Multigrid Solution for High-Order Discontinuous Galerkin Discretizations of the Compressible Navier-Stokes Equations

by

Todd A. Oliver

B.S., Massachusetts Institute of Technology (2002)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2004

© Massachusetts Institute of Technology 2004. All rights reserved.

Author . . . .
Department of Aeronautics and Astronautics
August 20, 2004

Certified by . . . .
David L. Darmofal
Associate Professor
Thesis Supervisor

Accepted by . . . .
Jaime Peraire
Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

Multigrid Solution for High-Order Discontinuous Galerkin Discretizations of the Compressible Navier-Stokes Equations

by

Todd A. Oliver

Submitted to the Department of Aeronautics and Astronautics
on August 20, 2004, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

A high-order discontinuous Galerkin finite element discretization and p-multigrid solution procedure for the compressible Navier-Stokes equations are presented. The discretization has an element-compact stencil such that only elements sharing a face are coupled, regardless of the solution space. This limited coupling maximizes the effectiveness of the p-multigrid solver, which relies on an element-line Jacobi smoother. The element-line Jacobi smoother solves implicitly on lines of elements formed based on the coupling between elements in a $p = 0$ discretization of the scalar transport equation. Fourier analysis of 2-D scalar convection-diffusion shows that the element-line Jacobi smoother as well as the simpler element Jacobi smoother are stable independent of p and flow condition. Mesh refinement studies for simple problems with analytic solutions demonstrate that the discretization achieves optimal order of accuracy of $O(h^{p+1})$. A subsonic, airfoil test case shows that the multigrid convergence rate is independent of p but weakly dependent on h. Finally, higher-order is shown to outperform grid refinement in terms of the time required to reach a desired accuracy level.

Thesis Supervisor: David L. Darmofal
Title: Associate Professor

Acknowledgments

I would like to express my gratitude to the many people who have made this thesis possible.

First, I would like to thank my advisor, Prof. David Darmofal, for his insight, guidance, and encouragement throughout this research and for giving me the opportunity to work on Project X. I very much look forward to continuing our work together. Of course, this work would not have been possible without the tireless efforts of the entire PX team (Mike Brasher, David Darmofal, Krzysztof Fidkowski, Bob Haimes, James Lu, Paul Nicholson, Jaime Peraire, and Matthieu Serrano). Special thanks go to Krzysztof and Garrett Barter, for their insightful comments during the drafting of this thesis.

For their contributions to the results shown in this thesis, I would like to thank David Venditti, who provided the FUN2D solutions, and Charlie Swanson, who provided the baseline mesh for the NACA 0012 case.

I would like to thank everyone at ACDL for making the last two years a lot of fun. MIT can be a difficult place to work, but friends like Mike, Garrett, Hector, Paul, and Matthieu make even the stressful times bearable.

Of course, I would like to thank my family—Mom, Dad, Lee, and Lauren—for their constant support, without which I'm sure I would not have gotten this far. And, last but not least, I would like to thank Christy for her extraordinary optimism—especially when I'm in a bad mood because the code isn't working—and love and support.

This work was funded by the National Defense Science and Engineering Fellowship provided by the U.S. Department of Defense.

Contents

1 Introduction **13**

1.1 Motivation 13

1.2 Background 14

  1.2.1 Higher-Order Methods 14

  1.2.2 Discontinuous Galerkin Methods 16

  1.2.3 Multigrid for Aerodynamic Applications 17

1.3 Outline of Thesis 17

2 Discontinuous Galerkin Discretization **19**

2.1 DG for Euler 19

2.2 DG for Navier-Stokes 20

  2.2.1 Flux Formulation 21

  2.2.2 Primal Formulation 22

  2.2.3 Bassi and Rebay Discretization 24

2.3 The Stabilization Parameter, η_f 28

  2.3.1 Motivation 28

  2.3.2 Formulation 29

2.4 Boundary Treatment 31

  2.4.1 Geometry Representation: Curved Boundaries 31

  2.4.2 Boundary Conditions 32

2.5 Final Discrete System 35

3 Solution Method **37**

3.1 Preconditioners 37

3.2 Line Creation 38

  3.2.1 Connectivity Criterion 39

  3.2.2 Line Creation Algorithm 39

3.3 p-Multigrid 42

3.3.1 Motivation . . . 42
3.3.2 FAS and Two-level Multigrid . . . 42
3.3.3 V-cycles and FMG . . . 43

4 Stability Analysis **45**
4.1 Outline of Analysis . . . 45
4.2 One Dimensional Analysis . . . 46
4.3 Two Dimensional Analysis . . . 48

5 Numerical Results **53**
5.1 Poiseuille Flow . . . 53
5.2 Circular Channel Flow . . . 55
5.3 NACA 0012 Airfoil . . . 58
5.3.1 Accuracy Results . . . 59
5.3.2 Iterative Convergence Results . . . 61
5.3.3 Timing Results . . . 64

6 Conclusions **67**

List of Figures

1-1 Drag results for subsonic NACA 0012 test case. Taken from Zingg *et al.* [59]. Reproduced with permission. 16

2-1 1-D stencil for first Bassi and Rebay scheme 26
2-2 1-D stencil for BR2 scheme 27
2-3 Mesh refinement results for $0 \leq p \leq 2$ using $\eta_f = 3$ 29
2-4 Definitions of h^+, h^-, and Δn 31
2-5 Mesh refinement results for $0 \leq p \leq 2$ using η_f as defined in Eqn **??** 32

3-1 Possible line configuration: (a) after Stage I and (b) after stage II. Reproduced with permission from [22]. 40
3-2 Lines formed around NACA 0012 in $M = 0.5$, $Re = 5000$, $\alpha = 0^o$ flow . . . 41

4-1 Eigenvalue footprints for element Jacobi preconditioned 1-D convection-diffusion 49
4-2 Stencil of element 0 50
4-3 Eigenvalue footprints for element Jacobi preconditioned 2-D convection-diffusion 52
4-4 Eigenvalue footprints for element-line Jacobi preconditioned 2-D convection-diffusion 52

5-1 Domain for Poiseuille flow test case 54
5-2 Accuracy results for $0 \leq p \leq 3$ for Poiseuille flow test case 55
5-3 Domain for circular channel test case 56
5-4 Accuracy results for $0 \leq p \leq 3$ for circular channel test case 57
5-5 Radius error for inner wall, coarse grid boundary face 58
5-6 Coarse NACA 0012 grid, 672 elements 59
5-7 Drag versus number of elements 60
5-8 Drag versus degrees of freedom 60
5-9 Residual convergence history 62
5-10 Drag convergence history 63

5-11 Absolute drag versus CPU time 64
5-12 Drag error versus CPU time 65
5-13 Solver comparison for 2688 element mesh 66

List of Tables

2.1 Types of inflow/outflow boundary conditions. 33

Chapter 1

Introduction

The simulation of complex physical phenomena using numerical methods has become an invaluable part of modern science and engineering. The utility of these simulations has led to the evolution of ever more efficient and accurate methods. In particular, numerous research efforts have been aimed at developing high-order accurate algorithms for solving partial differential equations. These efforts have led to many types of numerical schemes, including higher-order finite difference [33, 59, 56], finite volume [7, 57], and finite element [8, 20, 6, 40] methods for both structured and unstructured meshes. Despite these developments, in applied aerodynamics, most computational fluid dynamics (CFD) calculations are performed using methods that are at best second-order accurate. These methods are very costly, both in terms of computational resources and time required to reach engineering-required accuracy. Higher-order methods are of interest because they have the potential to provide significant reductions in the time necessary to obtain accurate solutions. Motivated by this potential, the goal of this thesis is to contribute to the development of a higher-order CFD algorithm which is practical for use in an applied aerodynamics setting. Specifically, the thesis details a high-order discontinuous Galerkin (DG) discretization of the compressible Navier-Stokes equations and a multigrid solution procedure for the resulting nonlinear discrete system.

1.1 Motivation

While CFD has matured significantly in past decades, in terms of time and computational resources, large aerodynamic simulations of aerospace vehicles are still very expensive. In this applied aerodynamics context, the discretization of the Euler and/or Navier-Stokes equations is performed almost exclusively by finite volume methods. The evolution of these methods, including the incorporation of upwinding mechanisms [51, 46, 52, 47, 53] and advances in solution techniques for viscous flows [4, 41, 37, 38], has made the simulation of complex problems possible. However, the standard algorithms remain at best second-order accurate, meaning that the error decreases as $O(h^2)$.

Moreover, while these methods are used heavily in aerospace design today, the time required to obtain reliably accurate solutions has hindered the realization of the full potential of CFD in the design process. In fact, it is unclear if the accuracy of current second-order finite volume methods is sufficient for engineering purposes. The results of the two AIAA Drag Prediction Workshops (DPW) [35, 31] suggest that the CFD technology in use today may not produce adequate accuracy given current grids. Numerous authors [54, 32, 22] have shown that the spread of the drag results obtained by the DPW participants is unacceptable given the stringent accuracy requirements of aircraft design.

This problem could be alleviated by the development of a high-order CFD algorithm. Specifically, a high-order method could reduce the gridding requirements and time necessary to achieve a desired accuracy level. Traditional finite volume methods rely on extended stencils to achieve high-order accuracy, which leads to difficulty in achieving stable iterative algorithms and higher-order accuracy on unstructured meshes. Alternatively, an attractive approach for achieving higher-order accuracy is the discontinuous Galerkin (DG) formulation in which element-to-element coupling exists only through the fluxes at the shared boundaries between elements.

Recently, Fidkowski and Darmofal [22, 23] developed a p-multigrid method for the solution of high-order, DG discretizations of the Euler equations of gas dynamics. They achieved significant reductions in the computational time required to obtain high accuracy solutions by using high-order discretizations rather than highly refined meshes. This thesis describes the extension of the algorithm introduced by Fidkowski and Darmofal to viscous flows.

1.2 Background

1.2.1 Higher-Order Methods

The first high-order accurate numerical methods were spectral methods [24, 15], where the solution of a differential equation is approximated over the entire domain using a high-order expansion. Choosing the expansion functions properly, one can achieve arbitrarily high-order accuracy. However, because of the global nature of the expansion functions, spectral methods are typically limited to very simple domains with simple boundary conditions.

Motivated by the prospect of obtaining the rapid convergence rates of spectral methods with the greater geometric versatility provided by finite element methods, researchers in the early 1980s introduced the p-type finite element method. In the p-type finite element method, the grid spacing, h, is fixed, and the interpolation order, p, is increased to drive the error down. In 1981, Babuska *et al.* [6] applied this method to elasticity problems. They concluded that based on degrees of freedom, the rate of convergence of the p-type method cannot be slower than that of the h-type and that, in cases with singularities present at vertices, the convergence rate of the p-type is twice as fast. In 1984, Patera [40] introduced a variant p-type method, known as the spectral element method, and used it to solve the incompressible Navier-Stokes equations for flow in an expanding duct. Korczak and Patera [30] later extended this method to more general, curved geometries.

Finite element methods are attractive for achieving high-order accuracy because, for smooth problems, the order of accuracy is controlled by the order of the solution and test function spaces. However, it is well known that standard, continuous Galerkin methods are unstable for the convection operator [55]. Thus, the solution of the Euler or Navier-Stokes equations requires the addition of a stabilization term, like that used in the Streamwise Upwind Petrov Galerkin method [27].

Also motivated by the possibility of obtaining spectral-like results in a more flexible geometric framework, Lele [33] introduced up to tenth-order, compact, finite difference schemes. Using a Fourier analysis of the differencing errors, he showed that the compact, high-order schemes have a larger resolving efficiency, where resolving efficiency is the fraction of waves that are resolved to a given accuracy, than traditional finite difference approximations. This work was extended to more general geometries by Visbal and Gaitonde [56], who used up to sixth-order, compact finite difference discretizations to solve the compressible Navier-Stokes equations on curvilinear meshes.

Further work for the compressible Navier-Stokes equations on structured meshes is presented by Zingg *et al.* [59], who compared the results of a fourth-order central difference discretization to a number of lower-order schemes. One of their results is shown in Figure 1-1. The higher-order scheme is seen to dramatically outperform the second-order (Matrix, CUSP, and Roe) methods in terms of number of nodes required to accurately compute the drag on an airfoil in subsonic flow.

In an unstructured, finite volume context, Barth [7] introduced the k-exact reconstruction method, which is based on a least-squares reconstruction procedure that requires an extended stencil. As noted above, the large stencil typically required is a limiting factor in the development and application of higher-order finite volume methods. Alternatively, Wang *et al.* [57] has recently developed the spectral volume method, where each cell (spectral volume) in the domain is subdivided into additional control volumes. The state averages within these control volumes are used to build a higher-order reconstruction within the

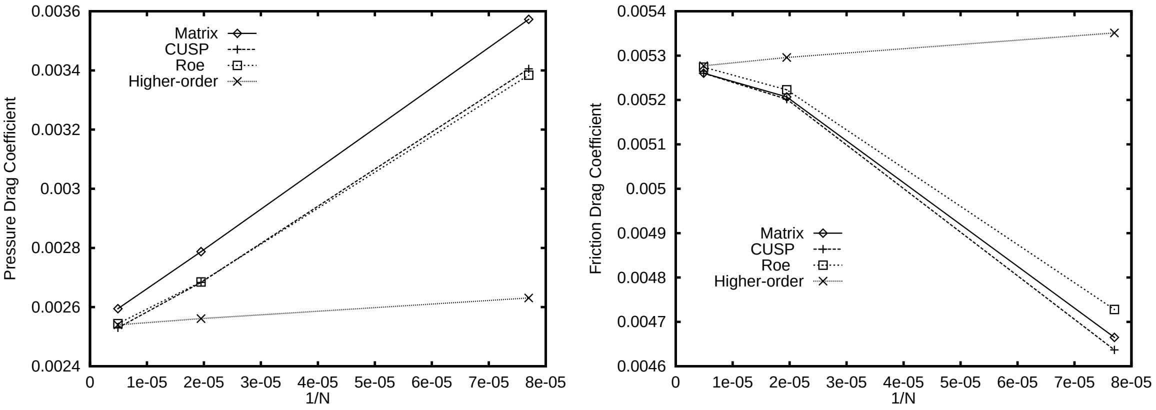


Figure 1-1: Drag results for subsonic NACA 0012 test case. Taken from Zingg *et al.* [59]. Reproduced with permission.

spectral volume. To increase the order of accuracy, additional degrees of freedom are added by further subdividing each spectral volume. Thus, at the spectral volume level, the scheme has a nearest neighbor stencil regardless of the order of accuracy.

1.2.2 Discontinuous Galerkin Methods

In 1973, Reed and Hill [44] introduced the DG method for the neutron transport equation. Since that time, development of the method has proceeded rapidly. Cockburn *et al.* present an extensive history of DG methods in [16]. Highlights of this history are mentioned here.

In 1974, LeSaint and Raviart [34] derived the first a priori error estimates of the DG method for linear hyperbolic problems. They proved a rate of convergence of $O(h^p)$ in the $L_2(\Omega)$-norm. Johnson and Pitkaranta [29] and Richter [45] later improved upon these original estimates. Johnson and Pitkaranta proved that, in the most general case, $O(h^{p+1/2})$ is the optimal convergence rate, while Richter showed that, assuming the characteristic direction is not exactly aligned with the grid, $O(h^{p+1})$ can be obtained.

A breakthrough in the application of DG methods to nonlinear hyperbolic problems was made by Cockburn and Shu [18], who introduced the Runge Kutta Discontinuous Galerkin (RKDG) method. The original RKDG method uses an explicit TVD second-order Runge Kutta scheme introduced by Shu and Osher [49]. In 1989, Cockburn and Shu [17] generalized the method to be higher-order in time as well as space. Cockburn and Shu summarize the RKDG method, including the details of a generalized slope limiter for controlling oscillations, in [20].

Independent of the above work, Allmaras [1] and Allmaras and Giles [3] developed a second-order DG scheme for the 2-D Euler equations. Their method is the extension of van Leer's method of moments [50] from the 1-D, linear wave equation to the 2-D Euler equations. Thus, it requires that state and gradient averages be computed at each cell to allow linear reconstruction of the state variables. Halt [25] later extended this work to be higher-order accurate.

For elliptic operators, in the late 1970s and early 1980s, Arnold [5] and Wheeler [58] introduced discontinuous finite element methods known as penalty methods. While these schemes were not developed as DG methods, they have now been brought into the unified DG framework [21]. More recently, many researchers [13, 9, 8, 20, 19, 42] have applied DG methods to diffusive operators. One procedure, pioneered by Bassi and Rebay [9, 11] and generalized by Cockburn and Shu [20, 19], is to rewrite a second-order equation as a first-order system and then discretize the first-order system using the DG formulation. This method has been successfully applied to the compressible Navier-Stokes equations and Reynolds Averaged Navier-Stokes equations by Bassi and Rebay [11, 10]. Arnold *et al.* [21] provides a unified analysis, including error estimates, of most of the DG methods available for elliptic operators.

1.2.3 Multigrid for Aerodynamic Applications

The use of multigrid for the solution of the Euler equations was pioneered by Jameson in 1983, who demonstrated a significant convergence speedup in the solution of 2-D transonic flows on structured meshes [28]. Since that time, there have been many advances in the application of multigrid methods to aerodynamic problems. For example, Mavriplis [36] introduced a method of performing multigrid on unstructured triangular meshes; Allmaras [2] has examined the requirements for the elimination of all error modes; and Pierce and Giles [41] presented efficient methods for the Euler and Navier-Stokes equations. Specifically for DG discretizations, in 2002, Bassi and Rebay [12] introduced a semi-implicit p-multigrid algorithm and used it to solve the DG discretization of the Euler equations.

This work builds directly on that of Fidkowski and Darmofal [22, 23], who developed a p-multigrid algorithm with element-line Jacobi relaxation for solving the DG discretization of the Euler equations. They were able to achieve p-independent asymptotic convergence rates as well as significant time savings versus a simpler element Jacobi preconditioned p-multigrid scheme. The multigrid solver implemented by Fidkowski is used here with only minor modification to the element coupling criterion used by the line creation procedure.

1.3 Outline of Thesis

This thesis presents a multigrid solution technique for a high-order DG discretization of the compressible Navier-Stokes equations. The equations are discretized using the second method of Bassi and Rebay (BR2) [11, 10], which is described in detail in Chapter 2. Chapter 3 describes the multigrid algorithm developed by Fidkowski and Darmofal [22, 23] and its extention to high Reynolds number viscous flows. Stability analysis, presented in Chapter 4, shows that the single-step element and element-line Jacobi relaxation schemes are stable independent of p and flow conditions. Finally, 2-D laminar results shown in Chapter 5 demonstrate that higher-order schemes provide significant savings in terms of the number of elements, degrees of freedom, and time required to achieve a desired accuracy level.

Chapter 2

Discontinuous Galerkin Discretization

This chapter develops a high-order accurate discretization of the compressible Navier-Stokes equations. The discretization of the inviscid terms uses the standard DG formulation, which relies on Riemann solvers for the calculation of the inter-element fluxes. The bulk of the chapter focuses on the discretization of the viscous terms, which is done using the second formulation of Bassi and Rebay (BR2) [11, 10].

2.1 DG for Euler

The two-dimensional Euler equations in strong, conservation form are given by

$$\mathbf{u}_t + \nabla \cdot \mathcal{F}_i(\mathbf{u}) = 0, \tag{2.1}$$

where $\mathbf{u}$ is the conservative state vector,

$$\mathbf{u} = (\rho \quad \rho u \quad \rho v \quad \rho E)^T,$$

$\mathcal{F}_i = (\mathbf{F}_i^x, \mathbf{F}_i^y)$ is the inviscid flux vector,

$$\mathbf{F}_i^x = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \end{pmatrix}, \quad \mathbf{F}_i^y = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho v H \end{pmatrix},$$

ρ is the fluid density, u and v are velocity components, p is the pressure, and E is the total internal energy per unit mass. Thus, the total enthalpy per unit mass, H, is given by $H = E + p/\rho$, and, assuming the fluid obeys the perfect gas equation of state, the pressure is $p = (\gamma - 1)\rho[E - (u^2 + v^2)/2]$, where γ is the ratio of specific heats of the fluid.

