics.* Springer, New York, 1996.

[33] B. A. Finlayson. *The method of weighted residuals and variational principles, with application in fluid mechanics, heat and mass transfer.* Academic Press, New York, 1972.

[34] C. A. J. Fletcher. *Computational techniques for fluid dynamics.* Springer-Verlag, 2nd edition, 1991.

[35] A. S. Franca and K. Haghighi. Error estimation and adaptivity in finite element analysis of convective heat transfer problems, part I: theoretical considerations. *Numer. heat transfer Part B*, 29:479–490, 1996.

[36] J. Freels. *A Taylor weak statement finite element algorithm for real-gas compressible Navier-Stokes simulation.* Ph.D. dissertation, University of Tennessee at Knoxville, 1992.

[37] C. J. Freitas. Perspective: selected benchmarks from commercial CFD codes. *Trans. ASME Fluids Engrg. Div.*, 117:208–218, 1995.

[38] D. K. Gartling. A test problem for outflow boundary conditions-flow over a backward-facing step. *Int. J. numer. meth. fluids*, 11:953–967, 1990.

[39] U. Ghia, K. N. Ghia, and C. T. Shin. High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *J. comp. phys.*, 48:387–411, 1982.

[40] M. A. Gill. Dam break problem. In N. P. Cheremisinoff, editor, *Encyclopedia of fluid mechanics*, chapter 36, pages 1429–1473. Gulf Pub. Co., 1986.

[41] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins University Press, 2nd edition, 1989.

[42] W. G. Gray and G. F. Pinder. An analysis of the numerical solution of the transport equation. *Water resources research*, 12:547–555, 1976.

[43] P. M. Gresho. Incompressible fluid dynamics: some fundamental formulation issues. *Ann. rev. fluid mech.*, 23:649–686, 1991.

[44] P. M. Gresho and R. L. Lee. Comments on 'the group velocity of some numerical schemes'. *Int. J. numer. meth. fluids*, 7:1357–1362, 1987.

[45] P. M. Gresho, R. L. Lee, and R. L. Sani. Advection-dominated flows, with emphasis on the consequences of mass lumping. In R. H. Gallagher et al., editor, *Finite elements in fluids-Volume 3*, New York, 1978. Wiley.

[46] A. Harten and H. Tal-Ezer. On a fourth order accurate implicit finite difference scheme for hyperbolic conservation laws: I. nonstiff strongly dynamic problems. *Math. comp.*, 36:353–373, 1981.

[47] T. S. Heines and L. K. Peters. The effect of a horizontal impervious layer caused by a temperature inversion aloft on the dispersion of pollutants in the atmosphere. *Atmos. environment*, 7:39–48, 1973.

[48] C. Hirsch. *Numerical computation of internal and external flows*, volume 1. Wiley, New York, 1988.

[49] J. S. Iannelli. NEWS: a non-linear element-upstream weak statement for the Euler and Navier-Stokes equations. In *Proceedings of the 34th AIAA aerospace sciences meeting*, Reno, NV, January 1996.

[50] A. Iserles and G. Strang. The optimal accuracy of difference schemes. *Trans. Amer. Math. Soc*, 277:779–803, 1983.

[51] Bo-Nan Jiang and G. F. Carey. Least-square finite element methods for compressible Euler equations. *Int. J. numer. meth. fluids*, 10:557–568, 1990.

[52] N. Kondo, N. Tosaka, and T. Nishimura. Computation of the incompressible viscous flows by the third-order upwind finite element method. *Int. J. numer. meth. fluids*, 15:1013–1024, 1992.

[53] P. Lax and B. Wendroff. Systems of conservation laws. *Comm. pure appl. math.*, 8:217–237, 1960.

[54] W. Layton. On the principal axes of diffusion in difference schemes for 2d transport problems. *J. comp. phys.*, 90:336–347, 1983.

[55] W. Layton. Optimal difference schemes for 2d transport problems. *J. comp. appl. math.*, 45:337–341, 1993.

[56] A. Leonard. Energy cascade in large-eddy simulations. In *Advances in geophysics 18*, pages 244–247. Academic Press, New York, 1994.

[57] B. P. Leonard. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Comput. meth. appl. mech. eng.*, 19:59–98, 1979.

[58] B. P. Leonard and S. Mokhtari. ULTRA-SHARP solution of the Smith-Hutton problem. Technical memorandum 105435, NASA Lewis Research Center, Cleveland, Ohio, 1992.

[59] W. Noronha. *Accuracy, convergence, and stability of a finite element algorithm for incompressible flow*. Ph.D. dissertation, University of Tennessee at Knoxville, 1989.

[60] E. S. Oran and J. P. Boris. *Numerical simulation of reactive flow*. Elsevier, New York, 1987.

[61] S. V. Patankar. *Numerical heat transfer and fluid flow*. Hemisphere, Washington, 1980.

[62] W. H. Press. *Numerical recipes in FORTRAN : the art of scientific computing*. Cambridge, New York, 2nd edition, 1992.

[63] P. Le Quéré. Accurate solutions to the square thermally driven cavity at high Rayleigh number. *Computers & fluids*, 20:29–41, 1991.

[64] R. Rannacher. Finite element solution of diffusion problems with irregular data. *Numerische mathematik*, 43:309–327, 1984.

[65] W. H. Raymond and A. Garder. Selective damping in a Galerkin method for solving wave problems with variable grids. *Monthly weather review*, 104:1583–1590, 1976.

[66] J. N. Reddy and A. Satake. A comparison of a penalty finite element model with the stream function-vorticity model of natural convection in enclosures. *Trans. ASME Heat Transfer Div.*, 102:659–666, 1980.

[67] P. J. Roache. *Computational fluid dynamics*. Hermosa, Albuquerqe, NM, 1972.

[68] S. Roy. *On improved methods for monotone CFD solution accuracy*. Ph.D. dissertation, University of Tennessee at Knoxville, 1994.

[69] S. Roy and A. J. Baker. A non-linear sub-grid embedded finite element basis for accurate monotone steady CFD solutions. *Numer. heat transfer Part B*, 31:135–176, 1997.

[70] S. Roy and A. J. Baker. A non-linear sub-grid embedded finite element basis for accurate monotone steady CFD solutions-Part II:benchmark Navier-Stokes solutions, to be published in. *Numer. heat transfer Part B*, 1997.

[71] M. Schäfer and S. Turek. Benchmark computations of laminar flow around a cylinder. In E. H. Hirschel, editor, *Flow simulation with high-performance computers II*, pages 547–556. Friedr. Vieweg, 1996.

[72] T. W. H. Sheu, S. F. Tsai, and M. M. T. Wang. A monotone multidimensional upwind finite element method for advection-diffusion problems. *Numer. heat transfer Part B*, 29:325–344, 1996.

[73] H. L. Stone and P. L. T. Brian. Numerical solution of convective transport problems. *Amer. Inst. Chem. Engrs. J.*, 9:681–688, 1964.

[74] C. R. Swaminathan and V. R. Voller. Streamline upwind scheme for control-volume finite elements, Part 1. Formulations. *Numer. heat transfer Part B*, 22:95–123, 1992.

[75] T.E. Tezduyar and D.K. Ganjoo. Petrov-Galerkin forms with weight functions depending on spatial and temporal discretization. *Comput. meth. appl. mech. eng.*, 59:49–71, 1986.

[76] V. Thomee. Finite difference methods. In P. G. Ciarlet and J. L. Lions, editors, *Handbook of numerical analysis*, volume 1. North-Holland, 1990.

[77] M. Th. van Genuchten and W. G. Gray. Analysis of some dispersion corrected numerical schemes for solution of the transport equation. *Int. J. numer. meth. engrg.*, 12:387–404, 1978.

[78] R. S. Varga. *Matrix iterative analysis.* Prentice-Hall, Englewood Cliffs, N.J., 1962.