Multiplying Eqn. 2.1 by a vector-valued test function $\mathbf{v}$ and integrating by parts, one obtains the weak formulation:

$$\int_\Omega \mathbf{v}^T \mathbf{u}_t \, d\mathbf{x} - \int_\Omega \nabla \mathbf{v}^T \cdot \mathcal{F}_i \, d\mathbf{x} + \int_{\partial\Omega} \mathbf{v}^T \mathcal{F}_i \cdot \hat{\mathbf{n}} \, ds = 0, \quad \forall \mathbf{v} \in H^1(\Omega),$$

where Ω is the domain, $\partial\Omega$ is its boundary, and $\hat{\mathbf{n}}$ is the outward pointing unit normal. To discretize in space, define $\mathcal{V}_h^p$ to be the space of discontinuous vector-valued polynomials of degree p on a subdivision T_h of the domain into non-overlapping elements such that $\Omega = \bigcup_{\kappa \in T_h} \kappa$. Thus, the solution and test function space is defined by

$$\mathcal{V}_h^p = \{\mathbf{v} \in L^2(\Omega) \mid \mathbf{v}|_\kappa \in P^p, \, \forall \kappa \in T_h\}, \tag{2.2}$$

where P^p is the space of polynomial functions of degree at most p. The discrete problem then takes the following form: find $\mathbf{u}_h \in \mathcal{V}_h^p$ such that $\forall \mathbf{v}_h \in \mathcal{V}_h^p$,

$$\begin{aligned} \sum_{\kappa \in T_h} \Big\{ & \int_\kappa \mathbf{v}_h^T (\mathbf{u}_h)_t \, d\mathbf{x} - \int_\kappa \nabla \mathbf{v}_h^T \cdot \mathcal{F}_i \, d\mathbf{x} \\ & + \int_{\partial\kappa \backslash \partial\Omega} \mathbf{v}_h^{+T} \mathcal{H}_i(\mathbf{u}_h^+, \mathbf{u}_h^-, \hat{\mathbf{n}}) \, ds + \int_{\partial\kappa \cap \partial\Omega} \mathbf{v}_h^{+T} \mathcal{H}_i^b(\mathbf{u}_h^+, \mathbf{u}_h^b, \hat{\mathbf{n}}) \, ds \Big\} = 0, \end{aligned} \tag{2.3}$$

where $\mathcal{H}_i(\mathbf{u}_h^+, \mathbf{u}_h^-, \hat{\mathbf{n}})$ and $\mathcal{H}_i^b(\mathbf{u}_h^+, \mathbf{u}_h^b, \hat{\mathbf{n}})$ are numerical flux functions defined for interior and boundary faces, respectively. In this work, the Roe-averaged flux [46] is used for the inviscid numerical flux on interior faces. The boundary conditions are imposed weakly by constructing an exterior boundary state, $\mathbf{u}_h^b$, which is a function of the interior state and known boundary data. Boundary conditions are discussed in more detail in Section 2.4.2. Furthermore, the $(\cdot)^+$ and $(\cdot)^-$ notation is used to indicate the trace value taken from the interior and exterior of the element, respectively.

2.2 DG for Navier-Stokes

The compressible, two-dimensional Navier-Stokes equations in strong, conservation form are

$$\mathbf{u}_t + \nabla \cdot \mathcal{F}_i(\mathbf{u}) - \nabla \cdot \mathcal{F}_v(\mathbf{u}, \nabla \mathbf{u}) = 0, \tag{2.4}$$

where the conservative state, $\mathbf{u}$, and inviscid flux vector, $\mathcal{F}_i$, are defined in Section 2.1. The viscous flux, $\mathcal{F}_v = \mathcal{A}_v \nabla \mathbf{u} = (\mathbf{F}_v^x, \mathbf{F}_v^y)$, is given by

$$\begin{aligned}
\mathbf{F}_v^x &= \begin{pmatrix} 0 \\ \frac{2}{3}\mu(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}) \\ \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \\ \frac{2}{3}\mu(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y})u + \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})v + \kappa\frac{\partial T}{\partial x} \end{pmatrix}, \\
\mathbf{F}_v^y &= \begin{pmatrix} 0 \\ \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \\ \frac{2}{3}\mu(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}) \\ \frac{2}{3}\mu(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x})v + \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})u + \kappa\frac{\partial T}{\partial y} \end{pmatrix},
\end{aligned}$$

where μ is the dynamic viscosity and κ is the thermal conductivity.

2.2.1 Flux Formulation

The first step in the flux formulation is to rewrite Eqn. 2.4 as a first-order system. To accomplish this reduction of order, a new variable, $\mathcal{Q} = \mathcal{A}_v \nabla \mathbf{u}$, is defined. Thus, Eqn. 2.4 can be written as a first-order system in terms of $\mathcal{Q}$:

$$\begin{aligned}
\mathbf{u}_t + \nabla \cdot \mathcal{F}_i - \nabla \cdot \mathcal{Q} &= 0, \\
\mathcal{Q} - \mathcal{A}_v \nabla \mathbf{u} &= 0. \qquad (2.5)
\end{aligned}$$

Multiplying Eqns. 2.5 by test functions $\mathbf{v}$ and $\boldsymbol{\tau}$, respectively, and integrating by parts gives the weak formulation:

$$\begin{aligned}
\int_\Omega \mathbf{v}^T \mathbf{u}_t \, d\mathbf{x} &+ \int_{\partial\Omega} \mathbf{v}^T \mathcal{F}_i \cdot \hat{\mathbf{n}} \, ds - \int_\Omega \nabla \mathbf{v}^T \cdot \mathcal{F}_i \, d\mathbf{x} \\
&- \int_{\partial\Omega} \mathbf{v}^T \mathcal{Q} \cdot \hat{\mathbf{n}} \, ds + \int_\Omega \nabla \mathbf{v}^T \cdot \mathcal{Q} \, d\mathbf{x} = 0, \quad \forall \mathbf{v} \in H^1(\Omega), \\
\int_\Omega \boldsymbol{\tau}^T \cdot \mathcal{Q} \, d\mathbf{x} &- \int_{\partial\Omega} \mathbf{u}^T (\mathcal{A}_v^T \boldsymbol{\tau}) \cdot \hat{\mathbf{n}} \, ds \\
&+ \int_\Omega \mathbf{u}^T \nabla \cdot (\mathcal{A}_v^T \boldsymbol{\tau}) \, d\mathbf{x} = 0, \quad \forall \boldsymbol{\tau} \in [H^1(\Omega)]^2.
\end{aligned}$$

Then, using the space $\mathcal{V}_h^p$ and triangularization T_h defined in Section 2.1, the spatial discretization of Eqn. 2.4 takes the following form: find $\mathbf{u}_h \in \mathcal{V}_h^p$ and $\mathcal{Q}_h \in [\mathcal{V}_h^p]^2$ such that

$\forall \mathbf{v}_h \in \mathcal{V}_h^p$ and $\forall \boldsymbol{\tau}_h \in [\mathcal{V}_h^p]^2$,

$$\begin{aligned}
&\sum_{\kappa \in T_h} \Big\{ \int_\kappa \mathbf{v}_h^T (\mathbf{u}_h)_t \, d\mathbf{x} - \int_\kappa \nabla \mathbf{v}_h^T \cdot \mathcal{F}_i \, d\mathbf{x} + \int_{\partial\kappa \backslash \partial\Omega} \mathbf{v}_h^{+T} \mathcal{H}_i(\mathbf{u}_h^+, \mathbf{u}_h^-, \hat{\mathbf{n}}) \, ds \\
&\qquad + \int_{\partial\kappa \cap \partial\Omega} \mathbf{v}_h^{+T} \mathcal{H}_i^b(\mathbf{u}_h^+, \mathbf{u}_h^b, \hat{\mathbf{n}}) \, ds \Big\} - \sum_{\kappa \in T_h} \Big\{ \int_{\partial\kappa \backslash \partial\Omega} \mathbf{v}_h^{+T} \mathcal{H}_v(\mathcal{Q}_h^+, \mathcal{Q}_h^-) \cdot \hat{\mathbf{n}} \, ds \\
&\qquad + \int_{\partial\kappa \cap \partial\Omega} \mathbf{v}_h^{+T} \mathcal{H}_v^b(\mathcal{Q}_h^+, \mathcal{Q}_h^b) \cdot \hat{\mathbf{n}} \, ds - \int_\kappa \nabla \mathbf{v}_h^T \cdot \mathcal{Q}_h \, d\mathbf{x} \Big\} = 0, \\
&\sum_{\kappa \in T_h} \Big\{ \int_\kappa \boldsymbol{\tau}_h^T \cdot \mathcal{Q}_h \, d\mathbf{x} - \int_{\partial\kappa \backslash \partial\Omega} \mathbf{h}_u(\mathbf{u}_h^+, \mathbf{u}_h^-)^T (\mathcal{A}_v^T \boldsymbol{\tau}_h)^+ \cdot \hat{\mathbf{n}} \, ds \\
&\qquad - \int_{\partial\kappa \cap \partial\Omega} \mathbf{u}_h^{b\,T} (\mathcal{A}_v^T \boldsymbol{\tau}_h)^+ \cdot \hat{\mathbf{n}} \, ds + \int_\kappa \mathbf{u}_h^T \nabla \cdot (\mathcal{A}_v^T \boldsymbol{\tau}_h) \, d\mathbf{x} \Big\} = 0, \qquad (2.6)
\end{aligned}$$

where $\mathbf{h}_u$ and $\mathbf{u}_h^b$ are numerical fluxes approximating $\mathbf{u}$, and $\mathcal{H}_v$ and $\mathcal{H}_v^b$ are numerical fluxes approximating $\mathcal{A}_v \nabla \mathbf{u}$ on interior and boundary faces, respectively. Thus, given definitions of these numerical fluxes, Eqn. 2.6 gives the semi-discrete flux form of the Navier-Stokes equations.

Note that the first summation over κ in Eqn. 2.6 is exactly the same as that in the Euler discretization given by Eqn. 2.3. These terms are not modified by the addition of the viscous discretization; thus, from this point forward, they are denoted simply as $\mathbb{E}$.

2.2.2 Primal Formulation

To simplify the notation of the primal form, jump, $[\![\cdot]\!]$, and average, $\{\cdot\}$, operators are defined for interior faces. For spatially scalar quantities, the operators are given by

$$\begin{aligned}
[\![s]\!] &= s^+ \hat{\mathbf{n}}^+ + s^- \hat{\mathbf{n}}^-, \\
\{s\} &= \frac{1}{2}(s^+ + s^-),
\end{aligned}$$

where $(\cdot)^+$ and $(\cdot)^-$ refer to trace values taken from opposite sides of the face. Note that the unit normals $\hat{\mathbf{n}}^+$ and $\hat{\mathbf{n}}^-$ are outward pointing with reference to the sides $(\cdot)^+$ and $(\cdot)^-$, and, thus, $\hat{\mathbf{n}}^+ = -\hat{\mathbf{n}}^-$. Furthermore, for spatially vector quantities,

$$\begin{aligned}
[\![\boldsymbol{\varphi}]\!] &= \boldsymbol{\varphi}^+ \cdot \hat{\mathbf{n}}^+ + \boldsymbol{\varphi}^- \cdot \hat{\mathbf{n}}^-, \\
\{\boldsymbol{\varphi}\} &= \frac{1}{2}(\boldsymbol{\varphi}^+ + \boldsymbol{\varphi}^-).
\end{aligned}$$

Via substitution, one can show that,

$$\sum_{\kappa\in T_h}\int_{\partial\kappa\backslash\partial\Omega}\mathbf{s}^{+T}\boldsymbol{\varphi}^{+}\cdot\hat{\mathbf{n}}\,ds=\int_{\Gamma_i}[\![\mathbf{s}]\!]^T\cdot\{\boldsymbol{\varphi}\}\,ds+\int_{\Gamma_i}\{\mathbf{s}\}^T[\![\boldsymbol{\varphi}]\!]\,ds, \tag{2.7}$$

where Γ_i is the union of all interior faces. Applying Eqn. 2.7, Eqns. 2.6 become

$$\begin{aligned}
&\mathbb{E}+\sum_{\kappa\in T_h}\Big[\int_\kappa\nabla\mathbf{v}_h^T\cdot\mathcal{Q}_h\,d\mathbf{x}\Big]-\int_{\Gamma_i}[\![\mathbf{v}_h]\!]^T\cdot\{\mathcal{H}_v\}\,ds\\
&\qquad-\int_{\Gamma_i}\{\mathbf{v}_h\}^T[\![\mathcal{H}_v]\!]\,ds-\int_{\partial\Omega}\mathbf{v}_h^{+T}\mathcal{H}_v^b\cdot\hat{\mathbf{n}}\,ds=0,\\
&\sum_{\kappa\in T_h}\Big[\int_\kappa\boldsymbol{\tau}_h^T\cdot\mathcal{Q}_h\,d\mathbf{x}+\int_\kappa\mathbf{u}_h^T\nabla\cdot(\mathcal{A}_v^T\boldsymbol{\tau}_h)\,d\mathbf{x}\Big]-\int_{\Gamma_i}[\![\mathbf{h}_u]\!]^T\cdot\{\mathcal{A}_v^T\boldsymbol{\tau}_h\}\,ds\\
&\qquad-\int_{\Gamma_i}\{\mathbf{h}_u\}^T[\![\mathcal{A}_v^T\boldsymbol{\tau}_h]\!]\,ds-\int_{\partial\Omega}\mathbf{u}_h^{b\,T}(\mathcal{A}_v^T\boldsymbol{\tau}_h)^+\cdot\hat{\mathbf{n}}\,ds=0.
\end{aligned}\tag{2.8}$$

Integrating by parts and using Eqn. 2.7, the term $\sum_{\kappa\in T_h}\int_\kappa{\mathbf{u}_h}^T\nabla\cdot(\mathcal{A}_v^T\boldsymbol{\tau}_h)\,d\mathbf{x}$ in Eqn 2.8 can be rewritten:

$$\begin{aligned}
\sum_{\kappa\in T_h}\int_\kappa\mathbf{u}_h^T\nabla\cdot(\mathcal{A}_v^T\boldsymbol{\tau}_h)\,d\mathbf{x}\;&=\;\int_{\Gamma_i}\Big([\![\mathbf{u}_h]\!]^T\cdot\{\mathcal{A}_v^T\boldsymbol{\tau}_h\}+\{\mathbf{u}_h\}^T[\![\mathcal{A}_v^T\boldsymbol{\tau}_h]\!]\Big)\,ds\\
&\quad+\int_{\partial\Omega}\mathbf{u}_h^{+T}(\mathcal{A}_v^T\boldsymbol{\tau}_h)^+\cdot\hat{\mathbf{n}}\,ds\\
&\quad-\sum_{\kappa\in T_h}\int_\kappa\boldsymbol{\tau}_h^T\cdot(\mathcal{A}_v\nabla\mathbf{u}_h)\,d\mathbf{x}.
\end{aligned}\tag{2.9}$$

Then, substituting Eqn. 2.9 into the second of Eqns. 2.8 yields

$$\begin{aligned}
&\sum_{\kappa\in T_h}\Big\{\int_\kappa\boldsymbol{\tau}_h^T\cdot\mathcal{Q}_h\,d\mathbf{x}-\int_\kappa\boldsymbol{\tau}_h^T\cdot(\mathcal{A}_v\nabla\mathbf{u}_h)\,d\mathbf{x}\Big\}\\
&\qquad-\int_{\Gamma_i}\Big([\![\mathbf{h}_u-\mathbf{u}_h]\!]^T\cdot\{\mathcal{A}_v^T\boldsymbol{\tau}_h\}+\{\mathbf{h}_u-\mathbf{u}_h\}^T[\![\mathcal{A}_v^T\boldsymbol{\tau}_h]\!]\Big)\,ds\\
&\qquad-\int_{\partial\Omega}(\mathbf{u}_h^b-\mathbf{u}_h^+)^T(\mathcal{A}_v^T\boldsymbol{\tau}_h)^+\cdot\hat{\mathbf{n}}\,ds=0.
\end{aligned}$$

Finally, defining lifting operators $\boldsymbol{\delta}$ and $\boldsymbol{\delta}^i$ as

$$\begin{aligned}
\sum_{\kappa\in T_h}\int_\kappa\boldsymbol{\tau}_h^T\cdot\boldsymbol{\delta}\,d\mathbf{x}\;&=\;-\int_{\Gamma_i}[\![\mathbf{h}_u-\mathbf{u}_h]\!]^T\cdot\{\mathcal{A}_v^T\boldsymbol{\tau}_h\}\,ds-\int_{\partial\Omega}(\mathbf{u}_h^b-\mathbf{u}_h^+)^T(\mathcal{A}_v^T\boldsymbol{\tau}_h)^+\cdot\hat{\mathbf{n}}\,ds,\\
\sum_{\kappa\in T_h}\int_\kappa\boldsymbol{\tau}_h^T\cdot\boldsymbol{\delta}^i\,d\mathbf{x}\;&=\;-\int_{\Gamma_i}\{\mathbf{h}_u-\mathbf{u}_h\}^T[\![\mathcal{A}_v^T\boldsymbol{\tau}_h]\!]\,ds,
\end{aligned}$$

one can write $\mathcal{Q}_h$ in terms of $\mathbf{u}_h$, $\boldsymbol{\delta}$, and $\boldsymbol{\delta}^i$:

$$\mathcal{Q}_h = \mathcal{A}_v \nabla \mathbf{u}_h - \boldsymbol{\delta} - \boldsymbol{\delta}^i.$$

Thus, taking $\boldsymbol{\tau}_h = \nabla \mathbf{v}_h$, the primal form is given by the following: find $\mathbf{u}_h \in \mathcal{V}_h^p$ such that $\forall \mathbf{v}_h \in \mathcal{V}_h^p$,

$$\begin{aligned}
\mathbb{E} \quad &+ \sum_{\kappa \in T_h} \int_\kappa \nabla \mathbf{v}_h^T \cdot (\mathcal{A}_v \nabla \mathbf{u}_h)\, d\mathbf{x} + \int_{\Gamma_i} \Big([\![\mathbf{h}_u - \mathbf{u}_h]\!]^T \cdot \{\mathcal{A}_v^T \nabla \mathbf{v}_h\} - [\![\mathbf{v}_h]\!]^T \cdot \{\mathcal{H}_v\} \Big)\, ds \\
&+ \int_{\Gamma_i} \Big(\{\mathbf{h}_u - \mathbf{u}_h\}^T [\![\mathcal{A}_v^T \nabla \mathbf{v}_h]\!] - \{\mathbf{v}_h\}^T [\![\mathcal{H}_v]\!] \Big)\, ds \\
&+ \int_{\partial\Omega} \Big[(\mathbf{u}_h^b - \mathbf{u}_h^+)^T (\mathcal{A}_v^T \nabla \mathbf{v}_h)^+ \cdot \hat{\mathbf{n}} - \mathbf{v}_h^{+T} \mathcal{H}_v^b \cdot \hat{\mathbf{n}} \Big]\, ds = 0. \qquad (2.10)
\end{aligned}$$

Again, to fully define the discretization, one must define the numerical fluxes $\mathbf{h}_u(\mathbf{u}_h^+, \mathbf{u}_h^-)$ and $\mathcal{H}_v(\mathcal{Q}_h^+, \mathcal{Q}_h^-)$ for interior faces and $\mathbf{u}_h^b$ and $\mathcal{H}_v^b$ for boundary faces. These definitions are considered in Section 2.2.3.

2.2.3 Bassi and Rebay Discretization

Motivated by the lack of any upwinding mechanism in the viscous terms of the Navier-Stokes equations, one might consider using central fluxes for $\mathbf{h}_u$ and $\mathcal{H}_v$:

$$\begin{aligned}
\mathbf{h}_u &= \{\mathbf{u}_h\}, \\
\mathcal{H}_v &= \{\mathcal{Q}_h\}.
\end{aligned}$$

These fluxes were originally developed and applied by Bassi and Rebay [8] with moderate success. However, they are used here only to motivate the final flux choice.

Clearly, both fluxes are single-valued on a given face, thus,

$$\begin{aligned}
\{\mathcal{H}_v\} &= \mathcal{H}_v, \\
[\![\mathcal{H}_v]\!] &= 0,
\end{aligned}$$

and

$$\begin{aligned}
\{\mathbf{h}_u - \mathbf{u}_h\} &= 0, \\
[\![\mathbf{h}_u - \mathbf{u}_h]\!] &= -[\![\mathbf{u}_h]\!].
\end{aligned}$$

To complete the interior face flux definition, note that

$$\mathcal{H}_v = \{\mathcal{Q}_h\} = \{\mathcal{A}_v \nabla \mathbf{u}_h - \boldsymbol{\delta} - \boldsymbol{\delta}^i\}.$$

However,

$$\sum_{\kappa \in T_h} \int_\kappa \boldsymbol{\tau}_h^T \cdot \boldsymbol{\delta}^i d\mathbf{x} = -\int_{\Gamma_i} \{\mathbf{h}_u - \mathbf{u}_h\}^T [\![\mathcal{A}_v^T \boldsymbol{\tau}_h]\!] ds = 0.$$

Thus,

$$\mathcal{H}_v = \{\mathcal{A}_v \nabla \mathbf{u}_h\} - \{\boldsymbol{\delta}\}.$$

Finally, boundary conditions are of the form

$$\begin{aligned} \mathbf{u}_h^b &= \mathbf{u}_h^b(\mathbf{u}_h^+, \text{BC Data}), \\ \mathcal{H}_v^b &= \mathcal{H}_v^b(\mathbf{u}_h^+, \nabla \mathbf{u}_h^+, \text{BC Data}) = (\mathcal{A}_v \nabla \mathbf{u}_h)^b - \boldsymbol{\delta}^b. \end{aligned}$$

Thus, the final form of the discretization is as follows: find $\mathbf{u}_h \in \mathcal{V}_h^p$ such that $\forall \mathbf{v}_h \in \mathcal{V}_h^p$,

$$\begin{aligned} \mathbb{E} &+ \sum_{\kappa \in T_h} \int_\kappa \nabla \mathbf{v}_h^T \cdot (\mathcal{A}_v \nabla \mathbf{u}_h)\, d\mathbf{x} - \int_{\Gamma_i} \Big([\![\mathbf{u}_h]\!]^T \cdot \{\mathcal{A}_v^T \nabla \mathbf{v}_h\} + [\![\mathbf{v}_h]\!]^T \cdot \{\mathcal{A}_v \nabla \mathbf{u}_h\} \Big)\, ds \\ &+ \int_{\Gamma_i} [\![\mathbf{v}_h]\!]^T \cdot \{\boldsymbol{\delta}\}\, ds + \int_{\partial\Omega} (\mathbf{u}_h^b - \mathbf{u}_h^+)^T (\mathcal{A}_v^T \nabla \mathbf{v}_h)^+ \cdot \hat{\mathbf{n}}\, ds \\ &- \int_{\partial\Omega} \mathbf{v}_h^{+T} (\mathcal{A}_v \nabla \mathbf{u}_h)^b \cdot \hat{\mathbf{n}}\, ds + \int_{\partial\Omega} \mathbf{v}_h^{+T} \boldsymbol{\delta}^b \cdot \hat{\mathbf{n}}\, ds = 0. \end{aligned}$$

Unfortunately, as shown by numerous authors [20, 21, 42], this discretization is problematic for multiple reasons. First, on some meshes the viscous contribution to the Jacobian may be singular. While the inviscid terms should make the discrete system non-singular, this result is clearly undesirable. Second, stability and optimal order of accuracy in L^2 cannot be proven [21]. In fact, Cockburn and Shu [20] have shown that for purely elliptic problems, odd order interpolants produce sub-optimal order of accuracy equal to $O(h^p)$. Finally, the scheme is not compact. This non-compactness is introduced through the global lifting operator $\boldsymbol{\delta}$, as illustrated in 1-D by Figure 2-1. The figure shows that, while the residual on element k, R_k, depends explicitly on only $\mathbf{u}_h$ and $\boldsymbol{\delta}$ on the element and its neighbors (elements $k-1$, k, and $k+1$), the global lifting operators on the neighboring elements also depend on their neighbors. Thus, the dependence of R_k is extended to elements $k-2$ and $k+2$ if the additional degrees of freedom for $\boldsymbol{\delta}$ are eliminated.

Given the drawbacks of the previous scheme, a modification of this scheme which will make it stable, optimally convergent, and compact is desired. Bassi and Rebay [11, 10] have proposed modified numerical fluxes which accomplish these goals by replacing the global

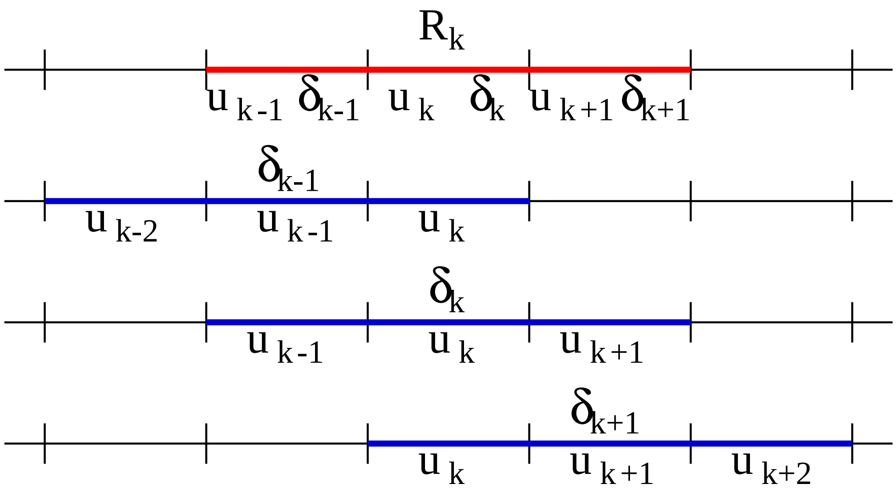


Figure 2-1: 1-D stencil for first Bassi and Rebay scheme

lifting operator with a local lifting operator. The local lifting operator, or auxiliary variable, $\boldsymbol{\delta}_f$, is defined by the following problem: find $\boldsymbol{\delta}_f \in [\mathcal{V}_h^p]^2$ such that $\forall \boldsymbol{\tau}_h \in [\mathcal{V}_h^p]^2$,

$$\int_{\kappa^\pm} \boldsymbol{\tau}_h^T \cdot \boldsymbol{\delta}_f^\pm \, d\mathbf{x} = \int_{\sigma_f} [\![\mathbf{u}_h]\!]^T \cdot \{\mathcal{A}_v^T \boldsymbol{\tau}_h\} \, ds \tag{2.11}$$

for interior faces, and

$$\int_{\kappa^+} \boldsymbol{\tau}_h^T \cdot \boldsymbol{\delta}_f^b d\mathbf{x} = -\int_{\sigma_f^b} (\mathbf{u}_h^b - \mathbf{u}_h^+)^T [(\mathcal{A}_v^T \boldsymbol{\tau}_h) \cdot \hat{\mathbf{n}}]^+ \, ds, \tag{2.12}$$

for boundary faces, where σ_f denotes a single interior face and σ_f^b denotes a single boundary face.

Replacing the global lifting operator, $\boldsymbol{\delta}$, with the local lifting operator, $\boldsymbol{\delta}_f$, and multiplying by a stabilization parameter, η_f, which is discussed in Section 2.3, the numerical fluxes $\mathcal{H}_v$ and $\mathcal{H}_v^b$ become

$$\begin{aligned} \mathcal{H}_v &= \{\mathcal{A}_v \nabla \mathbf{u}_h\} - \eta_f \{\boldsymbol{\delta}_f\}, \\ \mathcal{H}_v^b &= (\mathcal{A}_v \nabla \mathbf{u}_h)^b - \eta_f \boldsymbol{\delta}_f^b. \end{aligned}$$

The $\mathbf{h}_u$ and $\mathbf{u}_h^b$ numerical fluxes are not modified. Thus, substituting the new numerical fluxes into Eqn. 2.10 gives the second Bassi and Rebay discretization: find $\mathbf{u}_h \in \mathcal{V}_h^p$ such

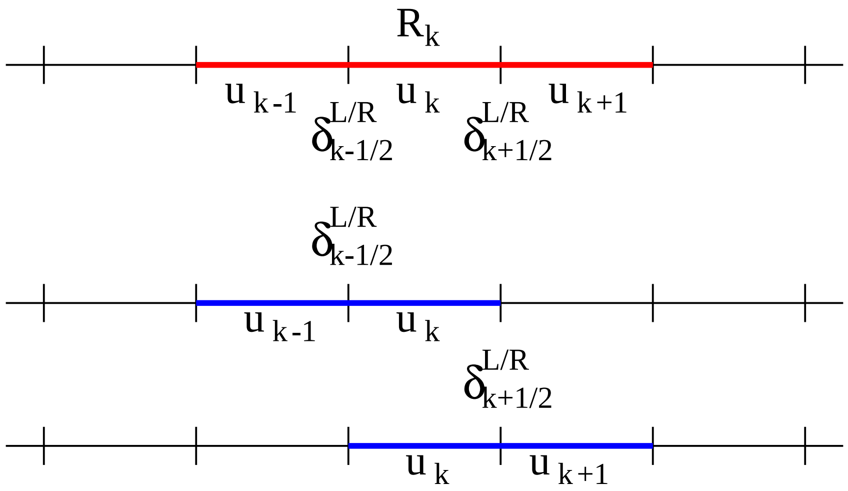


Figure 2-2: 1-D stencil for BR2 scheme

that $\forall \mathbf{v}_h \in \mathcal{V}_h^p$,

$$\begin{aligned}
\mathbb{E} \quad &+ \sum_{\kappa\in T_h}\int_\kappa \nabla \mathbf{v}_h^T\cdot(\mathcal{A}_v\nabla\mathbf{u}_h)\,d\mathbf{x} - \int_{\Gamma_i}\Big([\![\mathbf{u}_h]\!]^T\cdot\{\mathcal{A}_v^T\nabla\mathbf{v}_h\} + [\![\mathbf{v}_h]\!]^T\cdot\{\mathcal{A}_v\nabla\mathbf{u}_h\}\Big)\,ds \\
&+ \int_{\Gamma_i}[\![\mathbf{v}_h]\!]^T\cdot\eta_f\{\boldsymbol{\delta}_f\}\,ds + \int_{\partial\Omega}(\mathbf{u}^b - \mathbf{u}_h^+)^T(\mathcal{A}_v^T\nabla\mathbf{v}_h)^+\cdot\hat{\mathbf{n}}\,ds \\
&- \int_{\partial\Omega}\mathbf{v}_h^{+T}(\mathcal{A}_v\nabla\mathbf{u}_h)^b\cdot\hat{\mathbf{n}}\,ds + \int_{\partial\Omega}\mathbf{v}_h^{+T}\eta_f\boldsymbol{\delta}_f^b\cdot\hat{\mathbf{n}}\,ds = 0. \qquad (2.13)
\end{aligned}$$

Note that, unlike the original choice of fluxes, the BR2 form has a compact stencil, meaning that only elements that share a face are coupled. Figure 2-2 shows the stencil of the BR2 scheme in 1-D. It shows that the compact stencil follows from the fact that the local lifting operators, $\boldsymbol{\delta}_f$, at a given face depend only on the elements that share that face.

Furthermore, for purely elliptic problems with homogeneous Dirichlet boundary conditions, Arnold *et al.* [21] proves optimal error convergence in the L^2 norm for $p \geq 1$ when $\eta_f > 3$ (note: the condition $\eta_f > 3$ is required to prove stability). However, in Section 2.3 we propose a definition of η_f which is generally less than this value but is required to produce optimal accuracy for $p = 0$; in practice, we have not found that this lower value of η_f causes a loss of stability.

While the BR2 discretization is certainly not the only scheme proposed in the literature which achieves optimal order of accuracy, it has some clear advantages. Other schemes that achieve optimal order of accuracy include the local discontinuous Galerkin (LDG) method [19, 20], a penalty method proposed by Brezzi [42], and the Baumann and Oden scheme [13]. However, each of these has some drawbacks. LDG is not compact on general, unstructured meshes. The stabilization parameter required for optimal accuracy in the Brezzi scheme can grow very large at high p because $\eta_f \approx h^{-2p}$, and the Baumann and Oden scheme is only stable for $p \geq 2$. In fact, the BR2 scheme is the only one proposed in the literature to achieve optimal order of accuracy for all $p \geq 1$ with a compact stencil.

The final discrete form is completed by choosing a basis and time discretization, as shown in Section 2.5.

2.3 The Stabilization Parameter, η_f

As discussed in Section 2.2.3, $\eta_f > 3$ is required to prove stability for the BR2 scheme. For $p \geq 1$, there are no other conditions on the choice of η_f. However, for $p = 0$, if η_f is not set appropriately, the error may not converge with h. A convergent $p = 0$ discretization is desired such that it may be used as the coarsest order in the p-multigrid algorithm. This section shows the derivation of the choice of η_f used in this work.

2.3.1 Motivation

To motivate the derivation of a non-constant η_f, results from the solution of Poisson's equation using $\eta_f = 3$ are shown. Thus, the problem is

$$\begin{aligned} -\nabla \cdot (\nabla u) &= f \quad \text{in} \quad \Omega \\ u &= g \quad \text{on} \quad \partial\Omega, \end{aligned} \tag{2.14}$$

where

$$\begin{aligned} f &= -6(x+y), \\ g &= x^3 + y^3. \end{aligned}$$

For brevity, the BR2 discretization of this problem is not given here. However, it is given for $g = 0$ in Eqn. 2.15.

To conduct a mesh refinement study, the domain, a unit square whose lower left corner lies on the origin, is triangulated into grids of 44, 176, and 704 elements. Results of the study are shown in Figure 2-3. For $p \geq 1$, the L_2-norm of the solution error converges optimally at a rate of $O(h^{p+1})$. However, for $p = 0$, the L_2-norm of the error is approximately constant with grid refinement. This behavior results from the fact that for $p = 0$, only the auxiliary variable terms remain on the left hand side of the BR2 form. If η_f is not set appropriately, the $p = 0$ discretization is not consistent. Thus, a definition of η_f that makes

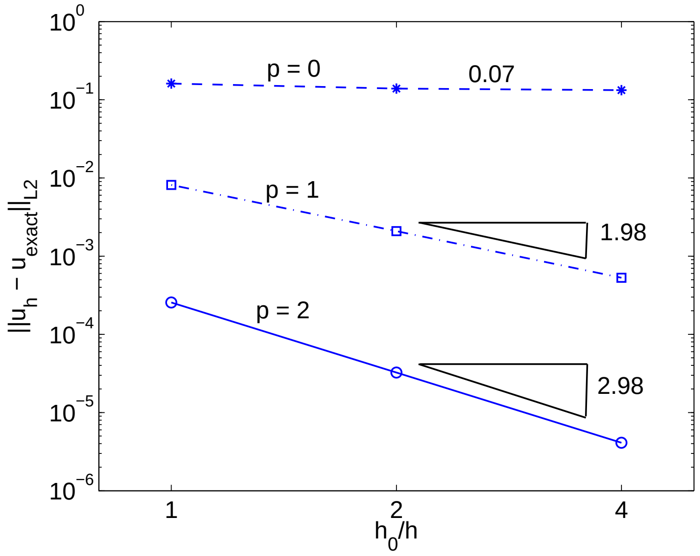


Figure 2-3: Mesh refinement results for $0 \le p \le 2$ using $\eta_f = 3$

the discretization consistent is sought.

2.3.2 Formulation

For simplicity, the analysis is shown for Poisson's equation with homogeneous Dirichlet boundary conditions. Thus, the problem is given by Eqn. 2.14 with $g = 0$. For this case, the BR2 discretization is given by Eqn. 2.15: find $u_h \in \mathcal{V}_h^p$ such that $\forall v_h \in \mathcal{V}_h^p$,

$$\begin{aligned}\sum_{\kappa \in T_h} \nabla v_h \cdot \nabla u_h \, d\mathbf{x} - \int_{\Gamma_i} \Big([\![u_h]\!] \cdot \{\nabla v_h\} + [\![v_h]\!] \cdot \{\nabla u_h\} \Big) \, ds \\ + \int_{\Gamma_i} \eta_f [\![v_h]\!] \cdot \{\boldsymbol{\delta}_f\} \, ds + \int_{\partial\Omega} -u_h^+ \nabla v_h^+ \cdot \hat{\mathbf{n}} \, ds \\ - \int_{\partial\Omega} v_h^+ \nabla u_h^+ \cdot \hat{\mathbf{n}} \, ds + \int_{\partial\Omega} \eta_f v_h^+ \boldsymbol{\delta}_f^b \cdot \hat{\mathbf{n}} \, ds = \sum_{\kappa \in T_h} \int_\kappa v_h f \, d\mathbf{x},\end{aligned} \tag{2.15}$$

where $\mathcal{V}_h^p$ is now the space of discontinuous scalar-valued polynomials of degree p on a subdivision T_h of the domain into non-overlapping elements such that $\Omega = \bigcup_{\kappa \in T_h} \kappa$.

For $p = 0$, $\nabla u_h = \nabla v_h = 0$. Thus, Eqn. 2.15 becomes

$$\int_{\Gamma_i} \eta_f [\![v_h]\!] \cdot \{\boldsymbol{\delta}_f\} \, ds + \int_{\partial\Omega} \eta_f v_h^+ \boldsymbol{\delta}_f^b \cdot \hat{\mathbf{n}} \, ds = \sum_{\kappa \in T_h} \int_\kappa v_h f \, d\mathbf{x}. \tag{2.16}$$

Examining the test function $v_{h,i}$ associated with element κ_i,

$$v_{h,i}(\mathbf{x}) = \begin{cases} 1 & \text{if} \quad \mathbf{x} \in \kappa_i \\ 0 & \text{if} \quad \mathbf{x} \notin \kappa_i, \end{cases}$$

Eqn. 2.16 reduces to

$$\frac{1}{2} \sum_{\sigma \in \partial\kappa_i} \int_\sigma \eta_f [(\boldsymbol{\delta}_f \cdot \hat{\mathbf{n}})^+ - (\boldsymbol{\delta}_f \cdot \hat{\mathbf{n}})^-] \, ds = \int_{\kappa_i} f \, d\mathbf{x}, \tag{2.17}$$

where κ_i has no faces on the domain boundary. Using the definition of the auxiliary variable given in Eqn. 2.11, a $p = 0$ basis for $\boldsymbol{\tau}_h$ yields

$$(\boldsymbol{\delta}_f \cdot \hat{\mathbf{n}})^+ = \frac{1}{2} \frac{1}{A_\sigma^+} (u_h^+ - u_h^-) s_\sigma, \tag{2.18}$$

$$(\boldsymbol{\delta}_f \cdot \hat{\mathbf{n}})^- = -\frac{1}{2} \frac{1}{A_\sigma^-} (u_h^+ - u_h^-) s_\sigma, \tag{2.19}$$

on a given edge, $\sigma \in \partial\kappa_i$, of length s_σ. A_σ^+ and A_σ^- are the areas of the elements adjoining edge σ. Substituting Eqns. 2.18 and 2.19 into Eqn. 2.17 gives

$$\sum_{\sigma \in \partial\kappa_i} \left[\frac{1}{4} \eta_f s_\sigma^2 (u_h^+ - u_h^-) \left(\frac{1}{A_\sigma^+} + \frac{1}{A_\sigma^-} \right) \right] = \int_{\kappa_i} f d\mathbf{x}. \tag{2.20}$$

To motivate the definition of η_f for $p = 0$, apply the Divergence theorem to Poisson's equation,

$$\int_{\kappa_i} f \, d\mathbf{x} = -\int_{\kappa_i} \nabla \cdot (\nabla u) \, d\mathbf{x} = -\int_{\partial\kappa_i} \nabla u \cdot \hat{\mathbf{n}} \, ds.$$

For each edge, define an approximation to the directional derivative in the $\hat{\mathbf{n}}$ direction to be

$$\nabla u \cdot \hat{\mathbf{n}} = \frac{u_h^- - u_h^+}{\Delta n},$$

where Δn is the distance between the + and − element centroids projected onto the $\hat{\mathbf{n}}$ direction. For triangles,

$$\Delta n = \frac{1}{3}(h_\sigma^+ + h_\sigma^-),$$

where $h_\sigma^\pm$ denotes the element height for elements adjacent to face σ, as shown in Figure 2-4. Thus,

$$\int_{\kappa_i} f \, d\mathbf{x} = -\int_{\partial\kappa_i} \nabla u \cdot \hat{\mathbf{n}} \, ds = \sum_{\sigma \in \partial\kappa_i} \left[\frac{u_h^+ - u_h^-}{\Delta n} \right]_\sigma s_\sigma. \tag{2.21}$$

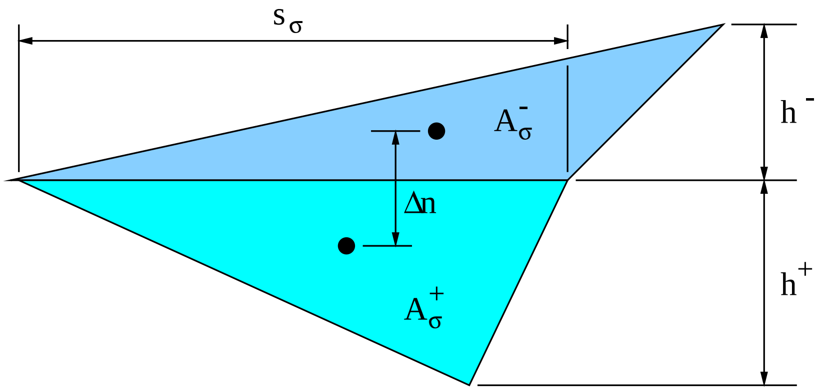


Figure 2-4: Definitions of h^+, h^-, and Δn

Then, substituting Eqn. 2.21 into Eqn. 2.20 and solving for η_f yields

$$\eta_f = \frac{4}{s_\sigma \Delta n_\sigma} \frac{A_\sigma^+ A_\sigma^-}{A_\sigma^+ + A_\sigma^-}. \tag{2.22}$$

Eqn. 2.22 is valid for general elements. For triangles, $A_\sigma^\pm = \frac{1}{2} h_\sigma^\pm s_\sigma$, thus,

$$\eta_f = \frac{3}{1 + \frac{1}{2}\left(\frac{h_\sigma^+}{h_\sigma^-} + \frac{h_\sigma^-}{h_\sigma^+}\right)}.$$

An analogous procedure shows that, for triangular elements with straight edges, on the boundary, $\eta_f = 3/2$ always.