[79] R. Vichnevetsky and F. De Schutter. A frequency analysis of finite difference and finite-element methods for initial value problems. In R. Vichnevetsky, editor, *Advances in computer methods for partial differential equations*, pages 46–52, Ghent, Belgium, 1975. AICA.

[80] J. von Neumann and R. D. Richtmeyer. A method for the numerical calculation of hydrodynamic shocks. *J. appl. phys.*, 21:232–237, 1950.

[81] R. Wait. Conforming methods for self-adjoint elliptic problems. In *State-of-the-art surveys of finite element technology*, New York, 1975. ASME.

[82] R. Wait and A. R. Mitchell. *Finite element analysis and applications.* Wiley, New York, 1985.

[83] R. F. Warming and B. J. Hyett. The modified equation approach to the stability and accuracy analysis of finite difference methods. *J. comp. phys.*, 14:159–179, 1974.

[84] J. J. Westerink and D. Shea. Consistent higher degree Petrov-Galerkin methods for the solution of the transient convection-diffusion equation. *Int. J. numer. meth. engrg.*, 28:1077–1101, 1989.

[85] F. M. White. *Viscous fluid flow.* McGraw-Hill, New York, 1974.

[86] P. T. Williams. *A three-dimensional, time-accurate, incompressible Navier-Stokes finite element computational fluid dynamics algorithm.* Ph.D. dissertation, University of Tennessee at Knoxville, 1993.

[87] J. Zhang. *A selective Taylor weak-statement based, quadrature-free finite element algorithm for aerodynamic flow.* Ph.D. dissertation, University of Tennessee at Knoxville, 1995.

[88] O. C. Zienkiewicz and C. Taylor. *The finite element method*, volume 1. McGraw-Hill, New York, 4th edition, 1989.