Thus, using this definition, $\eta_f \leq 3$ for every face. This result appears problematic given that the stability proof requires $\eta_f > 3$. However, in practice, no problems have arisen. Results for the case examined in Section 2.3.1 are given in Figure 2-5. The figure shows that the L_2 norm of the solution is converging at $O(h^{p+1})$ for $0 \leq p \leq 2$. Furthermore, for compressible Navier-Stokes, Bassi and Rebay [11, 10] successfully applied the discretization with $\eta_f = 1$ for $p \geq 1$, and it is used here with η_f as given in Eqn. 2.22 without detrimental effects. This fact is shown by the accuracy studies presented in Chapter 5.

2.4 Boundary Treatment

2.4.1 Geometry Representation: Curved Boundaries

As shown by Bassi and Rebay [9], high-order DG methods are highly sensitive to the geometry representation. Thus, it is necessary to build a higher-order representation of the domain boundary. In this work, the geometry is represented using a nodal Lagrange basis.

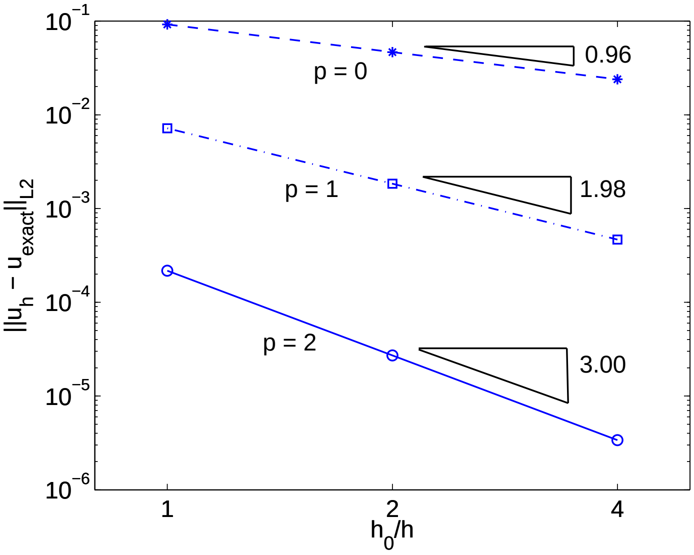


Figure 2-5: Mesh refinement results for $0 \leq p \leq 2$ using η_f as defined in Eqn 2.22

Thus, the mapping between the canonical triangle and physical space is given by

$$\mathbf{x} = \sum_j \mathbf{x}_j \phi_j(\xi),$$

where ϕ_j is the jth basis function, ξ is the location in the reference space, and $\mathbf{x}_j$ is the location of the jth node in physical space. In general, the Jacobian of this mapping need not be constant, meaning that triangles with curved edges can be mapped to the straight edged canonical element. Thus, by placing the non-interior, higher-order nodes on the real domain boundary, a higher-order geometry representation is achieved.

Two notes about this geometry representation must be made. First, on a curved element, basis functions that are polynomials of order p on the canonical element are not polynomials of order p in physical space. Thus, the interpolation order in physical space is less than p. Second, as discussed in Section 5.2, it is unclear if this choice of geometry representation is optimal as oscillations in the interpolated geometry may have detrimental effects on the order of accuracy.

2.4.2 Boundary Conditions

Boundary conditions are enforced weakly via the domain boundary integrals appearing in Eqn. 2.13. To evaluate these integrals, two previously undefined terms must be defined: $\mathbf{u}_h^b$ and $(\mathcal{A}_v \nabla \mathbf{u}_h)^b$. This section defines these terms for various boundary conditions.

Full State Condition

In some circumstances, the entire state vector at the boundary, $\mathbf{u}_h^b$, may be known. In these cases, the inviscid flux is computed using the Riemann solver exactly as if the face were an interior face:

$$\mathcal{H}_i^b(\mathbf{u}_h^+, \mathbf{u}_h^b, \hat{\mathbf{n}}) = \mathcal{H}_i(\mathbf{u}_h^+, \mathbf{u}_h^b, \hat{\mathbf{n}}).$$

No conditions are set on the viscous flux, thus, $(\mathcal{A}_v \nabla \mathbf{u}_h)^b$ is set by interpolating $(\mathcal{A}_v \nabla \mathbf{u}_h)^+$ to the boundary. $\boldsymbol{\delta}_f^b$ is computed using $\mathbf{u}_h^b$ as shown in Eqn. 2.12.

Inflow/Outflow Conditions

At an inflow/outflow boundary, the boundary state, $\mathbf{u}_h^b$, is defined using the outgoing Riemann invariants and given boundary data. Table 2.1 details the inflow/outflow conditions used in this work. Note that, while the Euler equations are well-posed with just the boundary conditions listed in Table 2.1, the Navier-Stokes equations are not. Conditions—numeric and/or physical—are needed to set $(\mathcal{A}_v \nabla \mathbf{u}_h)^b$ and $\boldsymbol{\delta}_f^b$. For this work, $(\mathcal{A}_v \nabla \mathbf{u}_h)^b$ is set by extrapolating from the interior of the domain, and $\boldsymbol{\delta}_f^b$ is computed as shown in Eqn. 2.12. These gradient conditions have not been theoretically investigated and may be expected to degrade accuracy and stability at low Re.

Table 2.1: Types of inflow/outflow boundary conditions.

Condition	Number of BCs	Value Specified	Outgoing Invariant
Subsonic Inflow	3	T_T, p_T, α	J^+
Supersonic Inflow	4	ρ, ρu, ρv, ρE	None
Subsonic Outflow	1	p	J^+, v, s
Supersonic Outflow	0	None	J^+, J^-, v, s

No Slip Wall Conditions

Two no slip wall conditions are used: adiabatic and isothermal. At a no slip, isothermal wall, the velocity components and static temperature are set. Thus,

$$\begin{aligned} u^b &= 0, \\ v^b &= 0, \\ T^b &= T_{wall}. \end{aligned}$$

To set the full state, these conditions are combined with the static pressure, p, which is computed using the interior state interpolated to the boundary and given boundary data ($u^b = v^b = 0$):

$$p = (\gamma - 1)\rho E^+.$$

Thus, the boundary state is given by

$$\mathbf{u}_h^b = \begin{pmatrix} \rho^b \\ \rho u^b \\ \rho v^b \\ \rho E^b \end{pmatrix} = \begin{pmatrix} \frac{p}{RT^b} \\ 0 \\ 0 \\ \rho E^+ \end{pmatrix}.$$

No physical conditions are set on the viscous flux at the wall. However, the scheme requires that this flux be set. Thus, it is extrapolated from the interior. The auxiliary variable, $\boldsymbol{\delta}_f$, is computed as stated in Eqn. 2.12.

At a no slip, adiabatic wall, the velocity components and the heat transfer to the wall are given:

$$\begin{aligned} u^b &= 0, \\ v^b &= 0, \\ \left.\frac{\partial T}{\partial n}\right|_b &= 0. \end{aligned}$$

Only two conditions on the boundary state are known, thus, two variables must be set using interior data and used to compute the boundary state. Static pressure, p, and stagnation enthalpy per unit mass, H, have been chosen:

$$\begin{aligned} p &= (\gamma - 1)\rho E^+, \\ H &= \frac{\rho E^+ + p}{\rho^+}. \end{aligned}$$

Thus, the boundary state is given by

$$\mathbf{u}_h^b = \begin{pmatrix} \rho^b \\ \rho u^b \\ \rho v^b \\ \rho E^b \end{pmatrix} = \begin{pmatrix} \frac{\gamma}{\gamma-1}\frac{p}{H} \\ 0 \\ 0 \\ \rho E^+ \end{pmatrix}.$$

The adiabatic condition, combined with the no slip condition, requires the viscous flux associated with the energy equation—the fourth flux component—to be zero. The other components of the viscous flux are unconstrained by this condition. Therefore, the interior viscous flux is used for these components and, thus, the boundary viscous flux is given by

$$[(\mathcal{A}_v \nabla \mathbf{u}_h)^b \cdot \hat{\mathbf{n}}] = \begin{pmatrix} [(\mathcal{A}_v \nabla \mathbf{u}_h)^+ \cdot \hat{\mathbf{n}}]_1 \\ [(\mathcal{A}_v \nabla \mathbf{u}_h)^+ \cdot \hat{\mathbf{n}}]_2 \\ [(\mathcal{A}_v \nabla \mathbf{u}_h)^+ \cdot \hat{\mathbf{n}}]_3 \\ 0 \end{pmatrix}.$$

Furthermore, since the energy equation viscous flux is specified, the corresponding component of the local lifting operator is set to zero. All other auxiliary variables are computed in the usual fashion. Thus,

$$\boldsymbol{\delta}_f^b \cdot \hat{\mathbf{n}} = \begin{pmatrix} [\boldsymbol{\delta}_f^b \cdot \hat{\mathbf{n}}]_1 \\ [\boldsymbol{\delta}_f^b \cdot \hat{\mathbf{n}}]_2 \\ [\boldsymbol{\delta}_f^b \cdot \hat{\mathbf{n}}]_3 \\ 0 \end{pmatrix}.$$

2.5 Final Discrete System

To define the final discrete form, it is necessary to select a basis for the space $\mathcal{V}_h^p$ and discretize in time. For the basis, a set of element-wise discontinuous functions, $\{\phi_j\}$, is chosen such that each ϕ_j has local support on one element only. Thus, the discrete solution has the form

$$\mathbf{u}_h(\mathbf{x}, t) = \sum_j \bar{\mathbf{u}}_j(t) \phi_j(\mathbf{x}).$$

In this work, a nodal Lagrange basis with uniformly spaced nodes is used. However, it should be noted that this basis becomes poorly conditioned as p increases, which can degrade the iterative convergence rate. A hierarchical basis like that used in [22] can eliminate this problem with the additional benefit of simplifying the multigrid prolongation and restriction operators.

A backward Euler discretization is used for the time integration. Thus, the discrete system is given by

$$\frac{1}{\Delta t} \mathcal{M}(\bar{\mathbf{u}}^{n+1} - \bar{\mathbf{u}}^n) + \mathbf{R}(\bar{\mathbf{u}}^{n+1}) = 0,$$

where $\mathcal{M}$ is the mass matrix and $\mathbf{R}(\bar{\mathbf{u}}^{n+1})$ is the steady residual vector.

This work is principally concerned with steady problems. However, the unsteady term is included to improve the performance of the solver in the initial iterations. After this initial transient period, $\Delta t \to \infty$ [22].

Chapter 3

Solution Method

Applying the discretization developed in Chapter 2, the discretized compressible Navier-Stokes equations are given by a nonlinear system of equations $\mathbf{R}(\mathbf{u}) = 0$. To solve this system, a p-multigrid scheme with element-line Jacobi smoothing, developed by Fidkowski and Darmofal [22, 23], is used. A general preconditioned iterative scheme can be written

$$\mathbf{u}^{n+1} = \mathbf{u}^{n} - \mathbf{P}^{-1}\mathbf{R}(\mathbf{u}^{n}),$$

where the preconditioner matrix, $\mathbf{P}$, is an approximation to the Jacobian, $\frac{\partial \mathbf{R}}{\partial \mathbf{u}}$. Two types of preconditioners are examined: element Jacobi, where the unknowns on a single element are solved simultaneously, and element-line Jacobi, where the unknowns on a line of elements are solved simultaneously. Details of both smoothers as well as the line creation procedure and multigrid solver are presented. The discussion draws heavily on [22].

3.1 Preconditioners

For the element Jacobi scheme, the unknowns on a single element are solved simultaneoulsy. The diagonal blocks of the Jacobian matrix represent the influence of the state variables in a given element on the residual in that element. Thus, the preconditioner matrix is the block diagnonal of the Jacobian. To improve the robustness during the initial iterations, the block diagonal is augmented by an unsteady term. Thus, the preconditioner is,

$$\mathbf{P} = \mathbf{D} + \frac{1}{\Delta t}\mathcal{M},$$

where $\mathbf{D}$ is the block diagonal of $\frac{\partial \mathbf{R}}{\partial \mathbf{u}}$ and $\mathcal{M}$ is the mass matrix. As the solution converges, $\Delta t \to \infty$.

The addition of the unsteady term does not change the block diagonal structure of $\mathbf{P}$. Thus, it is inverted one block at a time using Gaussian elimination.

The element-line Jacobi scheme is slightly more complex. In strongly convective systems, transport of information proceeds along characteristic directions. By solving implicitly on lines of elements connected along these directions, one can alleviate the stiffness associated with strong convection. Furthermore, for viscous flows, the element-line Jacobi solver is an important ingredient in removing the stiffness associated with regions of high grid anisotropy frequently required in viscous layers [2, 37]. Thus, the element-line Jacobi scheme requires the ability to construct lines of elements based on some measure of element-to-element coupling and to solve implicitly on each line. This section considers the construction and inversion of the preconditioner matrix given a set of lines. The line creation algorithm is described in Section 3.2.

Given a set of N_l lines, the preconditioner matrix, $\mathbf{P}$, is composed of N_l tridiagonal systems constructed from the linearized flow equations. Let the tridiagonal system for line l, where $1 \leq l \leq N_l$, be written $\mathbf{M}_l$. Then, denoting the number of elements in line l as n_l, $\mathbf{M}_l$ is a block $n_l \times n_l$ matrix. As before, the on-diagonal blocks represent the influence of the state variables in a given element on the residual in that element. The off-diagonal blocks, $\mathbf{M}_l(j,k)$, represent the influence of the states in element k on the residual in element j.

As in the element smoother case, the element-line smoother is augmented by an unsteady term to improve robustness. Thus, the final form of the preconditioner is,

$$\mathbf{P} = \mathbf{M} + \frac{1}{\Delta t}\mathcal{M},$$

where $\mathbf{M}$ is the set of assembled $\mathbf{M}_l$ matrices.

Inversion of $\mathbf{P}$ uses a block-tridiagonal algorithm in which the diagonal block is LU decomposed. As the dominant cost of the element-line Jacobi solver (especially for higher-order schemes) is the LU decomposition of the diagonal, the computational cost of the element-line Jacobi smoother scales as that of the simpler element Jacobi. However, the performance of the element-line Jacobi smoother is significantly better due to the increased implicitness along strongly coupled directions.

3.2 Line Creation

The effectiveness of the element-line Jacobi smoother depends largely on the quality of lines produced by the line creation procedure. For inviscid or nearly inviscid flows, information flows along characteristic directions set by convection. Thus, the lines should be aligned with the streamline direction to alleviate the stiffness associated with strong convection. In viscous flows, the effects of diffusion and regions of high grid anisotropy, such as those typically associated with boundary layers, couple elements in directions other than the convection direction. Connecting elements in these off-convection directions can alleviate the stiffness associated with diffusion and grid anisotropy.

The line creation procedure is divided into two parts: the connectivity criterion, which is a measure of the coupling between elements, and the line creation algorithm, which connects elements into lines based on the connectivity criterion.

3.2.1 Connectivity Criterion

The measure of coupling used in this work is similar to that used in the nodal line creation algorithm of Okusanya [39]. In that algorithm, the coupling was taken directly from the discretization.

In this work, the coupling is based on a $p = 0$ discretization of the scalar transport equation,

$$\nabla \cdot (\rho \vec{u} \phi) - \nabla \cdot (\mu \nabla \phi) = 0,$$

where $\rho\vec{u}$ and μ are taken from the solution at the current iteration. More specifically, the coupling between two elements j and k that share a face is given by

$$\mathbf{C}_{j,k} = \max\left(\left| \frac{\partial R_j}{\partial \phi_k} \right|, \left| \frac{\partial R_k}{\partial \phi_j} \right| \right).$$

While this definition of coupling does not represent the exact coupling between elements for the higher-order Navier-Stokes discretization being solved, it captures the relevant features—the effects of convection and diffusion—and remains unique. In the $p = 0$ discretization of the scalar transport equation, the off-diagonal components of the Jacobian matrix are scalars. For a $p \geq 1$ discretization of the scalar transport equation or any order discretization of a system of equations, the off-diagonal blocks of the Jacobian are matrices. Thus, a matrix norm would be required to make the coupling unambiguous.

3.2.2 Line Creation Algorithm

After computing the elemental coupling, lines of elements are formed using the line creation algorithm developed by Fidkowski and Darmofal [22, 23]. The summary given here is based on [22].

The line creation process is divided into two stages: line creation and line connection. Let $N(j; f)$ denote the element adjacent to element j across face f, and let $F(j)$ denote the set of faces enclosing element j. Then, the line creation algorithm is as follows:

Stage I: Line Creation

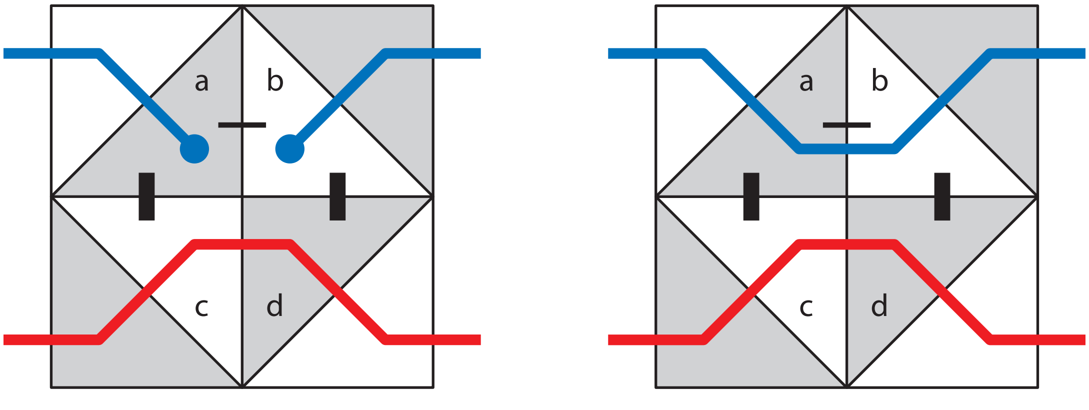
1. Obtain a seed element i
2. Call MakePath(i) - *Forward Path*
3. Call MakePath(i) - *Backward Path*
4. Return to (1). The algorithm is finished when no more seed elements exist.