Appendices

Appendix I. Macsyma program to generate 1D element matrices

```
\*linear basis*\
(c1) emt : zeromatrix(2,1)$
(c2) emt[1,1] : 1-eta$ (c3) emt[2,1] : eta$ (c4) emt$
(c5) dnde : zeromatrix(2,1)$ (c6) dnde : diff(emt,eta)$
(c7) m00 : zeromatrix(2,2)$ (c8) for k thru 2 do
(for l thru 2 do m00[k,l] : emt[k,1]*emt[l,1]);
(c9) integrate(m00,eta,0,1)$ (c10) i00 : expand(6*%);
           [ 2 1 ]
(d10)      [     ]
           [ 1 2 ]
(c11) m01 : zeromatrix(2,2)$ (c12) for k thru 2 do
(for l thru 2 do m01[k,l] : emt[k,1]*dnde[l,1]);
(c13) integrate(m01,eta,0,1)$ (c14) i01 : expand(2*%);
          [ -1  1 ]
(d14)     [       ]
          [ -1  1 ]
(c15) m11 : zeromatrix(2,2)$ (c16) for k thru 2 do
(for l thru 2 do m11[k,l] : dnde[k,1]*dnde[l,1]);
(c17) integrate(m11,eta,0,1)$ (c18) i11 : expand(%);
          [  1 -1  ]
(d18)     [        ]
          [ -1  1  ]
(c19) m001 : zeromatrix(2,2)$ (c20) for k thru 2 do
(for l thru 2 do m001[k,l] : emt*emt[k,1]*dnde[l,1]);
(c21) integrate(m001,eta,0,1)$ (c22) i001 : expand(6*%);
         [ [ -2 ]   [ 2 ] ]
         [ [ -1 ]   [ 1 ] ]
(d22)    [                ]
         [ [ -1 ]   [ 1 ] ]
         [ [ -2 ]   [ 2 ] ]
(c23) m011 : zeromatrix(2,2)$ (c24) for k thru 2 do
(for l thru 2 do m011[k,l] : emt*dnde[k,1]*dnde[l,1]);
(c25) integrate(m011,eta,0,1)$ (c26) i011 : expand(2*%);
        [ [ 1 ]   [ -1 ] ]
        [ [ 1 ]   [ -1 ] ]
(d26)   [                ]
        [ [ -1 ]   [ 1 ] ]
        [ [ -1 ]   [ 1 ] ]
\*quadratic basis*\
(c76) emt : zeromatrix(3,1)$
(c77) emt[1,1] : (1-eta)*(1-2*eta)$
(c78) emt[2,1] : (1-eta)*(4*eta)$
(c79) emt[3,1] : eta*(2*eta-1)$
(c81) eta : 0; (c82) ev(emt,eval);
          [ 1 ]
(d82)     [ 0 ]
          [ 0 ]
(c83) eta : 1; (c84) ev(emt,eval);
          [ 0 ]
(d84)     [ 0 ]
          [ 1 ]
```

```
(c85) eta : 1/2; (c86) ev(emt,eval);
          [ 0 ]
(d86)     [ 1 ]
          [ 0 ]
(c87) remvalue(eta);
(c88) dnde : zeromatrix(3,1)$ (c89)
dnde : diff(emt,eta);
(c90) m00 : zeromatrix(3,3)$ (c92) for k thru 3 do
(for l thru 3 do m00[k,l] : emt[k,1]*emt[l,1]);
(c93) integrate(m00,eta,0,1)$ (c94) i00 : factor(%);
(c95) 30*i00;
            [  4    2    -1  ]
            [  2    16    2  ]
            [ -1    2     4  ]
(c91) m01 : zeromatrix(3,3)$ (c96) for k thru 3 do
(for l thru 3 do m01[k,l] : emt[k,1]*dnde[l,1]);
(c97) integrate(m01,eta,0,1)$ (c98) i01 : factor(%);
(c99) factor(6*%); (c100) expand(%);
            [ -3   4   -1  ]
(d100)      [ -4   0    4  ]
            [  1   -4   3  ]
(c101) m11 : zeromatrix(3,3)$ (c102) for k thru 3 do
(for l thru 3 do m11[k,l] : dnde[k,1]*dnde[l,1]);
(c103) integrate(m11,eta,0,1)$ (c104) i11 : factor(6*%);
(c105) expand(%); (c106) expand(%/2);
            [  7   -8   1  ]
(d106)      [ -8  16   -8  ]
            [  1   -8   7  ]
(c107) m001 : zeromatrix(3,3)$ (c108) for k thru 3 do
(for l thru 3 do m001[k,l] : emt*emt[k,1]*dnde[l,1]);
(c111) integrate(m001,eta,0,1)$ (c112) expand(30*%);
           [ [ -10 ]   [ 12 ]    [ -2 ] ]
           [ [ -6  ]   [ 8  ]    [ -2 ] ]
           [ [  1  ]   [ 0  ]    [ -1 ] ]
           [                            ]
           [ [ -6  ]   [  8 ]    [ -2 ] ]
(d112)     [ [ -16 ]   [  0 ]    [ 16 ] ]
           [ [  2  ]   [ -8 ]    [  6 ] ]
           [                            ]
           [   [ 1   ] [  0  ]   [ -1 ] ]
           [   [ 2   ] [ -8  ]   [  6 ] ]
           [   [ 2   ] [ -12 ]   [ 10 ] ]
(c113) m011 : zeromatrix(3,3)$ (c114) for k thru 3 do
(for l thru 3 do m011[k,l] : emt*dnde[k,1]*dnde[l,1]);
(c115) integrate(m011,eta,0,1)$ (c117) expand(30*%);
           [ [ 37 ]    [ -44 ]   [  7 ] ]
           [ [ 36 ]    [ -32 ]   [ -4 ] ]
           [ [ -3 ]    [ -4  ]   [  7 ] ]
           [                            ]
           [ [ -44]    [ 48 ]    [ -4 ] ]
(d117)     [ [ -32]    [ 64 ]    [ -32] ]
           [ [ -4 ]    [ 48 ]    [ -44] ]
           [                            ]
           [ [  7 ]    [ -4  ]   [ -3 ] ]
           [ [ -4 ]    [ -32 ]   [ 36 ] ]
           [ [  7 ]    [ -44 ]   [ 37 ] ]
```

```
\*cubic basis*\
(c143) remvalue(eta);
(c144) emt[1,1] : -(eta-1)*(3*eta-2)*(3*eta-1)/2$
(c145) emt[2,1] : 9*(eta-1)*(3*eta-2)*eta/2$
(c146) emt[3,1] : -9*(eta-1)*(3*eta-1)*eta/2$
(c147) emt[4,1] : (3*eta-1)*(3*eta-2)*eta/2$
(c148) emt; (c149) eta : 0$ (c150) ev(emt,eval);
         [ 1 ]
         [ 0 ]
         [ 0 ]
         [ 0 ]
(c151) eta : 1/3$ (c152) ev(emt,eval);
         [ 0 ]
         [ 1 ]
         [ 0 ]
         [ 0 ]
(c153) eta : 2/3$ (c154) ev(emt,eval);
         [ 0 ]
         [ 0 ]
         [ 1 ]
         [ 0 ]
(c155) eta : 1$ (c156) ev(emt,eval);
         [ 0 ]
         [ 0 ]
         [ 0 ]
         [ 1 ]
(c157) remvalue(eta);
(c158) dnde : zeromatrix(4,1)$ (c159) dnde : diff(emt,eta);
(c160) m00 : zeromatrix(4,4)$ (c161) for k thru 4 do
(for l thru 4 do m00[k,l] : emt[k,1]*emt[l,1]);
(c162) integrate(m00,eta,0,1)$ (c163) i00 : expand(1680*%);
      [ 128    99   -36    19  ]
      [  99   648   -81   -36  ]
      [ -36   -81   648    99  ]
      [  19   -36    99   128  ]
(c164) m01 : zeromatrix(4,4)$ (c165) for k thru 4 do
(for l thru 4 do m01[k,l] : emt[k,1]*dnde[l,1]);
(c166) integrate(m01,eta,0,1)$ (c167) i01 : expand(80*%);
      [ -40   57  -24    7 ]
      [ -57    0   81  -24 ]
      [  24  -81    0   57 ]
      [ -7    24  -57   40 ]
(c168) m11 : zeromatrix(4,4)$ (c169) for k thru 4 do
(for l thru 4 do m11[k,l] : dnde[k,1]*dnde[l,1]);
(c170) integrate(m11,eta,0,1)$ (c171) i011 : expand(40*%);
         [ 148  -189    54  -13 ]
         [-189   432  -297   54 ]
         [  54  -297   432 -189 ]
         [ -13    54  -189  148 ]
(c173) m001 : zeromatrix(4,4)$ (c174) for k thru 4 do
(for l thru 4 do m001[k,l] : emt*emt[k,1]*dnde[l,1]);
```

```
(c175) integrate(m001,eta,0,1)$ (c176) i001 : expand(84*%);
        [ [-2240 ]  [   3222 ] [ -1260 ] [   278 ] ]
        [ [-1611 ]  [   2025 ] [  -405 ] [    -9 ] ]
        [ [  630 ]  [   -648 ] [  -162 ] [   180 ] ]
        [ [ -139 ]  [    189 ] [  -189 ] [   139 ] ]
        [                                          ]
        [ [ -1611 ] [  2025 ]  [  -405 ] [    -9 ] ]
        [ [ -4050 ] [     0 ]  [  4374 ] [  -324 ] ]
        [ [  1053 ] [ -2187 ]  [  2187 ] [ -1053 ] ]
        [ [  -180 ] [   162 ]  [   648 ] [  -630 ] ]
(d177)  [                                          ]
        [  [  630 ] [  -648 ]  [  -162 ] [  180  ] ]
        [  [ 1053 ] [ -2187 ]  [  2187 ] [ -1053 ] ]
        [  [  324 ] [ -4374 ]  [     0 ] [  4050 ] ]
        [  [    9 ] [   405 ]  [ -2025 ] [  1611 ] ]
        [                                          ]
        [ [ -139 ]  [  189 ]   [  -189 ]   [  139 ]  ]
        [ [ -180 ]  [  162 ]   [   648 ]   [ -630 ]  ]
        [ [    9 ]  [  405 ]   [ -2025 ]   [ 1611 ]  ]
        [ [ -278 ]  [ 1260 ]   [ -3222 ]   [ 2240 ]  ]
(c182) m011 : zeromatrix(4,4)$ (c183) for k thru 4 do
(for l thru 4 do m011[k,l] : emt*dnde[k,l]*dnde[l,1]);
(c184) integrate(m011,eta,0,1); (c185) i011 : expand(2240*%);
        [ [  4795 ]  [ -6753 ]  [ 2481 ]   [ -523  ]  ]
        [ [  4539 ]  [ -4131 ]  [ -567 ]   [  159  ]  ]
        [ [ -1455 ]  [  1053 ]  [  243 ]   [  159  ]  ]
        [ [   409 ]  [  -753 ]  [  867 ]   [ -523  ]  ]
        [                ]
        [ [ -6753 ]   [ 9585 ]  [ -3699 ]  [  867  ]  ]
        [ [ -4131 ]   [ 8505 ]  [ -4617 ]  [  243  ]  ]
        [ [  1053 ]   [ 4131 ]  [ -4617 ]  [ -567  ]  ]
        [ [  -753 ]   [ 1971 ]  [ -3699 ]  [ 2481  ]  ]
(d186)  [                ]
        [ [ 2481 ]   [ -3699 ]   [ 1971 ]   [  -753 ] ]
        [ [ -567 ]   [ -4617 ]   [ 4131 ]   [  1053 ] ]
        [ [  243 ]   [ -4617 ]   [ 8505 ]   [ -4131 ] ]
        [ [  867 ]   [ -3699 ]   [ 9585 ]   [ -6753 ] ]
        [                ]
        [ [ -523 ]   [  867 ]  [ -753  ]  [   409 ] ]
        [ [  159 ]   [  243 ]  [  1053 ]  [ -1455 ] ]
        [ [  159 ]   [ -567 ]  [ -4131 ]  [  4539 ] ]
        [ [ -523 ]   [ 2481 ]  [ -6753 ]  [  4795 ] ]
```

Appendix II. Higher-order accurate 2D linear basis matrices

Some higher-order approximations to transformed bilinear element matrices (2.37) are shown in this appendix. In the usual first-order approximation (2.40), the element coordinate transform data, functions of nondimensional coordinates (η,ξ), are assumed to be constants evaluated at the element centroid $(\eta=0,\xi=0)$. This allows the coordinate transform to be removed from the integral and the integrals may be completed analytically. Two difficulties are (1) it is not clear when the assumption is valid, and (2) a large number of transformations are encountered, particularly in three dimensions. A desirable option would be performing only those transformations that are required to meet some accuracy objective.

In this section the functional dependence of the transforms on (η,ξ) is retained in the integral, and an order-of-magnitude analysis is used to generate more accurate integrals for comparison with the first-order centroidal approximation. Metric data for a general bilinear element are redefined here for convenience as

$$\begin{aligned} x_1 &= -\delta_L\ ,\ x_2 = \Delta_x+\delta_R\ ,\ x_3 = \Delta_x-\delta_R\ ,\ x_4 = +\delta_L\ , \\ y_1 &= -\delta_B\ ,\ y_2 = +\delta_B\ ,\ y_3 = \Delta_y+\delta_T\ ,\ y_4 = \Delta_y-\delta_T\ , \end{aligned} \tag{A.1}$$

where Δ_x,Δ_y are the measures of a rectangle superimposed through the element mid-sides, and the four measures δ are deviations from that rectangular geometry. It is assumed that Δ_x and Δ_y are of $O(h)$, to be used for measuring the significance of various terms, and that δ are of $O(h^2)$, which would require the mesh to roughly follow coordinate directions.