MakePath(j)

While path not terminated:

> For element j, pick the face $f \in F(j)$ with highest connectivity, such that element $k = N(j; f)$ is not part of the current line. Terminate the path if any of the following conditions hold:
>
> - face f is a boundary face
> - element k is already part of a line
> - $C(j, k)$ is not one of the top two connectivities in element k
>
> Otherwise, assign element j to the current line, set $j = k$, and continue.

After completing Stage I, it is possible that the endpoints of two lines are adjacent to each other, as illustrated in Figure 3-1. Elements a and b have not been connected because the



(a) (b)

Figure 3-1: Possible line configuration: (a) after Stage I and (b) after stage II. Reproduced with permission from [22].

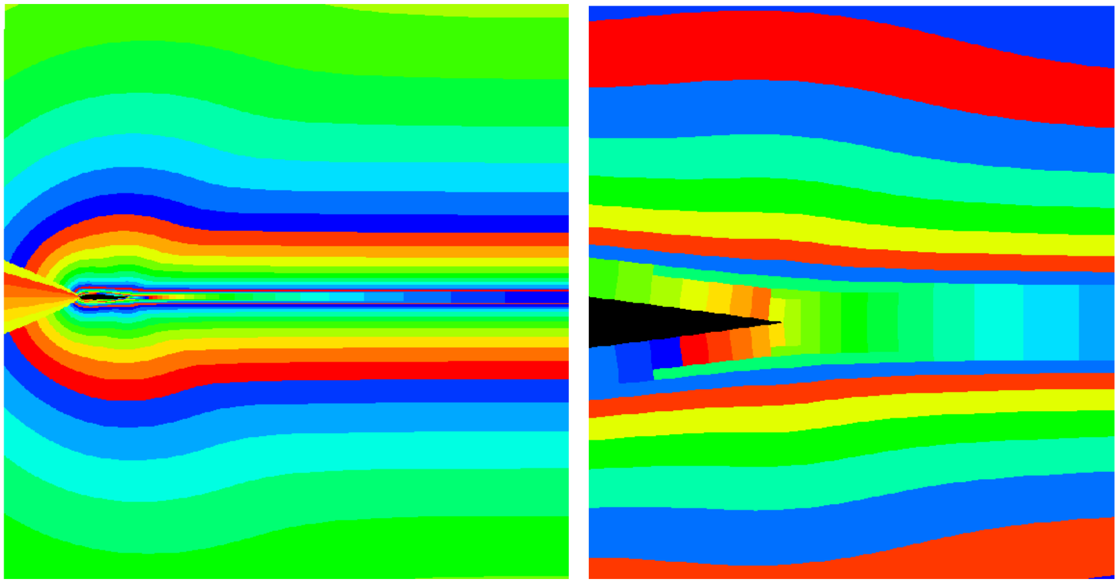
connectivity, $C_{a,b}$, is the minimum connectivity for both elements. However, the pairs a, c and b, d have not been connected because $C_{a,c}$ and $C_{b,d}$ are the minimum connectivities for elements c and d, respectively. For best solver performance, it is desirable to use lines of maximum length. Thus, it is necessary to connect elements a and b. The line connection stage accomplishes this goal.

Stage II: Line Connection

1. Loop through endpoint elements, j, of all lines. Denote by $H_j \subset F(j)$ the set of faces h of j that are boundary faces or that have $N(j;h)$ as a line endpoint.
2. Choose $h \in H_j$ of maximum connectivity. If h is not a boundary face, let $k = N(j;h)$.
3. If k has no other neighboring endpoints of higher connectivity, and no boundary faces of higher connectivity, then connect the two lines to which j and k belong.

Proofs that both stages of the line creation algorithm result in a unique set of lines, independent of seed element, are provided in [22].

The lines formed for a 2-D NACA 0012 test case are shown in Figure 3-2. As shown in Figure 3-2(a), the lines in the outer flow, where convection dominates, simply follow the streamline direction. In viscous regions—the boundary layer and wake in this problem—the



(a) Outer flow

(b) Trailing edge

Figure 3-2: Lines formed around NACA 0012 in $M = 0.5$, $Re = 5000$, $\alpha = 0^o$ flow

effects of high aspect ratio elements and diffusion produce elements that are tightly coupled in the direction normal to convection. Thus, as shown in Figure 3-2(b), lines are formed normal to the streamline direction. Details and results for this case are given in Chapter 5.

3.3 p-Multigrid

3.3.1 Motivation

The use of multigrid techniques is motivated by the observation that the smoothers developed in Section 3.1 are ineffective at eliminating low-frequency error modes on the fine grid. In standard h-multigrid, spatially coarser grids are used to correct the error on the fine grid. On a coarser grid, the low-frequency error modes from the fine grid appear as high-frequency modes, and, thus, the smoothers can effectively correct them. p-Multigrid uses the same principle except that lower-order approximations serve as the "coarse grid" [48, 26].

Furthermore, p-multigrid fits naturally into the high-order DG, unstructured grid framework. Unlike h-multigrid, spatially coarser meshes are not required. Thus, no element agglomeration or re-meshing procedures are necessary. Only prolongation and restriction between orders are required. Moreover, the prolongation and restriction operators are local, meaning that they must only be stored for the canonical element.

3.3.2 FAS and Two-level Multigrid

The multigrid method used here is the Full Approximation Scheme (FAS), introduced by Brandt [14]. The following description is taken from Fidkowski [22].

Consider the discretized system of equations given by

$$\mathbf{R}^p(\mathbf{u}^p) \quad = \quad \mathbf{f}^p,$$

where $\mathbf{u}^p$ is the discrete solution vector for p^{th} order interpolation on a given grid, $\mathbf{R}^p(\mathbf{u}^p)$ is the associated nonlinear system, and $\mathbf{f}^p$ is a source term (zero for the fine-level problem). Let $\mathbf{v}^p$ be an approximation to the solution vector and define the discrete residual, $\mathbf{r}^p(\mathbf{v}^p)$, by

$$\mathbf{r}^p(\mathbf{v}^p) \quad \equiv \quad \mathbf{f}^p - \mathbf{R}^p(\mathbf{v}^p).$$

In a basic two-level multigrid method, the exact solution on a coarse level is used to correct the solution on the fine level. This multigrid scheme is given as follows:

- Perform ν_1 smoothing interations on the fine level: $\mathbf{v}^{p,\,n+1} = \mathbf{v}^{p,\,n} - \mathbf{P}^{-1}\mathbf{r}^p(\mathbf{v}^{p,\,n})$

- Restrict the state and residual to the coarse level: $\mathbf{v}_0^{p-1} = \tilde{I}_p^{p-1}\mathbf{v}^p$, $\mathbf{r}^{p-1} = I_p^{p-1}\mathbf{r}^p$.
- Solve the coarse level problem: $\mathbf{R}^{p-1}(\mathbf{v}^{p-1}) = \mathbf{R}^{p-1}(\mathbf{v}_0^{p-1}) + \mathbf{r}^{p-1}$.
- Prolongate the coarse level error and correct the fine level state: $\mathbf{v}^p = \mathbf{v}^p + I_{p-1}^p(\mathbf{v}^{p-1} - \mathbf{v}_0^{p-1})$
- Perform ν_2 smoothing interations on the fine level: $\mathbf{v}^{p,\,n+1} = \mathbf{v}^{p,\,n} - \mathbf{P}^{-1}\mathbf{r}^p(\mathbf{v}^{p,\,n})$.

I_p^{p-1} is the residual restriction operator, and I_{p-1}^p is the state prolongation operator. $\tilde{I}_p^{p-1}$ is the state restriction operator and is not necessarily the same as the residual restriction. Alternatively, the FAS coarse level equation can be written as

$$\begin{aligned} \mathbf{R}^{p-1}(\mathbf{v}^{p-1}) &= I_p^{p-1}\mathbf{f}^p + \tau_p^{p-1}, \\ \tau_p^{p-1} &\equiv \mathbf{R}^{p-1}(\tilde{I}_p^{p-1}\mathbf{v}^p) - I_p^{p-1}\mathbf{R}^p(\mathbf{v}^p). \end{aligned}$$

The first equation differs from the original coarse level equation by the presence of the term τ_p^{p-1}, which improves the correction property of the coarse level. In particular, if the fine level residual is zero, the coarse level correction is zero since $\mathbf{v}^{p-1} = \mathbf{v}_0^{p-1}$.

3.3.3 V-cycles and FMG

To make multigrid practical, the two-level correction scheme is extended to V-cycles and Full Multigrid (FMG). In a V-cycle, one or more levels are used to correct the fine level solution. Descending from the finest level, after restriction, ν_1 smoothing steps are performed at each level until the coarsest level is reached. On the coarsest level, the problem may be solved exactly or smoothed a relatively large number of times. Ascending, the problem is smoothed ν_2 times on each level after prolongation until the finest level is reached. This procedure constitutes one V-cycle which is also refered to as one multigrid iteration.

Using only V-cycles to obtain high-order solutions is impractical because it requires starting the calculation on the finest level, where there are the most degrees of freedom and smoothing is most expensive. This problem can be eliminated by using FMG, where the calculation is started on the coarsest level. After converging or partially converging the solution, it is prolongated to the next finer level. Running V-cycles at this level, the solution is partially converged and then prolongated to the next finer level. This procedure continues until the desired solution order is reached.

For more details on the prolongation and restriction operators or the multigrid implementation, see [22, 23].

Chapter 4

Stability Analysis

To determine the stability of the smoothers discussed in Chapter 3, Fourier (Von Neumann) analysis is performed for convection-diffusion in one and two dimensions with periodic boundary conditions. The analysis follows that of [22] for advection. The convection-diffusion problem is given by

$$\begin{aligned} \vec{V} \cdot \nabla u - \nu \nabla^2 u &= f(\vec{x}), \\ a u_x - \nu u_{xx} &= f(x) \quad \text{on} \quad [-1,1], \\ a u_x + b u_y - \nu (u_{xx} + u_{yy}) &= f(x,y) \quad \text{on} \quad [-1,1] \times [-1,1]. \end{aligned}$$

For this analysis, the velocity, $\vec{V}$, is constant, u is the concentration variable, and f is a source term.

4.1 Outline of Analysis

To begin, the domain is triangulated, and the convection-diffusion equation is discretized following the steps in Chapter 2. The resulting discrete system is linear and will be written $\mathbf{Au} = \mathbf{f}$, where $\mathbf{u}$ is the exact solution. Denoting the current solution guess as $\mathbf{v}^n$, the general iterative solution procedure as defined in Chapter 3 is given by

$$\mathbf{v}^{n+1} = \mathbf{v}^n - \mathbf{P}^{-1}\mathbf{r}^n,$$

where

$$\mathbf{r}^n = \mathbf{A}\mathbf{v}^n - \mathbf{f}. \tag{4.1}$$

Defining the error at iteration n to be $\mathbf{e}^n = \mathbf{v}^n - \mathbf{u}$, Eqn. 4.1 can be written in terms of the error:

$$\mathbf{A}\mathbf{e}^n = \mathbf{r}^n. \tag{4.2}$$

Thus, the iterative scheme can also be put in terms of the error:

$$\mathbf{e}^{n+1} = \mathbf{S}\mathbf{e}^n,$$

where $\mathbf{S} = \mathbf{I} - \mathbf{P}^{-1}\mathbf{A}$ is the iteration matrix and $\mathbf{I}$ is the identity matrix. The spectral radius of the iteration matrix, $|\rho(\mathbf{S})|$, determines the growth or decay of the error. Thus, to determine the stability of the iterative scheme, it is necessary to compute the eigenvalue footprint of this matrix. For stability, the eigenvalues of $\mathbf{S}$ must lie in the unit circle centered at the origin. This stability condition requires that the eigenvalues of $-\mathbf{P}^{-1}\mathbf{A}$ lie in the unit circle centered at $(-1, 0)$. Thus, eigenvalue footprints of $-\mathbf{P}^{-1}\mathbf{A}$ are computed for both element and element-line Jacobi relaxation via Fourier analysis. The specifics of the discretization and Fourier analysis, including results, are presented in Sections 4.2 and 4.3.

4.2 One Dimensional Analysis

To discretize, the domain, $[-1, 1]$, is divided into a triangulation, T_h, of N elements, κ, of size $\Delta x = 2/N$, such that $\bigcup_{\kappa \in T_h} \kappa = [-1, 1]$. Then, using the solution and test function space $\mathcal{V}_h^p$ defined by Eqn. 2.2, the DG discretization of Eqn. 4.1 takes the following form: find $u_h \in \mathcal{V}_h^p$ such that $\forall v_h \in \mathcal{V}_h^p$,

$$\sum_{\kappa \in T_h} \left[\mathcal{H}(u_h) v_h^+|_{\kappa_R} - \mathcal{H}(u_h) v_h^+|_{\kappa_L} - \int_\kappa a u_h v_{h,x}\, dx + \int_\kappa \nu u_{h,x} v_{h,x}\, dx \right]$$
$$- \sum_\Gamma \left[[\![v_h]\!] \left(\nu\{u_{h,x}\} - \eta_f \{\delta_f\}\right) + [\![u_h]\!]\{\nu v_{h,x}\} \right] = \sum_{\kappa \in T_h} \int_\kappa v_h f\, dx, \tag{4.3}$$

where κ_L and κ_R denote the left and right boundaries, respectively, of the element κ, and Γ is the union of element boundaries over the entire domain. Full-upwinding is used for the inter-element inviscid flux, such that $\mathcal{H}(u_h)$ is given by,

$$\mathcal{H}(u_h) = \frac{1}{2}(a - |a|)u_{h,R} + \frac{1}{2}(a + |a|)u_{h,L},$$

where $u_{h,L}$ and $u_{h,R}$ are values of u_h taken from the left and right elements at an interface. In 1-D, the local lifting operator is a scalar defined by the following: find $\delta_f \in \mathcal{V}_h^p$ such that

$\forall \tau_h \in \mathcal{V}_h^p$,

$$\int_{\kappa^{L/R}} \tau_h \delta_f^{L/R} \, dx = -\frac{1}{2} \left[\nu \tau_h (u_{h,L} - u_{h,R}) \right]_{\sigma_f},$$

where σ_f denotes a single inter-element boundary, and $\kappa^{L/R}$ represents the elements to the left and right of this boundary. Finally, for uniform spacing, $\eta_f = 2$ at each interface.

Using a basis $\{\phi_j\}$ for the finite element space $\mathcal{V}_h^p$, the concentration, u_h, is represented by $u_h(x) = \sum_j u_j \phi_j(x)$. Thus, Eqn. 4.3 can be written concisely as $\mathbf{Au} = \mathbf{f}$, where $\mathbf{u}$ is the vector of solution coefficients, u_j. For this analysis, standard Lagrange basis functions are used.

Assuming the error varies sinusoidally on the elements, it can be decomposed into N modes,

$$\mathbf{e}^n = \sum_{j=-N/2+1}^{N/2} \mathbf{e}^n(\theta_j),$$

where the jth error mode is given by,

$$\mathbf{e}^n(\theta_j) = \begin{bmatrix} \bar{\mathbf{e}}_1^n(\theta_j) \\ \vdots \\ \bar{\mathbf{e}}_r^n(\theta_j) \\ \vdots \\ \bar{\mathbf{e}}_N^n(\theta_j) \end{bmatrix}, \qquad \bar{\mathbf{e}}_r^n(\theta_j) = \bar{\mathbf{v}}^n(\theta_j) \exp(ir\theta_j), \tag{4.4}$$

and $\theta_j = j\pi\Delta x$ and r is the element index. Thus, $\bar{\mathbf{e}}_r^n(\theta_j)$ is a vector of length $p+1$ corresponding to the jth error mode on element r.

Using the fact that the stencil is element compact, the Eqn. 4.2 can be written, for any element r, as

$$\widehat{\mathbf{A}}^W \bar{\mathbf{e}}_{r-1}^n + \widehat{\mathbf{A}}^0 \bar{\mathbf{e}}_r^n + \widehat{\mathbf{A}}^E \bar{\mathbf{e}}_{r+1}^n = \mathbf{r}_r^n, \tag{4.5}$$

where $\widehat{\mathbf{A}}^W$, $\widehat{\mathbf{A}}^0$, and $\widehat{\mathbf{A}}^E$ are the $(r, r-1)$, (r, r), and $(r, r+1)$ blocks of the matrix $\mathbf{A}$. Because the boundary conditions are periodic, if $r = 1$, $r-1$ refers to element N, and if $r = N$, $r+1$ refers to element 1. Substituting the form of the error from Eqn. 4.4 into Eqn. 4.5 gives

$$\left[\widehat{\mathbf{A}}^W \exp(-i\theta_j) + \widehat{\mathbf{A}}^0 + \widehat{\mathbf{A}}^E \exp(i\theta_j) \right] \bar{\mathbf{e}}_r^n = \mathbf{r}_r^n.$$

Thus, the system of $N(p+1)$ equations governing the error represented by Eqn. 4.2 can be reduced to a system of $(p+1)$ equations for each error mode. Furthermore, the iterative scheme can also be reduced such that the relaxation of the jth error mode is given by

$$\bar{\mathbf{e}}_r^{n+1}(\theta_j) = \tilde{\mathbf{S}}(\theta_j)\bar{\mathbf{e}}_r^n(\theta_j) = \tilde{\mathbf{S}}^{n+1}(\theta_j)\bar{\mathbf{v}}^0(\theta_j)\exp(ir\theta_j),$$

where $\tilde{\mathbf{S}}(\theta_j)$ is a $(p+1)\times(p+1)$ matrix corresponding to the iteration matrix, $\mathbf{S}$, for sinusoidal error variation. Thus, to determine the stability of an iterative scheme, one must compute the eigenvalues of $\tilde{\mathbf{S}}(\theta_j)$ for all j.

For the element Jacobi smoother, the preconditioner, $\mathbf{P}$, is the block diagonal of $\mathbf{A}$. Thus, for sinusoidal error, the $(p+1)\times(p+1)$ equivalent of the iteration matrix is given by

$$\tilde{\mathbf{S}}(\theta_j) = \tilde{\mathbf{I}} - \tilde{\mathbf{P}}^{-1}\tilde{\mathbf{A}}(\theta_j),$$

where

$$\tilde{\mathbf{P}}^{-1}\tilde{\mathbf{A}}(\theta_j) = (\widehat{\mathbf{A}}^0)^{-1}(\widehat{\mathbf{A}}^W\exp(-i\theta_j) + \widehat{\mathbf{A}}^0 + \widehat{\mathbf{A}}^E\exp(i\theta_j)).$$

Footprints of $-\tilde{\mathbf{P}}^{-1}\tilde{\mathbf{A}}$ for the 1-D element Jacobi smoother are shown in Figure 4-1. In 1-D, the footprints depend only on the solution order, p, and the element Reynolds number, $Re \equiv a\Delta x/\nu$. The figures show footprints for $p = 0, 1, 2, 3$ at four Re. Note that all the eigenvalues are stable and that, as $Re \to \infty$, all the eigenvalues associated with $p > 0$ are centered at the origin [22].

In 1-D the element-line Jacobi smoother becomes an exact solve. Thus, this smoother is only examined in 2-D.

4.3 Two Dimensional Analysis

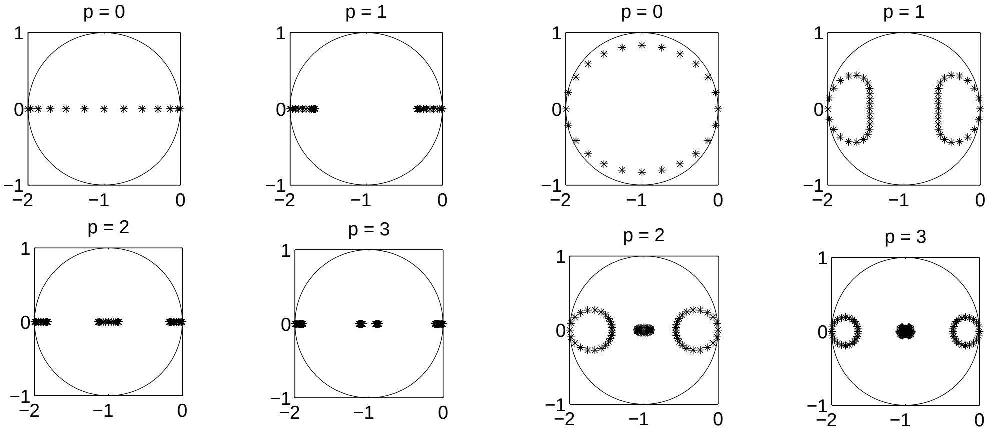
In 2-D, the domain is subdivided into N_xN_y rectangular elements, κ, of size Δx by Δy where $\Delta x = 2/N_x$ and $\Delta y = 2/N_y$. The discretization procedure is analogous to that for the 1-D case. The 2-D basis is given by the tensor product of the 1-D basis: $\phi_{\alpha\beta}(x,y) = \phi_\alpha(x)\phi_\beta(y)$, where ϕ_α and ϕ_β are 1-D Lagrange basis functions. Thus, there are $(p+1)^2$ basis functions per element.

Indexing elements by the ordered pair (r,s), error modes have the form

$$\bar{\mathbf{e}}_{r,s}^n(\theta_j,\theta_k) = \bar{\mathbf{v}}^n\exp(ir\theta_j + is\theta_k),$$

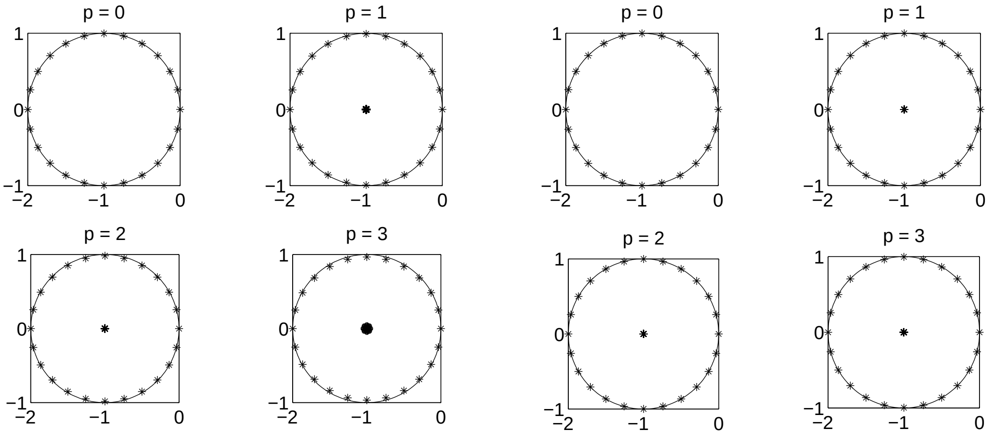
where $\theta_j = j\pi\Delta x$, $\theta_k = k\pi\Delta y$, and $j,k \in (-N/2+1, ..., N/2)$. Thus, for the element (r,s), Eqn. 4.2 becomes

$$\left[\widehat{\mathbf{A}}^W\exp(-i\theta_j) + \widehat{\mathbf{A}}^S\exp(-i\theta_k) + \widehat{\mathbf{A}}^0 + \widehat{\mathbf{A}}^E\exp(i\theta_j) + \widehat{\mathbf{A}}^N\exp(i\theta_k)\right]\bar{\mathbf{e}}_{r,s}^n = \mathbf{r}_{r,s}^n,$$



(a) $Re = 0$

(b) $Re = 10$



(c) $Re = 1000$

(d) $Re = 10000$

Figure 4-1: Eigenvalue footprints for element Jacobi preconditioned 1-D convection-diffusion

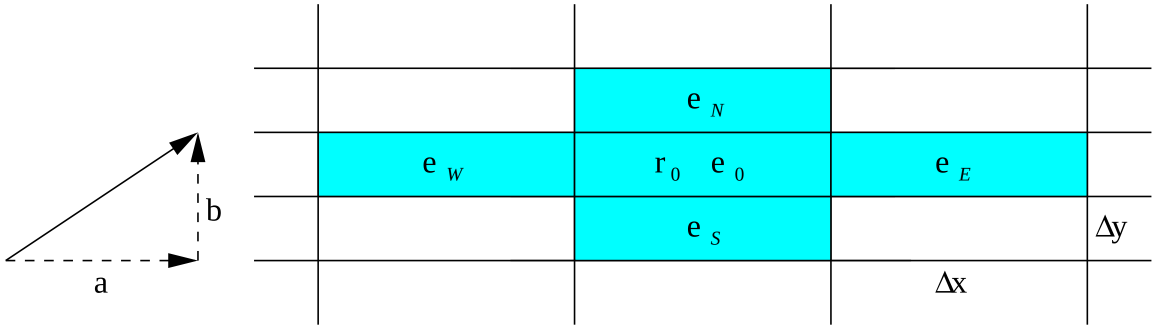


Figure 4-2: Stencil of element 0

where each $\widehat{\mathbf{A}}$ matrix is size $(p+1)^2 \times (p+1)^2$.

Thus, the $(p+1)^2 \times (p+1)^2$ iteration matrix corresponding to $\mathbf{S}$ for sinusoidal error is given by the following:

$$\tilde{\mathbf{S}}(\theta_j, \theta_k) = \tilde{\mathbf{I}} - \tilde{\mathbf{P}}^{-1}(\theta_j, \theta_k)\tilde{\mathbf{A}}(\theta_j, \theta_k).$$

The preconditioner in the element smoothing case is still simply the block diagonal:

$$\tilde{\mathbf{P}} = \widehat{\mathbf{A}}^0.$$

The form of the element-line smoother depends on the direction of the lines. As described in Section 3.2, lines are formed based on the coupling between elements for a $p = 0$ discretization of convection-diffusion. Figure 4-2 shows the dependence of $\mathbf{r}_0$ on the surrounding elements. For $p = 0$, the residual and error on each element are constant, thus, the residual on element zero is

$$\mathrm{r}_0 = \begin{bmatrix} \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_E} & \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_N} & \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_0} & \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_W} & \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_S} \end{bmatrix} \begin{bmatrix} \mathrm{e}_E \\ \mathrm{e}_N \\ \mathrm{e}_0 \\ \mathrm{e}_W \\ \mathrm{e}_S \end{bmatrix},$$

where

$$\begin{bmatrix} \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_E} \\ \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_N} \\ \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_0} \\ \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_W} \\ \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_S} \end{bmatrix} = \begin{bmatrix} \left(\frac{1}{2}\Delta y(a - |a|) - \frac{\Delta y}{\Delta x}\nu\right) \\ \left(\frac{1}{2}\Delta x(b - |b|) - \frac{\Delta x}{\Delta y}\nu\right) \\ \left(\Delta y|a| + \Delta x|b| + 2\nu(\frac{\Delta y}{\Delta x} + \frac{\Delta x}{\Delta y})\right) \\ \left(-\frac{1}{2}\Delta y(a + |a|) - \frac{\Delta y}{\Delta x}\nu\right) \\ \left(-\frac{1}{2}\Delta x(b + |b|) - \frac{\Delta x}{\Delta y}\nu\right) \end{bmatrix}.$$

Note that because the problem has constant coefficients and the grid is uniform,

$$\begin{aligned}
\frac{\partial \mathrm{r}_E}{\partial \mathrm{e}_0} &= \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_W}, \\
\frac{\partial \mathrm{r}_N}{\partial \mathrm{e}_0} &= \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_S}, \\
\frac{\partial \mathrm{r}_W}{\partial \mathrm{e}_0} &= \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_E}, \\
\frac{\partial \mathrm{r}_S}{\partial \mathrm{e}_0} &= \frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_N}.
\end{aligned}$$

These relationships imply that the vertical face and horizontal face connectivity values are constant throughout the domain. Thus, if the horizontal connectivity is greater than the vertical connectivity,

$$\max\left(\left|\frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_E}\right|, \left|\frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_W}\right|\right) \geq \max\left(\left|\frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_N}\right|, \left|\frac{\partial \mathrm{r}_0}{\partial \mathrm{e}_S}\right|\right),$$

which simplifies to

$$\Delta y\left(|a| + \frac{\nu}{\Delta x}\right) \geq \Delta x\left(|b| + \frac{\nu}{\Delta y}\right),$$

the lines are horizontal everywhere. Otherwise, they are vertical.

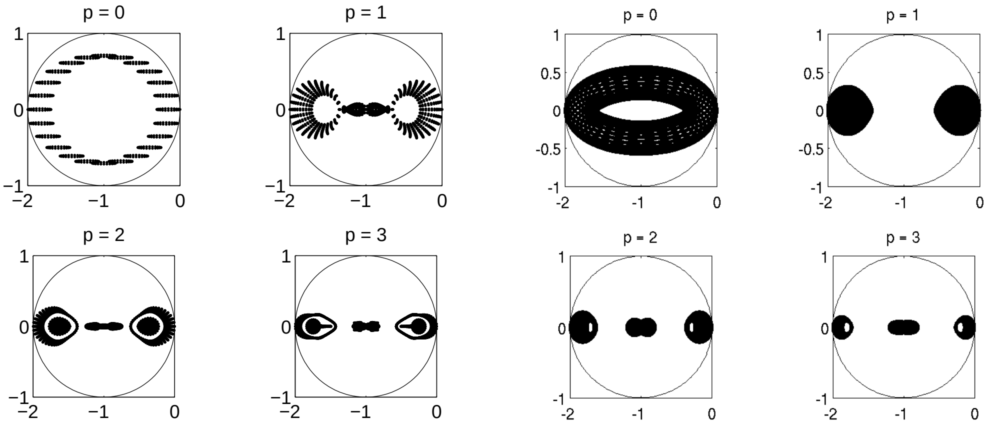
For the horizontal case, the preconditioner is

$$\tilde{\mathbf{P}}(\theta_j) = \widehat{\mathbf{A}}^W \exp(-i\theta_j) + \widehat{\mathbf{A}}^0 + \widehat{\mathbf{A}}^E \exp(i\theta_j),$$

and for the vertical case,

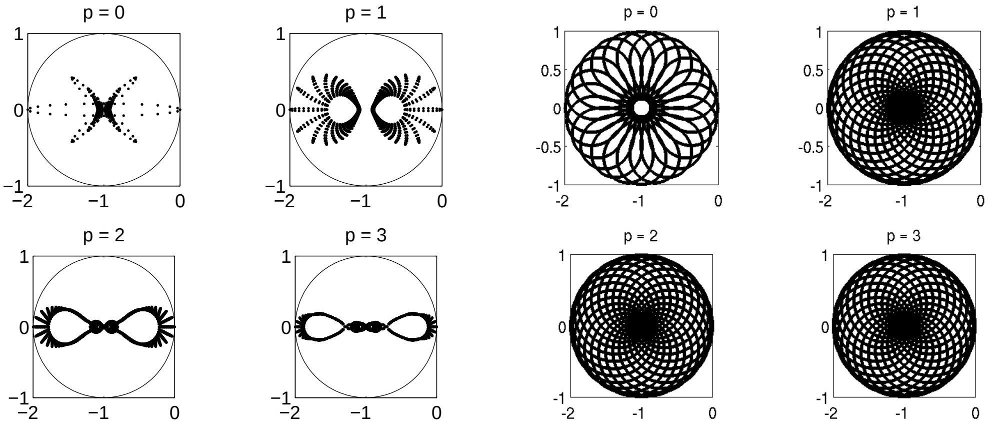
$$\tilde{\mathbf{P}}(\theta_k) = \widehat{\mathbf{A}}^S \exp(-i\theta_k) + \widehat{\mathbf{A}}^0 + \widehat{\mathbf{A}}^N \exp(i\theta_k).$$

In 2-D, the eigenvalue footprints depend of five parameters: smoother choice; solution order, p; element Reynolds number, $Re \equiv a\Delta x/\nu$; element aspect ratio, $\mathrm{AR} = \Delta x/\Delta y$; and flow angle, $\tan\alpha = b/a$. Footprints of $-\mathbf{P}^{-1}\mathbf{A}$ for both element and element-line Jacobi smoothing are shown in Figure 4-3 and Figure 4-4. Two flow conditions are shown: a low AR, low *Re* case, and a high AR, high *Re* case. The cases are meant to be representative of the range of flow conditions one would encounter in a typical aerodynamic simulation. The low AR, low *Re* case approximates the conditions one would expect near the leading edge of an airfoil, while the high AR, high *Re* case is representative of a boundary layer in a high speed flow. While only two cases are shown, a comprehensive study of the parameter space was conducted using $1 \leq \mathrm{AR} \leq 500$, $0 \leq Re \leq 10000$, and $1^o \leq \alpha \leq 25^o$. As in the 1-D case, the smoothers are stable independent of p and flow condition.



(a) AR = 1, $Re = 10$, $\alpha = 1^o$ (b) AR = 100, $Re = 10000$, $\alpha = 1^o$

Figure 4-3: Eigenvalue footprints for element Jacobi preconditioned 2-D convection-diffusion



(a) Horizontal lines, AR = 1, $Re = 10$, $\alpha = 1^o$ (b) Vertical lines, AR = 100, $Re = 10000$, $\alpha = 1^o$

Figure 4-4: Eigenvalue footprints for element-line Jacobi preconditioned 2-D convection-diffusion

Chapter 5

Numerical Results

In this chapter, three test cases are considered. For the first two cases, source terms are added to the compressible Navier-Stokes equations such that a desired analytic function is the exact solution. Thus, the L_2-norm of the error in the solution may be computed. Mesh refinement studies for these cases show optimal order of accuracy.

The last case presented is that of $M = 0.5$, $Re = 5000$, laminar flow over a NACA 0012 airfoil at $\alpha = 0^o$. A grid refinement study shows that the drag on the airfoil can be computed accurately in less time using high-order rather than highly refined meshes.

5.1 Poiseuille Flow

To numerically verify the order of accuracy of the discretization, the first test case considered is that of fully developed flow in a straight channel. Thus, the exact solution is of the form:

$$\begin{aligned} \rho &= \text{constant}, \\ u &= \frac{1}{2\mu}\left(\frac{dp}{dx}\right)y(y-b), \\ v &= 0. \end{aligned}$$

Of course, the classical, Poiseuille form is a solution of the incompressible Navier-Stokes equations. To make this form satisfy the compressible Navier-Stokes equations, constant viscosity is assumed, and a source term is added to the right-hand side of the energy equation. Thus, the equations are

$$\mathbf{u}_t + \nabla \cdot \mathcal{F}_i(\mathbf{u}) - \nabla \cdot \mathcal{F}_v(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{S}, \tag{5.1}$$

where,

$$\mathbf{S} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{2\mu}\left(\frac{dp}{dx}\right)^2 y(y-b)\left[\frac{1}{\gamma-1} - \frac{1}{2}y(y-b)\right] \end{pmatrix}.$$

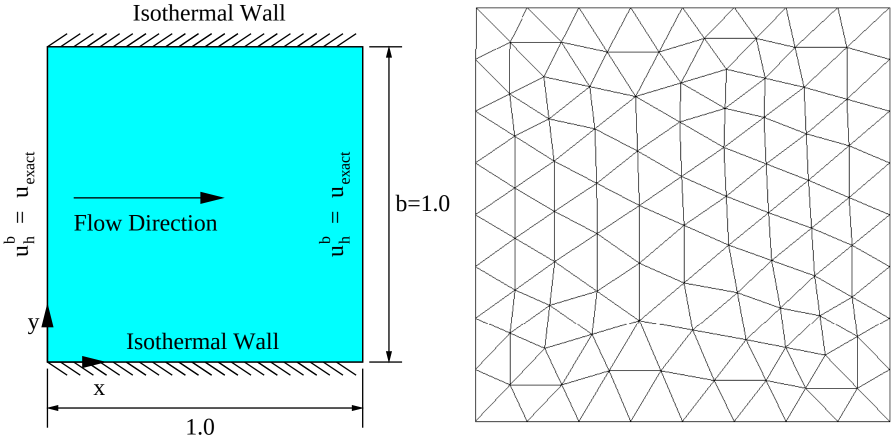
Finally, in terms of the conservative state variables, the exact solution is given by

$$\mathbf{u} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2\mu}\left(\frac{dp}{dx}\right) y(y-b) \\ 0 \\ \frac{1}{\gamma-1}(p_o + \frac{dp}{dx}x) + u^2 \end{pmatrix}.$$

For this test, the parameters μ, p_0, and $\frac{dp}{dx}$ are chosen as

$$\begin{aligned} \mu &= 1 \times 10^{-4}, \\ p_o &= 1, \\ \frac{dp}{dx} &= -8 \times 10^{-4}. \end{aligned}$$

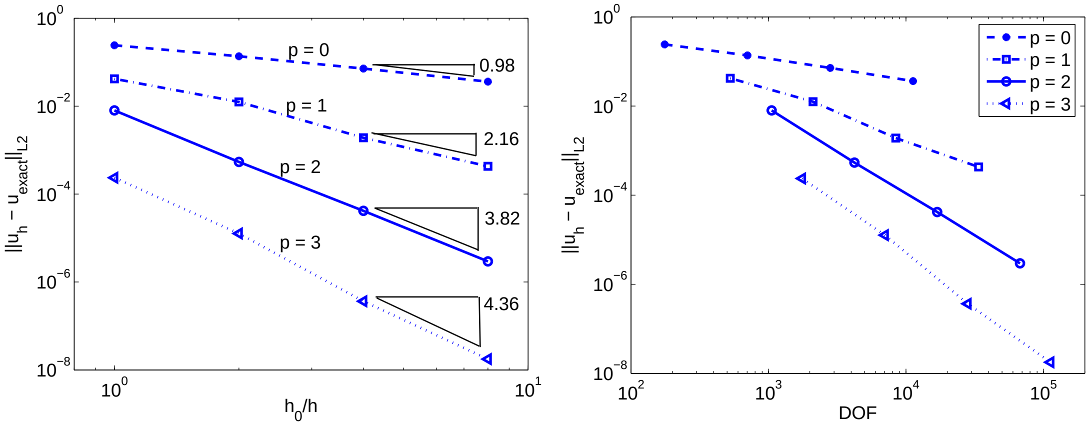
The domain of interest is a 1 × 1 square shown in Figure 5-1(a). On the left and right boundaries, the boundary state is set to the exact solution. On the top and bottom boundaries, no slip, isothermal wall conditions are imposed.



(a) Dimensions and BCs

(b) Medium mesh, 176 elements

Figure 5-1: Domain for Poiseuille flow test case



(a) Error vs. Grid Spacing

(b) Error vs. DOF

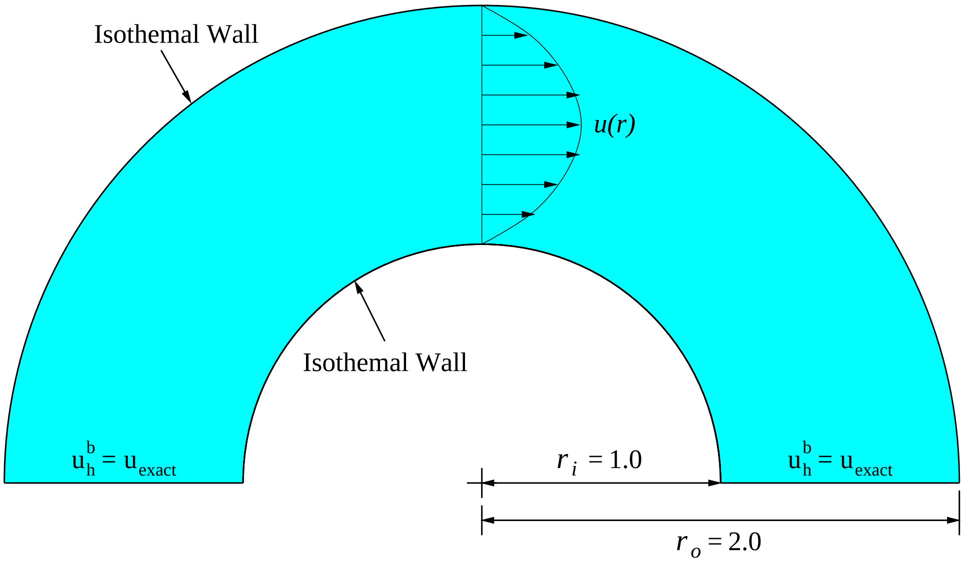
Figure 5-2: Accuracy results for $0 \leq p \leq 3$ for Poiseuille flow test case

The coarsest grid used contains 44 elements. The next finer grid (176 elements), which is shown in Figure 5-1(b), was generated by subdividing each element into four elements such that the grid spacing, h, is halved. This procedure was repeated a total of three times, resulting in four meshes containing 44, 176, 704, and 2816 elements. Note that, since the channel has straight geometry, it can be represented exactly with straight-edged ($q = 1$) triangles. From this point forward, q is used to represent the order of the basis used for the geometry interpolation.