If the element transform functions are to be retained, a method for approximately integrating diffusion integrals is required. Here a Taylor series in (η,ξ) of order two was taken of the rational polynomial integrand to allow analytical integration of the series. The element matrices were integrated, the lowest order corrections to the first-order solution (2.40) were retained, and terms of $O(h^4)$ were neglected.

For the interpolation integral the first matrix shown is the first-order approximation (2.43) with metric data (2.46), of $O(h^2)$, and the subsequent two matrices are corrections of $O(h^3)$,

$$\begin{aligned} \texttt{M200} = \frac{\Delta_x\Delta_y}{36}&\begin{bmatrix} 4 & 2 & 1 & 2 \\ 2 & 4 & 2 & 1 \\ 1 & 2 & 4 & 2 \\ 2 & 1 & 2 & 4 \end{bmatrix} \\ &+\frac{\Delta_y(\delta_R-\delta_L)}{36}\begin{bmatrix} -2 & -1 & 0 & 0 \\ -1 & -2 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}+\frac{\Delta_x(\delta_T-\delta_B)}{36}\begin{bmatrix} -2 & 0 & 0 & -1 \\ 0 & 2 & 1 & 0 \\ 0 & 1 & 2 & 0 \\ -1 & 0 & 0 & -2 \end{bmatrix}+O(h^4)\ . \end{aligned} \tag{A.2}$$

For the convection integrals, the first two matrices shown in each direction are the first-order approximations (2.44) with data (2.46). The first of those two matrices results from the larger diagonal Jacobian term and is $O(h)$, while the second results from the off-diagonal Jacobian term and is

$O(h^2)$. The last matrix in each direction is a correction of $O(h^2)$ and neglected terms are $O(h^4)$.

$$\begin{aligned}
\texttt{M20X} &= \frac{\Delta_y}{12}\begin{bmatrix} -2 & 2 & 1 & -1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ -1 & 1 & 2 & -2 \end{bmatrix} - \frac{(\delta_T + \delta_B)}{12}\begin{bmatrix} -2 & -1 & 1 & 2 \\ -1 & -2 & 2 & 1 \\ -1 & -2 & 2 & 1 \\ -2 & -1 & 1 & 2 \end{bmatrix} \\
&+ \frac{(\delta_T - \delta_B)}{12}\begin{bmatrix} 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix} + O(h^4)\,, \\
\texttt{M20Y} &= \frac{\Delta_x}{12}\begin{bmatrix} -2 & -1 & 1 & 2 \\ -1 & -2 & 2 & 1 \\ -1 & -2 & 2 & 1 \\ -2 & -1 & 1 & 2 \end{bmatrix} - \frac{(\delta_R + \delta_L)}{12}\begin{bmatrix} -2 & 2 & 1 & -1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ -1 & 1 & 2 & -2 \end{bmatrix} \\
&- \frac{(\delta_R - \delta_L)}{12}\begin{bmatrix} 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 \\ 1 & 0 & -1 & 0 \end{bmatrix} + O(h^4)\,.
\end{aligned} \tag{A3}$$

Only one of the higher-order diffusion integrals is shown, the x-direction integral M2XX. If mixed-derivative matrices are summed, there are three terms in the first-order approximation

$$\begin{aligned}
\texttt{M2XX} &= \frac{\Delta_y}{6\Delta_x}\begin{bmatrix} 2 & -2 & -1 & 1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ 1 & -1 & -2 & -2 \end{bmatrix} - \frac{(\delta_T + \delta_B)}{4\Delta_x}\begin{bmatrix} 2 & 0 & -2 & 0 \\ 0 & -2 & 0 & 2 \\ -2 & 0 & 2 & 0 \\ 0 & 2 & 0 & -2 \end{bmatrix} \\
&+ \frac{(\delta_T + \delta_B)^2}{6\Delta_x\Delta_y}\begin{bmatrix} 2 & 1 & -1 & -2 \\ 1 & 2 & -2 & -1 \\ -1 & -2 & 2 & 1 \\ -2 & -1 & 1 & 2 \end{bmatrix} + O(h)\,,
\end{aligned} \tag{A4}$$

where the Jacobian-diagonal (11) term is $O(1)$, the mixed-derivative (12+21) term is $O(h)$, and the (22) term is $O(h^2)$. The correction terms follow, with the first two $O(h)$, the next four $O(h^2)$, the

last two $O(h^3)$, and neglected terms $O(h^4)$. The correction terms are

$$\frac{\Delta_y(\delta_R-\delta_L)}{6\Delta_x^2}\begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix} + \frac{\Delta_x(\delta_T-\delta_B)}{6\Delta_x^2}\begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

$$+ \frac{\Delta_y^2(\delta_R-\delta_L)^2}{12\Delta_x^3\Delta_y}\begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} + \frac{\Delta_x^2(\delta_T-\delta_B)^2}{18\Delta_x^3\Delta_y}\begin{bmatrix} 2 & 1 & -1 & -2 \\ 1 & 2 & -2 & -1 \\ -1 & -2 & 2 & 1 \\ -2 & -1 & 1 & 2 \end{bmatrix}$$

$$+ \frac{\Delta_x\Delta_y(\delta_R-\delta_L)}{6\Delta_x^3\Delta_y}\left\{\delta_B\begin{bmatrix} -2 & 0 & 1 & 1 \\ 0 & 2 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} + \delta_T\begin{bmatrix} 0 & 0 & -1 & 1 \\ 0 & 0 & -1 & 1 \\ -1 & -1 & 2 & 0 \\ 1 & 1 & 0 & -2 \end{bmatrix}\right\}$$

$$+ \frac{\Delta_x\Delta_y(\delta_T+\delta_B)(\delta_R-\delta_L)}{9\Delta_x^3\Delta_y}\left\{\delta_B\begin{bmatrix} -6 & 0 & 1 & 5 \\ 0 & -6 & 5 & 1 \\ 1 & 5 & -4 & -2 \\ 5 & 1 & -2 & -4 \end{bmatrix} + \delta_T\begin{bmatrix} -4 & -2 & 1 & 5 \\ -2 & -4 & 5 & 1 \\ 1 & 5 & -6 & 0 \\ 5 & 1 & 0 & -6 \end{bmatrix}\right\}. \tag{A5}$$

The correction terms reveal a possible problem with the first-order approximation to the diffusion matrix `M2XX` in that the first six correction matrices are of the same order as some of the terms in the first-order approximation. A pattern may be seen in the terms in that each first-order off-diagonal Jacobian transform term contains a sum on δ, either $\delta_R+\delta_L$ or $\delta_T+\delta_B$, and each correction term contains a difference, either $\delta_R-\delta_L$ or $\delta_T-\delta_B$.

Two special geometric cases then illustrate the extremes. For parallelogram elements, $\delta_L=\delta_R$ and $\delta_T=\delta_B$. The sums of δ are maximized (in magnitude), and the first-order off-diagonal transform terms are maximized. The differences of δ are zero, and every correction is zero, including those for interpolation and convection matrices. The first-order approximations are accurate to at least $O(h^4)$.

For rhomboid elements, $\delta_L=-\delta_R$ and $\delta_T=-\delta_B$, and the reverse occurs. The correction terms are maximized and the first-order transforms are zero; in that case the first-order approximation appears inadequate.

The analysis might also be useful for artificial diffusion algorithms which use diffusion integrals along coordinates. In that case the artificial diffusion is presumably small and in any case approximate, and the single $O(1)$ Jacobian-diagonal diffusion term should be sufficiently accurate for most purposes. The next $O(h^2)$ level of accuracy requires the $O(1)$ term, the first $O(h)$ off-diagonal transform term, and the first two correction terms in (A5).