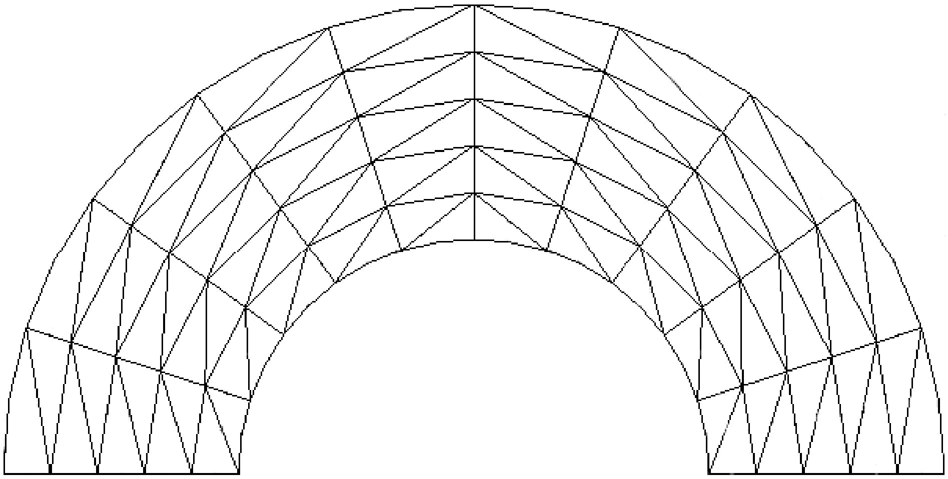
Results of the mesh refinement study are shown in Figure 5-2. Figure 5-2(a) shows that the L_2-norm of the error is converging at approximately $O(h^{p+1})$. Furthermore, as demostrated by Figure 5-2(b), the higher-order solutions are dramatically more accurate per degree of freedom (DOF). For the finest mesh, $p = 1$ solution, which has 3.38×10^4 DOF, the error is approximately 4.27×10^{-4}, while the coarsest mesh, $p = 3$ solution has only 7.04×10^3 DOF and gives an error of 2.36×10^{-4}.

5.2 Circular Channel Flow

The goal of the second test case was to numerically verify that the discretization achieves optimal order of accuracy for a problem with curved boundaries. The domain, shown in Figure 5-3(a), is a 180^o section of a circular channel. Figure 5-3(b) shows the coarsest grid used, which contains 100 elements. Note that on the circular boundaries the geometry is represented using a $q = 2$ Lagrange basis where the midpoint node on each boundary edge

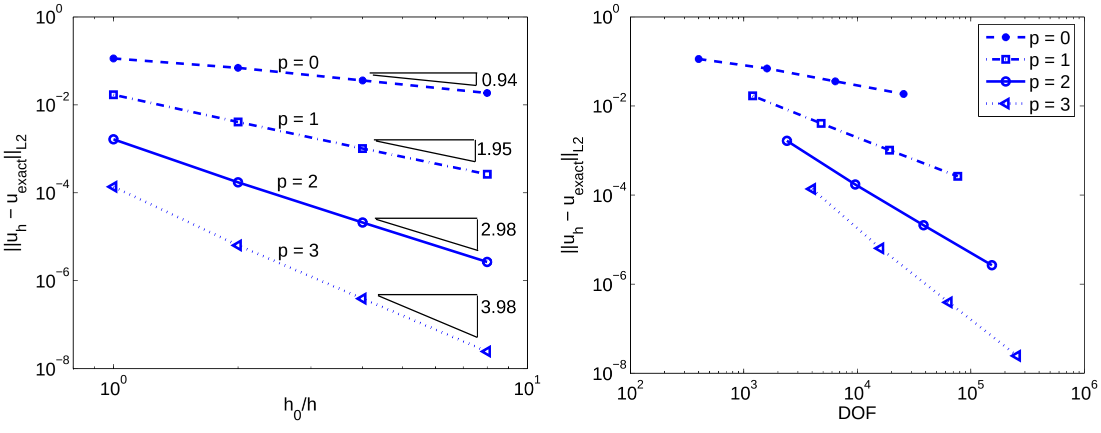

Isothemal Wall
u(r)
Isothemal Wall
$u_h^b = u_{exact}$
$r_i = 1.0$
$u_h^b = u_{exact}$
$r_o = 2.0$

(a) Dimensions and boundary conditions



(b) Coarse mesh, 100 elements, $q = 2$

Figure 5-3: Domain for circular channel test case



(a) Error vs. Grid Spacing

(b) Error vs. DOF

Figure 5-4: Accuracy results for $0 \leq p \leq 3$ for circular channel test case

is located on the actual geometry.

As in Section 5.1, a source term is added to the right-hand side of the compressible Navier-Stokes equations such that the exact solution to the equations is

$$\mathbf{u} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} = \begin{pmatrix} 1 \\ (r - r_i)(r - r_o)\sin\theta \\ (r - r_i)(r - r_o)\cos\theta \\ 5 + \frac{1}{2}(r - r_i)^2(r - r_o)^2 \end{pmatrix}.$$

Figures 5-4(a) and 5-4(b) show the L_2-norm of the error versus number of elements and DOF. As for the the straight channel Poiseuille flow problem, optimal order of accuracy is observed, and the higher-order solutions are seen to provide significantly lower error with fewer DOF. For $p = 3$, only 4000 DOF are required to achieve an error of 1.37×10^{-4}, while for $p = 1$, the error is 2.64×10^{-4} for 7.68×10^4 DOF.

While conducting this test, an unexpected phenomenon was observed. When using $q = 3$ geometry representation, the order of accuracy dropped from $O(h^{p+1})$ to $O(h^{p+1/2})$ for $p = 2$ and $p = 3$. While it is not clear at this time, this problem may be the result of oscillations in the geometry interpolation. Figure 5-5 shows the radius error—the difference between the interpolated radius and the exact radius—for a coarse grid element on the inner wall. The radius error is plotted versus the location, σ, on the reference edge. Clearly, both the $q = 2$ and $q = 3$ interpolations produce oscillatory geometry. In fact, the $q = 2$ oscillations are larger in amplitude. Yet, with $q = 2$ iterpolants, optimal accuracy is achieved. Further work

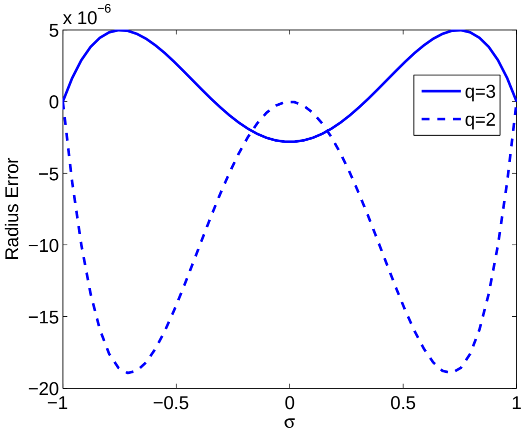


Figure 5-5: Radius error for inner wall, coarse grid boundary face

in this area is necessary to fully understand the requirements on the geometry convergence to achieve optimal order of accuracy. Specifically, it is unclear not only what norm of the geometry error is appropriate but also what rate this norm should converge to guarantee $O(h^{p+1})$ convergence of the solution error.

5.3 NACA 0012 Airfoil

The last test case is that of $M = 0.5$, $Re = 5000$, laminar flow over a NACA 0012 airfoil at $\alpha = 0^o$. Three sets of results are shown: accuracy results, iterative convergence results, and timing results. A set of four structured grids was used to generate the data. The meshes were created by modifying a baseline grid provided by Swanson [43].

The leading edge of the original Swanson mesh was refined by inserting points along the airfoil. The points were used to create new elements having lower aspect ratios than the original elements such that, when the boundary elements were converted from $q = 1$ to $q = 3$ by moving the higher-order points to the true airfoil surface, no negative area elements were created. Then, to create a family of meshes with varying h, the mesh was coarsened by removing every other node in both the streamwise and normal directions. The coarsening procedure was repeated three times, resulting in four, nested meshes containing 672, 2688, 10752, and 43008 elements. The coarsest mesh is shown in Figure 5-6. On the airfoil boundary, $q = 3$ elements were used. The boundary nodes were placed using a modified version of the analytic definition of the NACA 0012:

$$y = \pm 0.6(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1036x^4). \tag{5.2}$$

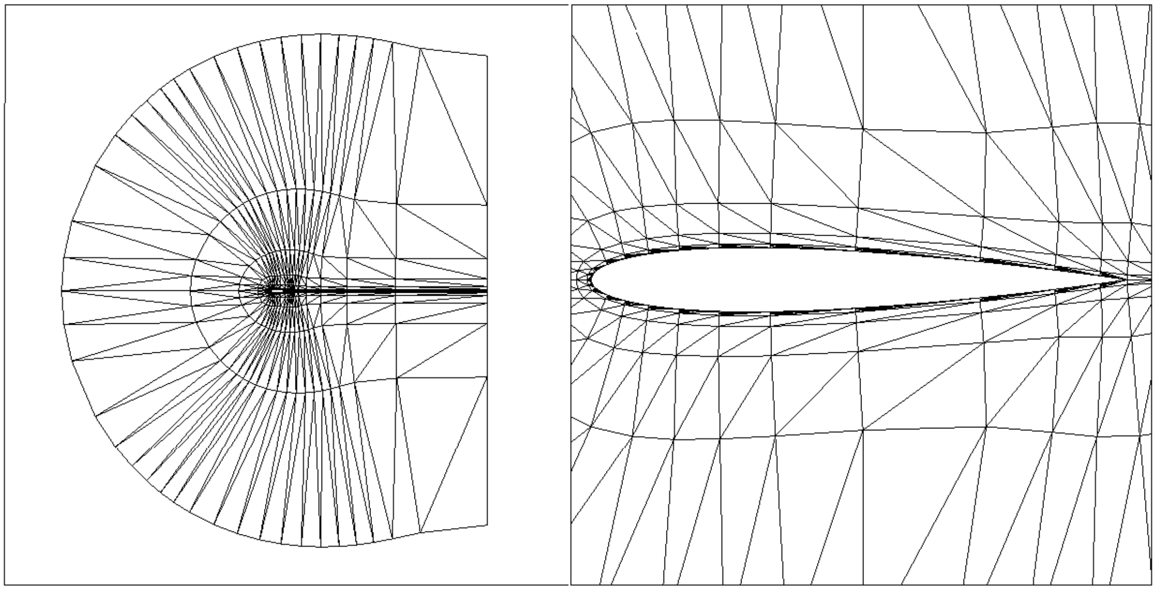


Figure 5-6: Coarse NACA 0012 grid, 672 elements

In Eqn. 5.2, the x^4 coefficient of the traditional definition of the NACA 0012 has been modified such that the trailing edge ($x = 1$) has zero thickness.

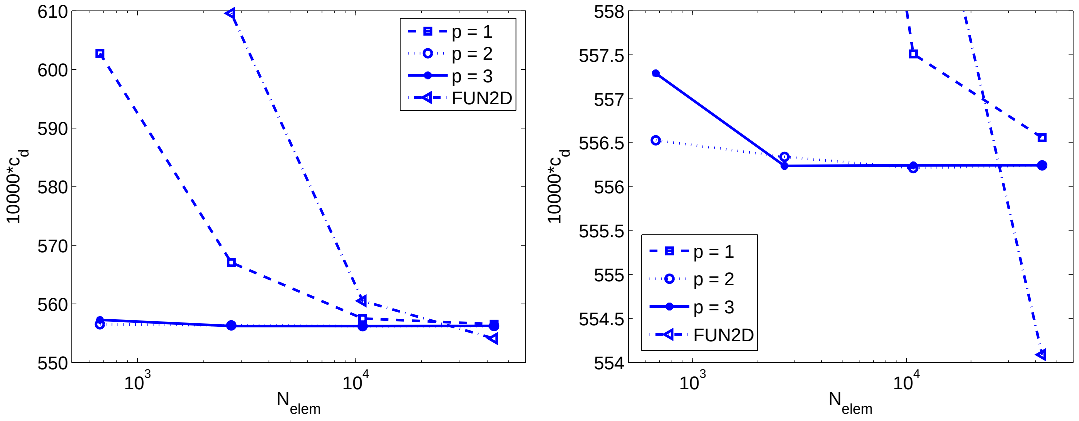
On the airfoil, the no slip, adiabatic boundary condition is used. At the exit plane, the subsonic outflow condition is imposed, thus, the pressure is set to the freestream value. Along the rest of the boundary, the full state condition is used, where the boundary state is set to the freestream state.

Unless otherwise noted, all results were obtained using the multigrid solver with element-line Jacobi smoothing. All computations were initialized from a fully converged $p = 0$ solution. When starting higher-order calculations, five V-cycles on each level were run before prolongating to the next finer level, and each V-cycle contained four pre- and post-smoothing iterations with 100 smoothing iterations on the coarsest level ($p = 0$).

5.3.1 Accuracy Results

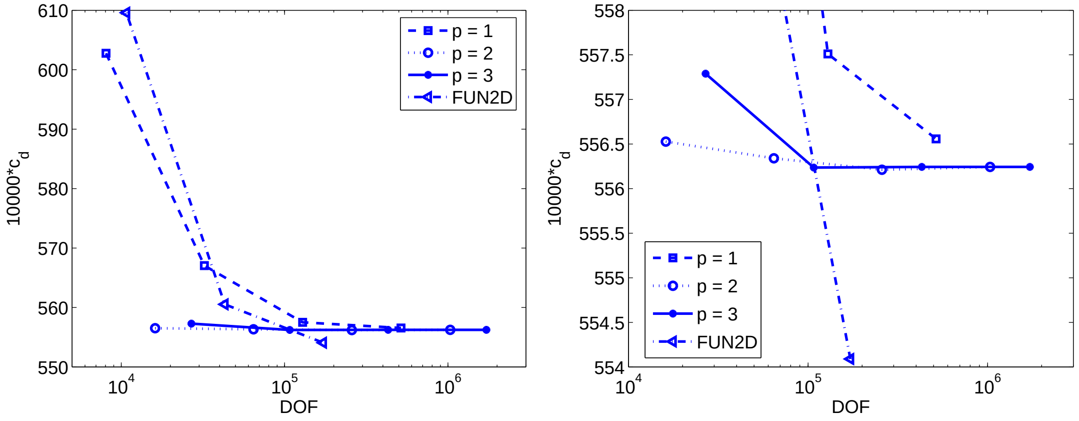
The computed drag for each grid and interpolation order is shown in Figure 5-7. Figure 5-8 shows the drag versus DOF. For comparison, results computed by FUN2D, which uses a node-centered finite volume algorithm, on the same meshes are shown. While it appears that FUN2D may be converging to a different final answer for the drag, qualitatively, the convergence of the FUN2D and $p = 1$ methods in terms of DOF is very similar.

However, the $p = 2$ and $p = 3$ results are dramatically better. On the 2688 element



(a) All results (b) Close-up of (a)

Figure 5-7: Drag versus number of elements



(a) All results (b) Close-up of (a)

Figure 5-8: Drag versus degrees of freedom

grid, both the $p = 2$ (6.45×10^4 DOF) and $p = 3$ (1.08×10^5 DOF) drag results are within 0.1 drag counts of the finest grid, $p = 3$ result. Alternatively, the drag error for $p = 1$ on the 43008 element mesh (5.16×10^5 DOF) is 0.3 drag counts. Thus, high-order solutions provide significantly better accuracy in fewer DOF than $p = 1$ solutions on very refined meshes.

It should be noted that in the results presented here, the drag error is not converging at $O(h^{p+1})$. While the $p = 1$ results are close—the convergence rate based on the $p = 1$, 10752 and 43008 element grids is 1.78—the $p = 2$ and $p = 3$ results are not. The loss of optimal order of accuracy could be caused by many factors. First, as noted in Section 5.2, the discretization is very sensitive to geometry errors. While a $q = 3$ geometry representation was used, the interpolated geometry is oscillatory, and it is possible that the geometry errors converging at suboptimal order of accuracy could be affecting the order of accuracy of the solution. Second, the smoothness of the exact solution could be affecting the order of accuracy. As is well known, the exact solution has a singularity at the trailing edge. No effort has been made here to over-refine the trailing edge region, and, thus, this singularity may be adversely affecting the convergence rate.

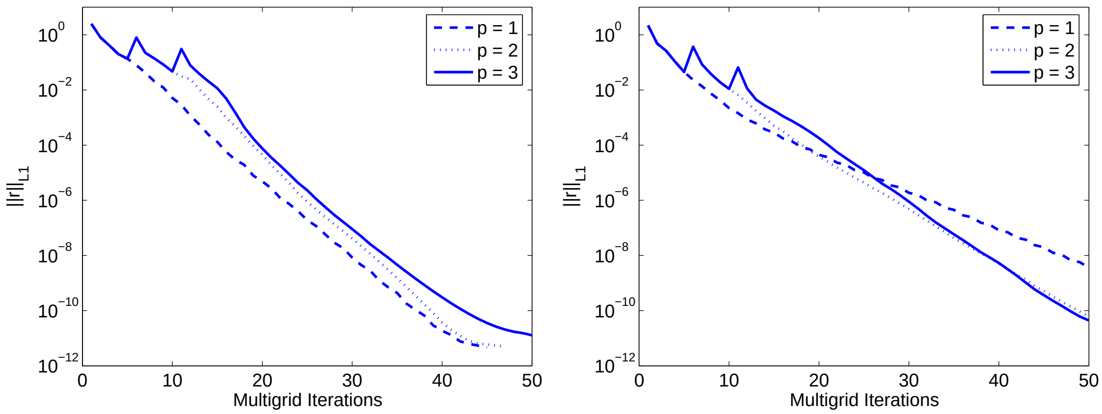
While the above factors appear to be the most likely culprits, many other problems could be degrading the order of accuracy. For example, the boundary conditions have not been thoroughly investigated, and the grid spacing does not vary smoothly throughout the domain. Finally, it should be noted that, while optimal order of accuracy has been proven for DG for linear hyperbolic and linear elliptic problems, no proof currently exists for the Navier-Stokes equations.

5.3.2 Iterative Convergence Results

Iterative convergence results are shown in Figures 5-9 and 5-10. Residual convergence for $1 \leq p \leq 3$ are shown for three meshes in Figure 5-9. The spikes clearly observable for $p = 2$ at five iterations and for $p = 3$ at both five and ten iterations are where the solver has begun to run V-cycles on the next finer level.