Appendix III. Macsyma program to generate 3D element matrices

```
(c1) emt:zeromatrix(3,1)$
(c2)emt[1,1]:e1$(c3)emt[2,1]:e2$(c4)emt[3,1]:e3$(c5)emt$
(c6) nmt:zeromatrix(8,1)$
(c7) nmt[1,1]:(1-e1)*(1-e2)*(1-e3)/8$
(c8) nmt[2,1]:(1+e1)*(1-e2)*(1-e3)/8$
(c9) nmt[3,1]:(1+e1)*(1+e2)*(1-e3)/8$
nmt[4,1]:(1-e1)*(1+e2)*(1-e3)/8$
nmt[5,1]:(1-e1)*(1-e2)*(1+e3)/8$
nmt[6,1]:(1+e1)*(1-e2)*(1+e3)/8$
nmt[7,1]:(1+e1)*(1+e2)*(1+e3)/8$
nmt[8,1]:(1-e1)*(1+e2)*(1+e3)/8$ 8*nmt$
c00:zeromatrix(8,8)$
for k thru 8 do(for l thru 8 do c00[k,l]:nmt[k,1]*nmt[l,1])$
integrate(c00,e1,-1,1)$ integrate(%,e2,-1,1)$
integrate(%,e3,-1,1)$ m00:expand(27*%);
                          [ 8  4  2  4  4  2  1  2 ]
                          [ 4  8  4  2  2  4  2  1 ]
                          [ 2  4  8  4  1  2  4  2 ]
                          [ 4  2  4  8  2  1  2  4 ]
                          [ 4  2  1  2  8  4  2  4 ]
                          [ 2  4  2  1  4  8  4  2 ]
                          [ 1  2  4  2  2  4  8  4 ]
                          [ 2  1  2  4  4  2  4  8 ]
dnde:zeromatrix(8,3)$
for i thru 8 do (for j thru 3 do
(t1:nmt[i,1],t2:emt[j,1],dnde[i,j]:diff(t1,t2)));
c01:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c01[k,l]:nmt[k,1]*dnde[1,1])$ integrate(c01,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
m01:expand(18*%);
                        [ -4  4  2  -2  -2  2  1  -1 ]
                        [ -4  4  2  -2  -2  2  1  -1 ]
                        [ -2  2  4  -4  -1  1  2  -2 ]
                        [ -2  2  4  -4  -1  1  2  -2 ]
                        [ -2  2  1  -1  -4  4  2  -2 ]
                        [ -2  2  1  -1  -4  4  2  -2 ]
                        [ -1  1  2  -2  -2  2  4  -4 ]
                        [ -1  1  2  -2  -2  2  4  -4 ]
c02:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c02[k,l]:nmt[k,1]*dnde[1,2])$ integrate(c02,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
m02:expand(18*%);
                        [ -4  -2  2  4  -2  -1  1  2 ]
                        [ -4  -2  2  4  -2  -1  1  2 ]
                        [ -2  -1  1  2  -4  -2  2  4 ]
                        [ -1  -2  2  1  -2  -4  4  2 ]
                        [ -1  -2  2  1  -2  -4  4  2 ]
                        [ -2  -1  1  2  -4  -2  2  4 ]
c03:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c03[k,l]:nmt[k,1]*dnde[1,3])$ integrate(c03,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
```

```
m03:expand(18*%);
                    [ -4  -2  -1  -2  4  2  1  2 ]
                    [ -2  -4  -2  -1  2  4  2  1 ]
                    [ -1  -2  -4  -2  1  2  4  2 ]
                    [ -2  -1  -2  -4  2  1  2  4 ]
                    [ -4  -2  -1  -2  4  2  1  2 ]
                    [ -2  -4  -2  -1  2  4  2  1 ]
                    [ -1  -2  -4  -2  1  2  4  2 ]
                    [ -2  -1  -2  -4  2  1  2  4 ]
c11:zeromatrix(8,8)$
for k thru 8 do
(for l thru 8 do c11[k,l]:dnde[k,1]*dnde[l,1])$
integrate(c11,e1,-1,1)$
integrate(%,e2,-1,1)$
integrate(%,e3,-1,1)$
m11:expand(18*%);
                  [  4   -4  -2   2   2  -2  -1   1 ]
                  [ -4    4   2  -2  -2   2   1  -1 ]
                  [ -2    2   4  -4  -1   1   2  -2 ]
                  [  2   -2  -4   4   1  -1  -2   2 ]
                  [  2   -2  -1   1   4  -4  -2   2 ]
                  [ -2    2   1  -1  -4   4   2  -2 ]
                  [ -1    1   2  -2  -2   2   4  -4 ]
                  [  1   -1  -2   2   2  -2  -4   4 ]
c22:zeromatrix(8,8)$
for k thru 8 do
(for l thru 8 do c22[k,l]:dnde[k,2]*dnde[l,2])$
integrate(c22,e1,-1,1)$
integrate(%,e2,-1,1)$
integrate(%,e3,-1,1)$
m22:expand(18*%);
                  [  4   2  -2  -4   2   1  -1  -2 ]
                  [  2   4  -4  -2   1   2  -2  -1 ]
                  [ -2  -4   4   2  -1  -2   2   1 ]
                  [ -4  -2   2   4  -2  -1   1   2 ]
                  [  2   1  -1  -2   4   2  -2  -4 ]
                  [  1   2  -2  -1   2   4  -4  -2 ]
                  [ -1  -2   2   1  -2  -4   4   2 ]
                  [ -2  -1   1   2  -4  -2   2   4 ]
c33:zeromatrix(8,8)$
for k thru 8 do
(for l thru 8 do c33[k,l]:dnde[k,3]*dnde[l,3])$
integrate(c33,e1,-1,1)$
integrate(%,e2,-1,1)$
integrate(%,e3,-1,1)$
m33:expand(18*%);
                  [  4   2   1   2  -4  -2  -1  -2 ]
                  [  2   4   2   1  -2  -4  -2  -1 ]
                  [  1   2   4   2  -1  -2  -4  -2 ]
                  [  2   1   2   4  -2  -1  -2  -4 ]
                  [ -4  -2  -1  -2   4   2   1   2 ]
                  [ -2  -4  -2  -1   2   4   2   1 ]
                  [ -1  -2  -4  -2   1   2   4   2 ]
                  [ -2  -1  -2  -4   2   1   2   4 ]
c12:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
```

```
c12[k,l]:dnde[k,1]*dnde[1,2])$ integrate(c12,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$

m12:expand(12*%);
                    [  2    2  -2   -2   1    1  -1   -1 ]
                    [ -2   -2   2    2  -1   -1   1    1 ]
                    [ -2   -2   2    2  -1   -1   1    1 ]
                    [  2    2  -2   -2   1    1  -1   -1 ]
                    [  1    1  -1   -1   2    2  -2   -2 ]
                    [ -1   -1   1    1  -2   -2   2    2 ]
                    [ -1   -1   1    1  -2   -2   2    2 ]
                    [  1    1  -1   -1   2    2  -2   -2 ]
c13:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c13[k,l]:dnde[k,1]*dnde[1,3])$ integrate(c13,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
m13:expand(12*%);
                    [  2    2   1    1  -2   -2  -1   -1 ]
                    [ -2   -2  -1   -1   2    2   1    1 ]
                    [ -1   -1  -2   -2   1    1   2    2 ]
                    [  1    1   2    2  -1   -1  -2   -2 ]
                    [  2    2   1    1  -2   -2  -1   -1 ]
                    [ -2   -2  -1   -1   2    2   1    1 ]
                    [ -1   -1  -2   -2   1    1   2    2 ]
                    [  1    1   2    2  -1   -1  -2   -2 ]
c21:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c21[k,l]:dnde[k,2]*dnde[1,1])$ integrate(c21,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
m21:expand(12*%);
                    [  2  -2   -2   2   1  -1   -1   1 ]
                    [  2  -2   -2   2   1  -1   -1   1 ]
                    [ -2   2    2  -2  -1   1    1  -1 ]
                    [ -2   2    2  -2  -1   1    1  -1 ]
                    [  1  -1   -1   1   2  -2   -2   2 ]
                    [  1  -1   -1   1   2  -2   -2   2 ]
                    [ -1   1    1  -1  -2   2    2  -2 ]
                    [ -1   1    1  -1  -2   2    2  -2 ]
c23:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c23[k,l]:dnde[k,2]*dnde[1,3])$ integrate(c23,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
m23:expand(12*%);
                    [  2    1   1    2  -2   -1  -1   -2 ]
                    [  1    2   2    1  -1   -2  -2   -1 ]
                    [ -1   -2  -2   -1   1    2   2    1 ]
                    [ -2   -1  -1   -2   2    1   1    2 ]
                    [  2    1   1    2  -2   -1  -1   -2 ]
                    [  1    2   2    1  -1   -2  -2   -1 ]
                    [ -1   -2  -2   -1   1    2   2    1 ]
                    [ -2   -1  -1   -2   2    1   1    2 ]
c31:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c31[k,l]:dnde[k,3]*dnde[1,1])$ integrate(c31,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
```

```
m31:expand(12*%);
                    [  2   -2   -1   1   2  -2   -1   1  ]
                    [  2   -2   -1   1   2  -2   -1   1  ]
                    [  1   -1   -2   2   1  -1   -2   2  ]
                    [  1   -1   -2   2   1  -1   -2   2  ]
                    [ -2   2    1   -1  -2   2    1   -1 ]
                    [ -2   2    1   -1  -2   2    1   -1 ]
                    [ -1   1    2   -2  -1   1    2   -2 ]
                    [ -1   1    2   -2  -1   1    2   -2 ]
c32:zeromatrix(8,8)$
for k thru 8 do (for l thru 8 do
c32[k,l]:dnde[k,3]*dnde[l,2])$ integrate(c32,e1,-1,1)$
integrate(%,e2,-1,1)$ integrate(%,e3,-1,1)$
m32:expand(12*%);
                    [  2    1  -1   -2   2    1  -1   -2 ]
                    [  1    2  -2   -1   1    2  -2   -1 ]
                    [  1    2  -2   -1   1    2  -2   -1 ]
                    [  2    1  -1   -2   2    1  -1   -2 ]
                    [ -2   -1   1    2  -2   -1   1    2 ]
                    [ -1   -2   2    1  -1   -2   2    1 ]
                    [ -1   -2   2    1  -1   -2   2    1 ]
                    [ -2   -1   1    2  -2   -1   1    2 ]
```

Appendix IV. Macsyma program to find modified equations

```
/*fourth order modified equation*/
/*first order accurate in t:t-rhs=0,t:rhs*/
rhs;
            4                    3           2              2
(d383)-f04 x  + (- f13 t-f03) x  + (- f22 t -f12 t-f02) x
          3      2              4     3      2
+ (- f31 t -f21 t -f11 t-f01) x-f40 t -f30 t -f20 t
expand(rhs^2)$ratsubst(0,x^5,%)$ratsubst(0,t*x^4,%)$
ratsubst(0,t^2*x^2,%)$ratsubst(0,t^3*x,%)$ratsubst(0,t^4,%);
ratsubst(%,t^2,rhs)$ratsubst(0,x^5,%)$ratsubst(0,t*x^4,%)$
ratsubst(0,t^2*x^2,%)$ratsubst(0,t^3*x,%)$ratsubst(0,t^4,%);
rhs:d396;
ratsubst(d390,t^2,%)$ratsubst(0,x^5,%)$ratsubst(0,t*x^4,%)$
ratsubst(0,t^2*x^2,%)$ratsubst(0,t^3*x,%)$ratsubst(0,t^4,%);
rhs:%$ ratsubst(rhs,t,rhs)$ ratsubst(0,x^5,%)$
ratsubst(0,t*x^4,%)$
ratsubst(0,t^2*x^2,%)$ratsubst(0,t^3*x,%)$ratsubst(0,t^4,%);
ratsubst(%,t,%)$ratsubst(0,x^5,%)$ratsubst(0,t*x^4,%)$
ratsubst(0,t^2*x^2,%)$ratsubst(0,t^3*x,%)$ratsubst(0,t^4,%);
grind(%);
(-f04+f01*f13+(f11-2*f20*f01)*f03-f01^2*f22+
(f02-2*f01*f11+3*f20*f01^2)*f12-f20*f02^2+
(-2*f01*f21-f11^2+6*f20*f01*f11+(3*f30-4*f20^2)*f01^2)*f02+
f01^3*f31+3*f01^2*f11*f21+f01*f11^3-5*f20*f01^2*f11^2+
(8*f20^2-2*f30)*f01^3*f11+(f20*f30-f40)*f01^4)*x^4
+(-f03+f01*f12+(f11-2*f20*f01)*f02-
f01^2*f21-f01*f11^2+3*f20*f01^2*f11+(f30-2*f20^2)*f01^3)*x^3+
(-f02+f01*f11-f20*f01^2)*x^2-f01*x$
/*check lax-wendroff explicit*/
f20:1/2$f30:1/6$f40:1/24$f50:1/120$f01:cn$f11:0$f21:0$
f31:0$f41:0$ f02:-cn^2/2$f12:0$f22:0$f23:0$f03:cn/6$f13:0$
f23:0$ f04:-cn^2/24$ f14:0$ f05:cn/120$
ev(d423,eval);
                        4     2          3
                      cn    cn    4    cn    cn   3
(d443)               (--- - ---) x  + (--- - --) x  - cn x
                      6     8          6    6
/*fifth order*/ rhs;
               5                    4           2              3
(d685)/r/-f05 x  + (- f14 t-f04) x  + (- f23 t -f13 t-f03) x
          3      2              2           4      3      2
 + (- f23 t -f22 t -f12 t-f02) x  + (- f41 t -f31 t -f21 t
 - f11 t - f01) x
       5         4         3         2
 - f50 t  - f40 t  - f30 t  - f20 t
/*derivation omitted, same proces, show backsubstitution*/
grind(d649);
(-f05+f01*f14+(f11-2*f20*f01)*f04-f01^2*f23+
(f02-2*f01*f11+3*f20*f01^2)*f13
+(f12-2*f20*f02-2*f01*f21
-f11^2+6*f20*f01*f11+(3*f30-6*f20^2)*f01^2)*f03+
f01^3*f23+(-2*f01*f02+3*f01^2*f11-4*f20*f01^3)*f22-f01*f12^2
+((-2*f11+6*f20*f01)*f02+3*f01^2*f21+3*f01*f11^2
-12*f20*f01^2*f11+(-4*f30+10*f20^2)*f01^3)
```

```
*f12+(-f21+3*f20*f11+(3*f30-6*f20^2)*f01)*f02^2
+(3*f01^2*f31+(6*f01*f11-12*f20*f01^2)*f21+f11^3
-12*f20*f01*f11^2
+(-12*f30+30*f20^2)*f01^2*f11
+(-4*f40+20*f20*f30-20*f20^3)*f01^3)
*f02-f01^4*f41+(-4*f01^3*f11+5*f20*f01^4)*f31-2*f01^3*f21^2
+(-6*f01^2*f11^2+20*f20*f01^3*f11+(5*f30-15*f20^2)*f01^4)*f21
-f01*f11^4+10*f20*f01^2*f11^3+(10*f30-30*f20^2)*f01^3*f11^2
+(5*f40-30*f20*f30+35*f20^3)*f01^4*f11
+(f50-6*f20*f40-3*f30^2+21*f20^2*f30-14*f20^4)
*f01^5)*x^5+(-f04+f01*f13+(f11-2*f20*f01)*f03-f01^2*f22+
(f02-2*f01*f11+3*f20*f01^2)*f12-f20*f02^2
+(-2*f01*f21-f11^2+6*f20*f01*f11
+(3*f30-6*f20^2)*f01^2)*f02+f01^3*f31
+(3*f01^2*f11-4*f20*f01^3)*f21+f01*f11^3-6*f20*f01^2*f11^2+
(-4*f30+10*f20^2)*f01^3*f11+(-f40+5*f20*f30-5*f20^3)*f01^4)*x^4
+(-f03+f01*f12+(f11-2*f20*f01)*f02
-f01^2*f21-f01*f11^2+3*f20*f01^2*f11+f30
-2*f20^2)*f01^3)*x^3+(-f02+f01*f11-f20*f01^2)*x^2-f01*x$
tt2:d649^2$ratsubst(0,x^6,%)$ratsubst(0,t*x^5,%)$
ratsubst(0,t^2*x^4,%)$ratsubst(0,t^3*x^3,%)$
ratsubst(0,t^4*x^2,%)$ratsubst(0,t^5*x,%)$
ratsubst(0,t^6,%)$tt2:%$ ratsubst(tt2,t^2,rhs)$
ratsubst(0,x^6,%)$ratsubst(0,t*x^5,%)$ratsubst(0,t^2*x^4,%)$
ratsubst(0,t^3*x^3,%)$ ratsubst(0,t^4*x^2,%)$
ratsubst(0,t^5*x,%)$ratsubst(0,t^6,%)$ ratsubst(d649,t,d669)$
ratsubst(0,x^6,%)$ratsubst(0,t*x^5,%)$ratsubst(0,t^2*x^4,%)$
ratsubst(0,t^3*x^3,%)$ ratsubst(0,t^4*x^2,%)$
ratsubst(0,t^5*x,%)$ratsubst(0,t^6,%)$ ratcoef(%,t);
(d683)/r/                                   0
expand(d682-d649);
(d684)                                      0
/*check lax-wendroff explicit*/
f20:1/2$f30:1/6$f40:1/24$f50:1/120$f01:cn$f11:0$f21:0$f31:0$
f41:0$f02:-cn^2/2$f12:0$f22:0$f23:0$f03:cn/6$f13:0$f23:0$
f04:-cn^2/24$ f14:0$ f05:cn/120$ ev(rhs,eval,expand);
                 5      2  4        3      2  2
             cn x     cn  x     cn x     cn  x
(d707)     - ----- + ------ - ----- + ------ - cn x -
              120       24        6        2
  5      4     3     2
t      t     t     t
---  -  --  -  --  -  --
120    24    6     2
ev(d682,eval,expand)$
       5        3         5          4          2   4
 (6 cn   - 5 cn   - cn) x   + (15 cn   - 15 cn ) x
 ------------------------------------------------------+
                   120
       3            3
(20 cn   - 20 cn) x   - 120 cn x
-------------------------------
                   120
ratcoef(d708,x,5)$factor(%);
                                                    2
                        (cn - 1) cn (cn + 1) (6 cn  + 1)
```

```
(d713)                    --------------------------------
                                        120
/*crank-nicolson with diffusion*/
xp:taylor(exp(x),x,0,5)$ xm:taylor(exp(-x),x,0,5);
tp:taylor(exp(t),t,0,5);
cne: tp-1 +cn*(tp+1)*(xp-xm)/4 +nu*(tp+1)*(xp+xm-2)/2;
rx0:ratcoef(cne,x,0);
                                  2    3    4    5
                                 t    t    t    t
(d730)/t/                   t + -- + -- + -- + --- + . . .
                                 2    6    24   120
f20:ratcoef(rx0,t,2)$ f30:ratcoef(rx0,t,3)$ f40:ratcoef(rx0,t,4)$
f50:ratcoef(rx0,t,5)$ rx1:ratcoef(cne,x,1);
                                  2       3       4       5
                        cn t    cn t    cn t    cn t    cn t
(d732)/t/          cn + ---- + ----- + ----- + ----- + ----- + . . .
                         2       4       12      48      240
f01:ratcoef(rx1,t,0)$f11:ratcoef(rx1,t,1)$f21:ratcoef(rx1,t,2)$
f31:ratcoef(rx1,t,3)$f41:ratcoef(rx1,t,4)$ rx2:ratcoef(cne,x,2);
                                  2       3       4       5
                        nu t    nu t    nu t    nu t    nu t
(d738)/t/          nu + ---- + ----- + ----- + ----- + ----- + . . .
                         2       4       12      48      240
f02:ratcoef(rx2,t,0)$f12:ratcoef(rx2,t,1)$f22:ratcoef(rx2,t,2)$
f23:ratcoef(rx2,t,3)$ rx3:ratcoef(cne,x,3);
                                  2       3       4       5
                  cn    cn t    cn t    cn t    cn t    cn t
(d743)/t/         -- + ---- + ----- + ----- + ----- + ----- + . . .
                  6       12      24      72      288     1440
f03:ratcoef(rx3,t,0)$f13:ratcoef(rx3,t,1)$f23:ratcoef(rx3,t,2)$
rx4:ratcoef(cne,x,4);
                                  2       3       4       5
                  nu    nu t    nu t    nu t    nu t    nu t
(d747)/t/         -- + ---- + ----- + ----- + ----- + ----- + . . .
                  12      24      48      144     576     2880
f04:ratcoef(rx4,t,0)$f14:ratcoef(rx4,t,1)$
rx5:ratcoef(cne,x,5);
                                 2       3       4       5
                  cn    cn t    cn t    cn t    cn t    cn t
(d750)/t/         --- + ---- + ----- + ----- + ----- + ----- + . . .
                  120   240     480     1440    5760    28800
f05:ratcoef(%,t,0)$
ev(rhs,eval,expand);
                5         4        4        2  3         3
          cn x    nu t x    nu x    cn t  x    cn t x
(d753)  - ----- - ------- - ----- - -------- - -------
           120       24       12        24        12
      3        3  2
  cn x     nu t  x
- ----- - --------
    6         12
       2  2          2                  4         3
   nu t  x     nu t x         2    cn t  x   cn t  x
 - -------- - ------- - nu x   - ------- - -------
       4           2                48        12
       2
```

```
  cn t  x   cn t x
- ------- - ------ - cn x
   4        2
    5     4    3    2
   t     t    t    t
 - --- - -- - -- - --
   120   24   6    2
ev(d682,eval,expand);
                          2       5        3          5
(d754)/r/ - ((60 cn nu  + 3 cn  + 10 cn  + 2 cn) x
        2             4
+ (60 cn  + 20) nu x
        3           3            2
+ (20 cn  + 40 cn) x  + 240 nu x  + 240 cn x)/240
ratcoef(d754,x,2);
(d755)/r/                                  - nu
ratcoef(d754,x,3)$factor(%);
                                             2
                                      cn (cn  + 2)
(d762)                                - ------------
                                            12
ratcoef(d754,x,4)$factor(%);
                                           2
                                    (3 cn  + 1) nu
(d764)                              - --------------
                                           12
ratcoef(d754,x,5)$factor(%);
                                    2       4        2
                           cn (60 nu  + 3 cn  + 10 cn  + 2)
(d760)                     - ---------------------------------
                                         240
/*galerkin trapezoidal fem with diffusion*/
cne:(tp-1)*(4+xp+xm)/6
+cn*(tp+1)*(xp-xm)/4+nu*(tp+1)*(xp+xm-2)/2;
rx0:ratcoef(cne,x,0); f20:ratcoef(rx0,t,2)$ f30:ratcoef(rx0,t,3)$
f40:ratcoef(rx0,t,4)$ f50:ratcoef(rx0,t,5)$
rx1:ratcoef(cne,x,1); f01:ratcoef(rx1,t,0)$ f11:ratcoef(rx1,t,1)$
f21:ratcoef(rx1,t,2)$ f31:ratcoef(rx1,t,3)$ f41:ratcoef(rx1,t,4)$
rx2:ratcoef(cne,x,2); f02:ratcoef(rx2,t,0)$ f12:ratcoef(rx2,t,1)$
f22:ratcoef(rx2,t,2)$ f23:ratcoef(rx2,t,3)$
rx3:ratcoef(cne,x,3); f03:ratcoef(rx3,t,0)$ f13:ratcoef(rx3,t,1)$
f23:ratcoef(rx3,t,2)$
rx4:ratcoef(cne,x,4); f04:ratcoef(rx4,t,0)$ f14:ratcoef(rx4,t,1)$
rx5:ratcoef(cne,x,5);
ev(d682,eval,expand);
                          2       5          5
(d790)/r/ - ((180 cn nu  + 9 cn  - 4 cn) x  +
        2             4         3  3
+ (180 cn  - 60) nu x  + 60 cn  x
          2
+ 720 nu x  + 720 cn x)/720
ratcoef(d790,x,1);
(d841)/r/                                  - cn
ratcoef(d790,x,2);
(d842)/r/                                  - nu
ratcoef(d790,x,3)$factor(%);
```

```
                                        3
                                      cn
(d844)                              - ---
                                      12

ratcoef(d790,x,4)$factor(%);
                                        2
                                  (3 cn  - 1) nu
(d846)                            - --------------
                                        12

ratcoef(d790,x,5)$factor(%);
                                        2       4
                             cn (180 nu  + 9 cn  - 4)
(d848)                     - ------------------------
                                      720
/*4th order trapezoidal fem with diffusion*/
cne: (tp-1)*((2+cn^2)*(xp+xm)+(8-2*cn^2))/12
 +cn*(tp+1)*(xp-xm)/4 +nu*(tp+1)*(xp+xm-2)/2;
ratcoef(d823,x,1);
(d828)/r/                             - cn
ratcoef(d823,x,2);
(d829)/r/                             - nu
ratcoef(d823,x,3)$factor(%);
       (d831)                           0
ratcoef(d823,x,4)$factor(%);
/*4th order acc. if nu=0*/
                                        2
                                  (2 cn  - 1) nu
(d833)                            - --------------
                                        12

ratcoef(d823,x,5)$factor(%);
ratcoef(d823,x,5)$factor(%);
                                  2     4       2
                           cn (180 nu  - cn  + 5 cn  - 4)
(d835)                   - -------------------------------
                                      720

/*2dimensions*/
/*substitute in 1dform*/
grind(%);
(-f04+f01*f13+(f11-2*f20*f01)*f03-f01^2*f22+
(f02-2*f01*f11+3*f20*f01^2)*f12-f20*f02^2+
(-2*f01*f21-f11^2+6*f20*f01*f11+(3*f30-4*f20^2)*f01^2)*f02+
f01^3*f31+3*f01^2*f11*f21+f01*f11^3-5*f20*f01^2*f11^2+
(8*f20^2-2*f30)*f01^3*f11+(f20*f30-f40)*f01^4)*x^4
+(-f03+f01*f12+(f11-2*f20*f01)*f02-
f01^2*f21-f01*f11^2+3*f20*f01^2*f11+(f30-2*f20^2)*f01^3)*x^3+
(-f02+f01*f11-f20*f01^2)*x^2-f01*x$
ratsubst(f010*x+f001*y,f01*x,%)$ ratsubst(f110*x+f101*y,f11*x,%)$
ratsubst(f210*x+f201*y,f21*x,%)$ ratsubst(f310*x+f301*y,f31*x,%)$
ratsubst(f020*x^2+f011*x*y+f002*y^2,f02*x^2,%)$
ratsubst(f120*x^2+f111*x*y+f102*y^2,f12*x^2,%)$
ratsubst(f220*x^2+f211*x*y+f202*y^2,f22*x^2,%)$
ratsubst(f030*x^3+f021*x^2*y+f012*x*y^2+f003*y^3,f03*x^3,%)$
ratsubst(f130*x^3+f121*x^2*y+f112*x*y^2+f103*y^3,f13*x^3,%)$
ratsubst(f040*x^4+f031*x^3*y+f022*x^2*y^2+f013*x*y^3
+f004*y^4,f04*x^4,%)$ ratsubst(f200,f20,%)$ ratsubst(f300,f30,%)$
ratsubst(f400,f40,%)$
```

Appendix V. Macsyma program to find cubic basis phase velocity

```
m:matrix([m11,m12,m13],[m13,648,m23],[m12,m32,648]);
[ m11  m12  m13 ]
[ m13  648  m23 ]
[ m12  m32  648 ]
u:matrix([u11,u12,u13],[u13,0,u23],[u12,u32,0]);
[ u11  u12  u13 ]
[ u13   0   u23 ]
[ u12  u32   0  ]
ml:m+u$ mr:m-u$ detml:det(ml)$ mli:factor(%*invert(ml))$
%[1,1];
- (u23 u32+m23 u32+m32 u23+m23 m32-419904)
mi:factor(mli . mr)$
%[1,1];
                                    2
u11 u23 u32-m11 u23 u32-u13  u32+m23 u11 u32-m11 m23 u32
                           2   2             2         2
+ m13 u32-u12 u23+m32 u11 u23-m11 m32 u23+m12 u23-m32 u13
          2       2
+ 1296 u12 u13-m23 u12+m23 m32 u11-419904 u11-m11 m23 m32
                  2
+ m13 m32+m12  m23-1296 m12 m13+419904 m11
ep3:exp(3*%i*z)$ ep2:exp(2*%i*z)$ ep1:exp(1*%i*z)$
em1:exp(-1*%i*z)$ em2:exp(-2*%i*z)$ em3:exp(-3*%i*z)$
m11:256+19*(ep3+em3); m12:99*em1-36*ep2; m13:-36*em2+99*ep1;
m23:-81*em1; m32:-81*ep1;
u11:7*cn*(7*(ep3-em3)); u12:7*cn*(57*em1+24*ep2);
u13:7*cn*(-24*em2-57*ep1); u23:-7*cn*(81*em1); u32:7*cn*(81*ep1);
cn:1/100; z:%pi/2$
...
cabs(eve);
1.0b0
carg(eve)$
ev(%/(z*cn),eval,numer);
1.001154
cn:1/2$
mie:ev(mi,infeval,expand,factor)$
%[1,1];
642978 (59 %i+90)
detmle:ev(detml,infeval,expand,factor)$
mir:rectform(mie/detmle)$
mpr:denom(factor(mir[3,3]));
204387
mirr:factor(mir*mpr);
[ 3 (17410 %i+22043)  27 (8497 %i+4227)  27 (7701 %i+1153) ]
[                                                          ]
[ 2 (322339 %i+6336)                                       ]
[ -------------------- 25435 %i+103874 -(136025 %i+18101)  ]
[  9                                                       ]
[                                                          ]
[ 14 (42165 %i+26404)                                      ]
[ ----------------------(77861 %i+89279)  7 (5057 %i+9772) ]
[  9                                                       ]
ev:eigenvalues(%)$
e1:first(ev[1])/mpr$ bfloat(%)$ eve:expand(%);
```

```
- 9.9831481065372748316b-1 %i-5.803049913116587443lb-2
cabs(eve);
1.0b0
carg(eve)$ ev(%/(z*cn),eval,numer);
- 2.073928
e2:second(ev[1])/mpr$ bfloat(%)$ eve:expand(%);
8.7046632124352331609b-1 %i+4.9222797927461139891b-1
cabs(eve);
1.0b0
carg(eve)$ ev(%/(z*cn),eval,numer);
1.34473
e3:last(ev[1])/mpr$ bfloat(%)$ eve:expand(%);
6.8103435739593711486b-1 %i+7.3225146230397040706b-1
cabs(eve);
1.0b0
carg(eve)$ ev(%/(z*cn),eval,numer);
0.9538786
```

Vita

David J. Chaffin, a native of Knoxville, Tennessee, received Bachelor's and Master's degrees in Mechanical Engineering from the University of Tennessee in 1979 and 1980. He was a mechanical engineer for the Tennessee Valley Authority before entering the Ph.D. program in Engineering Science and Mechanics.