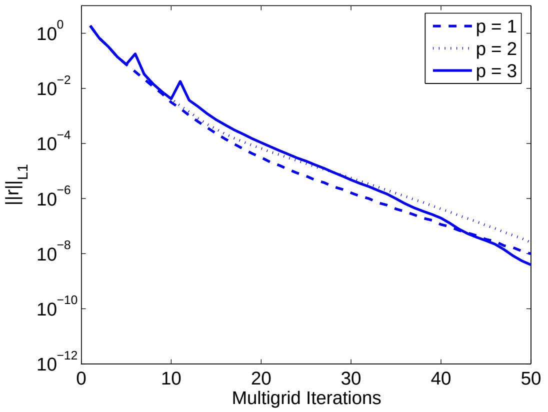
As demonstrated in [22] for the Euler equations, the asymptotic convergence rate of the residual is approximately independent of p. However, a slight h dependence is observed. Thus, while the residual on the 672 element mesh is fully converged in less than 50 iterations, the residual is approximately 10^{-8} after 50 iterations on the 10752 element mesh.

Figure 5-10 shows the drag history. At each iteration, the iterative drag error—the difference between the drag at that iteration and the final converged drag for that case—is plotted. Note that, unlike the residual convergence histories, only V-cycles on the finest



(a) 672 Elements

(b) 2688 Elements

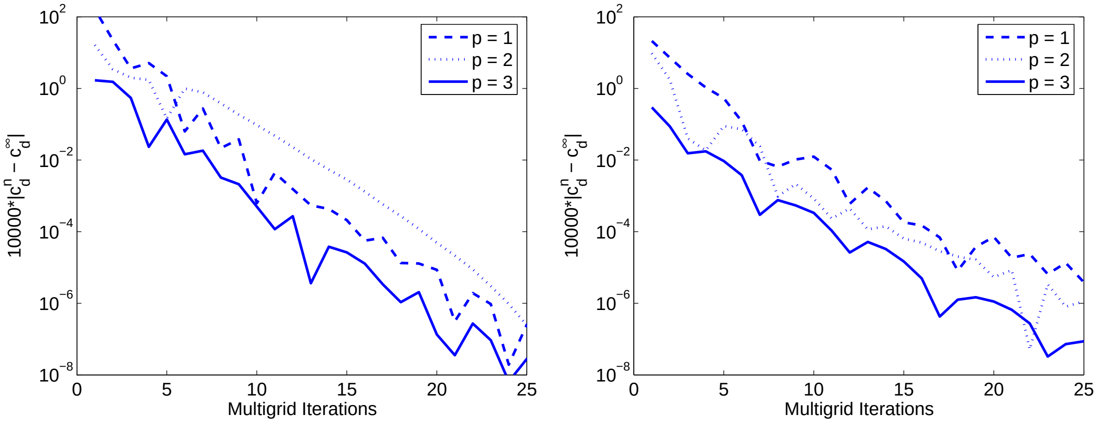


(c) 10752 Elements

Figure 5-9: Residual convergence history

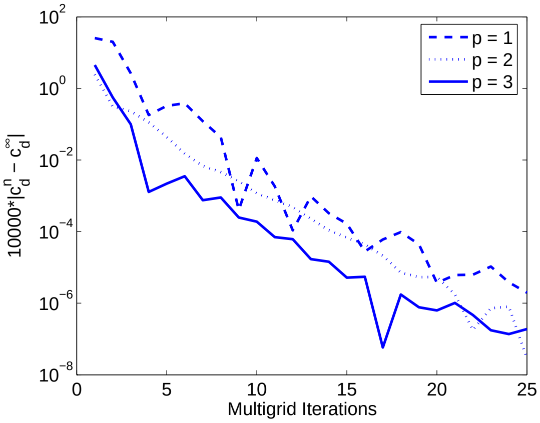
level are shown.

The results show that the drag converges to engineering tolerance much faster than the residual converges to zero. In fact, the iterative drag error usually converges to less than 0.01 drag counts in less than 10 multigrid iterations on the finest level, and more than 15 are never required. In practice, this means that the calculation may be stopped before the residual has converged to zero.



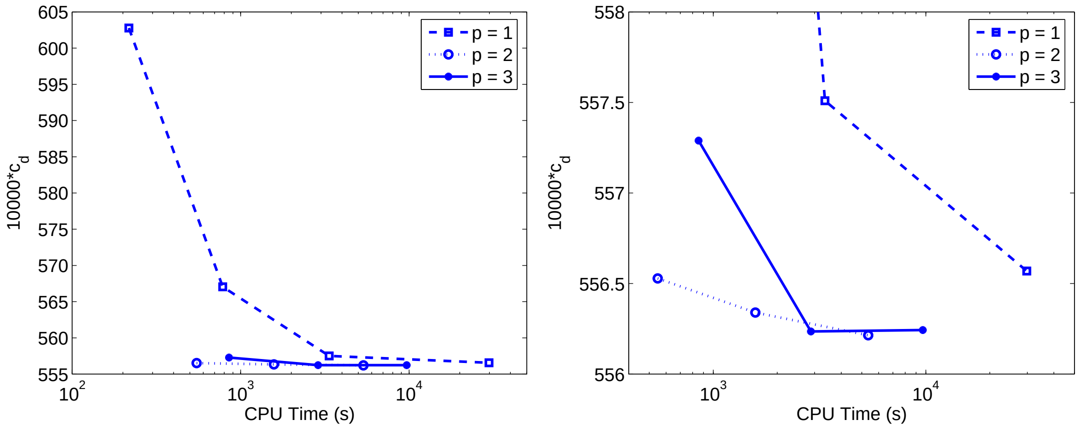
(a) 672 Elements

(b) 2688 Elements



(c) 10752 Elements

Figure 5-10: Drag convergence history



(a) All results

(b) Close-up of (a)

Figure 5-11: Absolute drag versus CPU time

5.3.3 Timing Results

Two sets of timing results are presented: a comparison of the time required to compute the drag to engineering tolerance using multigrid with element-line Jacobi smoothing for varying p and h and a comparison of the time required to drive the residual to zero using the element Jacobi smoother, element-line Jacobi smoother, multigrid with element Jacobi smoothing, and multigrid with element-line Jacobi smoothing. All timing study calculations were run on a single 3.0 gigahertz Intel Pentium 4 processor with 896 megabytes of memory.

Figure 5-11 shows the time required to compute the drag using multigrid with element-line Jacobi smoothing for varying p and h, and Figure 5-12 shows the drag error versus time, where drag error is the difference between the drag for a given case and the $p = 3$ result on the 43008 element grid. The values plotted were obtained using a drag based stopping criterion. Namely, when the change in the drag for one iteration was less that 0.01 drag counts, $10000|c_d^{n+1} - c_d^n| < 0.01$, for four iterations, the calculation was stopped and the drag and time were plotted. This stopping criterion is intended to minimize the computational time required to compute the drag given the knowledge that the drag converges to engineering tolerance faster than the residual converges to zero. Furthermore, it is intended to simulate engineering practices in that it does not require knowledge of the final, converged drag.

Clearly, the higher-order solutions give smaller error in less time than the $p = 1$ solution. It takes approximately 2.98×10^5 seconds to obtain the most accurate $p = 1$ solution, which has an error of 0.32 drag counts. Alternatively, the coarsest $p = 2$ solution has an error of

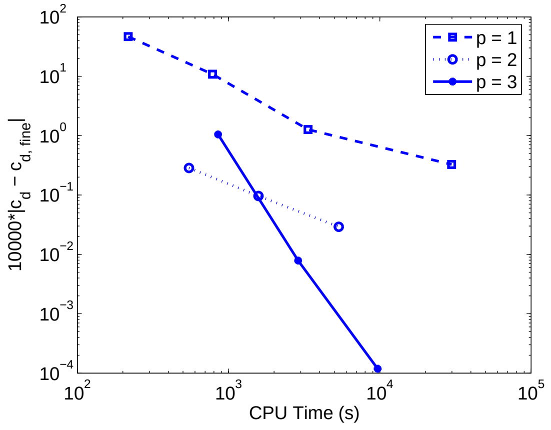
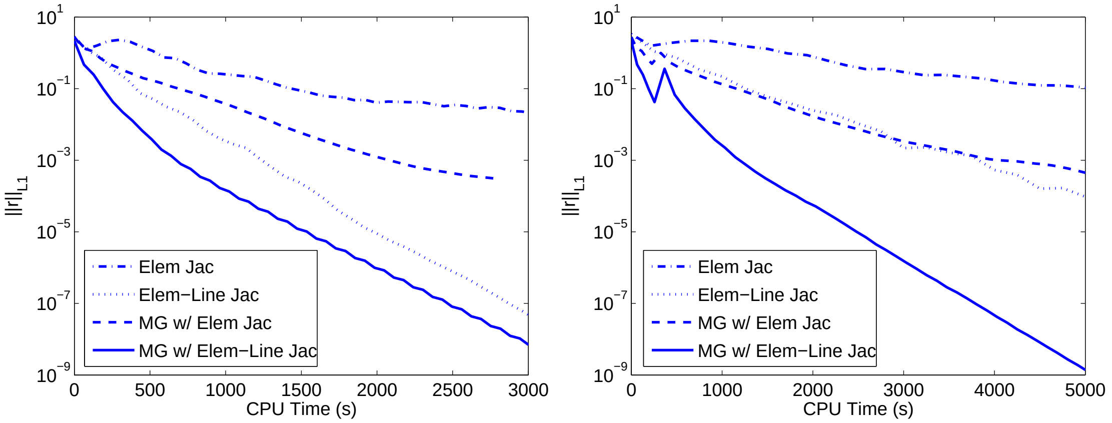


Figure 5-12: Drag error versus CPU time

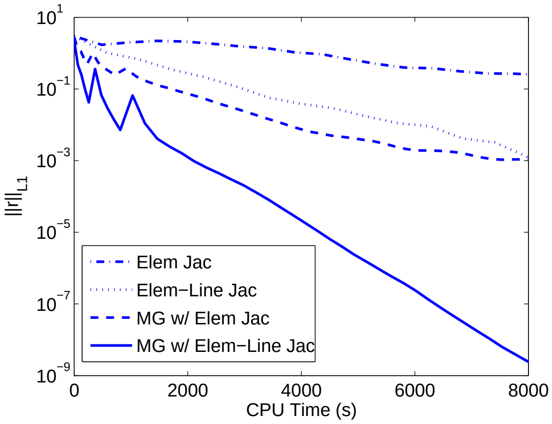
0.29 counts and was obtained in only 548 seconds.

Figure 5-13 compares the four solvers developed in this work: element Jacobi, element-line Jacobi, multigrid with element Jacobi smoothing, and multigrid with element-line Jacobi smoothing. As expected, the figure shows that the incorporation of the element-line Jacobi smoother significantly improves the performance of the solver. Thus, the element-line Jacobi solver is more efficient than the element Jacobi solver, and multigrid with element-line Jacobi smoothing is more efficient that multigrid with element Jacobi smoothing for all p. Furthermore, in all cases, multigrid with element-line Jacobi smoothing is the best solver, and, as p increases, it becomes relatively more efficient. For $p = 1$, the multigrid with element-line Jacobi smoothing is only slightly more efficient than pure element-line Jacobi. After 3000 seconds, the residual is approximately one order of magnitude lower using multigrid. However, for $p = 3$ the element-line Jacobi and multigrid with element Jacobi schemes both reduce the residual to approximately 10^{-3} in 8000 seconds while multigrid with element-line Jacobi drives the residual to 10^{-9} in the same time.



(a) $p = 1$

(b) $p = 2$



(c) $p = 3$

Figure 5-13: Solver comparison for 2688 element mesh

Chapter 6

Conclusions

In an effort to advance the state of the art in CFD in applied aerodynamics, a high-order algorithm for the compressible Navier-Stokes equations was presented. The algorithm employs a DG discretization with an element-compact stencil and a p-multigrid solver with element-line Jacobi smoothing.

The discretization has been shown to achieve optimal order of accuracy for simple test problems. While optimal order of accuracy was not obtained for the more practical airfoil problem, the higher-order discretizations were shown to provide significantly less error with fewer elements and DOF than second-order discretizations. Furthermore, timing studies showed that the reductions in elements and DOF translated into savings in computational time required to compute the drag to within 0.5 drag counts of an order of magnitude or more.

Via Fourier analysis of convection-diffusion, the element and element-line Jacobi smoothers were shown to be stable independent of order. Furthermore, the residual convergence rate was observed to be independent of order and only weakly dependent on grid spacing when using multigrid with element-line Jacobi smoothing. A timing study on the NACA 0012 test case using all four solvers confirmed that, especially for higher-order discretizations, multigrid with element-line Jacobi smoothing is the most efficient solution technique.

While these results are encouraging, much work remains. First, the 2-D viscous discretization must be extended to the 3-D case. Second, the geometry representation issue highlighted by the circular cylinder test case must be resolved. To obtain high-order accuracy for general problems, a complete understanding of the geometry representation and gridding requirements is crucial. Third, turbulence modeling and limiting capabilities must be added to allow consideration of a wider range of flows of engineering interest. In terms of turbulence modeling, the first model considered will be Spalart-Allmaras as this model is widely used in aerospace applications. In terms of limiting, although it is not yet clear, the most attractive approach may be hp-adaptivity where in the vicinity of a shock h and p are lowered to avoid oscillations in the solution. One hurdle in the introduction of such a limiter is the development of a robust shock detection scheme such that h and p are lowered near the shock but high-order accuracy is maintained in smooth regions.

Finally, optimization is required to make the method practical. This work will entail not only optimization of the current implementation but possibly also the introduction of approximations to lessen the impact of expensive computations. For example, in the current implementation, the block diagonal of the Jacobian is inverted for every element at every iteration. Especially for higher-order, 3-D calculations, this inversion requires significant computational work. Thus, if an approximate inverse could be introduced without severly degrading the convergence rate, significant savings could be obtained. Work in this area has already begun.

Bibliography

[1] S. R. Allmaras. *A coupled Euler/Navier-Stokes algorithm for 2-D unsteady transonic shock/boundary-layer interaction*. PhD thesis, Massachusetts Institute of Technology, 1989.

[2] S. R. Allmaras. Analysis of semi-implicit preconditioners for multigrid solution of the 2-D compressible Navier-Stokes equations. AIAA Paper Number 95–1651-CP, 1995.

[3] S. R. Allmaras and M. B. Giles. A second-order flux split scheme for the unsteady 2-D Euler equations on arbitrary meshes. AIAA 87-1119-CP, 1987.

[4] W. K. Anderson, R. D. Rausch, and D. L. Bonhaus. Implicit/multigrid algorithms for incompressible turbulent flows on unstructured grids. *J. Comput. Phys.*, 128:391–408, 1996.

[5] D. N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM J. Numer. Anal.*, 19:742–760, 1982.

[6] I. Babuska, B. A. Szabo, and I. N. Katz. The p-version of the finite element method. *SIAM J. Numer. Analysis*, 18(3):515–545, 1981.

[7] T. J. Barth. Recent developments in high-order k-exact reconstruction on unstructured meshes. AIAA-93-0668, 1993.

[8] F. Bassi and S. Rebay. A high-order accurate discountinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *J. Comput. Phys.*, 131:267–279, 1997.

[9] F. Bassi and S. Rebay. High-order accurate discountinuous finite element solution of the 2d Euler equations. *J. Comput. Phys.*, 138:251–285, 1997.

[10] F. Bassi and S. Rebay. An implicit high-order discontinuous galerkin method for the steady state compressible navier-stokes equations. In Papailiou; Tsahalis; Periaux;

Hirsh; and Pandolfi, editors, *Computational Fluid Dynamics 98, Proceedings of the Fourth European Computational Fluid Dynamics Conference*, pages 1227–1233. Wiley, New York, 1998.

[11] F. Bassi and S. Rebay. GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations. In Karniadakis Cockburn and Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 197–208. Springer, Berlin, 2000.

[12] F. Bassi and S. Rebay. Numerical solution of the Euler equations with a multiorder discontinuous finite element method. In *Second International Conference on Computational Fluid Dynamics, Sydney, Australia*, 2002.

[13] C. E. Baumann. *An hp-Adaptive Discontinuous Finite Element Method for Computational Fluid Dynamics*. PhD dissertation, University of Texas at Austin, December 1997.

[14] Achi Brandt. *Guide to Multigrid Development*. Springer-Verlag, 1982.

[15] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, New York, 1987.

[16] B. Cockburn, G. E. Karniadakis, and C. W. Shu. The development of discontinuous Galerkin methods. In Karniadakis Cockburn and Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 3–50. Springer, Berlin, 2000.

[17] B. Cockburn and C. W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws II: General framework. *Math. Comp.*, 52:411–435, 1989.

[18] B. Cockburn and C. W. Shu. The Runge-Kutta local projection p^1-discontinuous Galerkin method for scalar conservation laws. *RAIRO Model Math. Anal. Numer.*, 25:337–361, 1991.

[19] B. Cockburn and C. W. Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM J. Numer. Anal.*, 35(6):2440–2463, December 1998.

[20] B. Cockburn and C. W. Shu. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *J. of Sci. Comput.*, 16(3):173–261, September 2001.

[21] D. N. Arnold; F. Brezzi; B. Cockburn; and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39(5):1749–1779, 2002.

[22] K. Fidkowski. A high-order discontinuous Galerkin multigrid solver for aerodynamic applications. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2004.

[23] K. Fidkowski and D. L. Darmofal. Development of a higher-order solver for aerodynamic applications. In *Proceedings AIAA Aerospace Sciences Meeting*, Reno, 2004.

[24] D. Gottlieb and S. A. Orszag. *Numerical Analysis of Spectral Methods.* SIAM, Philadelphia, 1977.

[25] D. W. Halt. *A compact higher-order Euler solver for unstructured grids.* PhD thesis, Washington University, 1992.

[26] B. T. Helenbrook, D. J. Mavriplis, and H. A. Atkins. Analysis of p-multigrid for continuous and discontinuous finite element discretizations. AIAA Paper 2003-3989, 2003.

[27] T. J. R. Hughes, L.P. Franca, and G.M. Hulbert. A new finite element formulation for computational fluid dynamics: VIII The Galerkin/least-squares method for advective-diffusive systems. *Comput. Methods Appl. Mech. Engrg.*, 73:173–189, 1989.

[28] A. Jameson. Solution of the Euler equations for two-dimensional transonic flow by a multigrid method. *Applied Mathematics and Computation*, 13:327–356, 1983.

[29] C. Johnson and J. Pitkaranta. An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation. *Math. Comp.*, 46:1–26, 1986.

[30] K. Z. Korczak and A. T. Patera. An isoparametric spectral element method for solution of the Navier-Stokes equations in complex geometry. *J. Comput. Phys.*, 62:361–382, 1984.

[31] Kelly R. Laflin, John C. Vassberg, Richard A. Wahls, Joseph H. Morrison, Olaf Brodersen, Mark Rakowitz, Edward N. Tinoco, and Jean-Luc Godard. Summary of data from the Second AIAA CFD Drag Prediction Workshop. AIAA Paper 2004-0555, 2004.

[32] E. M. Lee-Rausch, N. T. Frink, D. J. Mavriplis, R. D. Rausch, and W. E. Milholen. Transonic drag prediction on a DLR-F6 transport configuration using unstructured grid solvers. AIAA-2004-0554, 2004.

[33] S. K. Lele. Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.*, 103:16–42, 1992.

[34] P. LeSaint and P. A. Raviart. On a finite element method for solving the neutron transport equation. In C. de Boor, editor, *Mathematical Aspects of finite elements in partial differential equations*, pages 89–145. Academic Press, 1974.

[35] David W. Levy, Thomas Zickuhr, John Vassberg, Shreekant Agrawal, Richard A. Wahls, Shahyar Pirzadeh, and Michael J. Hemsch. Data summary from the First AIAA Computational Fluid Dynamics Drag Prediction Workshop. *Journal of Aircraft*, 40(5):875–882, 2003.

[36] D. J. Mavriplis. Multigrid solution of the 2-D Euler equations on unstructured triangular meshes. *AIAA Journal*, 26:824–831, 1988.

[37] D. J. Mavriplis. Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes. *J. Comput. Phys.*, 145:141–165, 1998.

[38] D. J. Mavriplis and S. Pirzadeh. Large-scale parallel unstructured mesh computations for 3-D high-lift analysis. *AIAA Journal of Aircraft*, 36:987–998, 1999.

[39] Tolulope O. Okusanya. *Algebraic Multigrid for Stabilized Finite Element Discretizations of the Navier-Stokes Equations.* PhD dissertation, M.I.T., Department of Aeronautics and Astronautics, June 2002.

[40] A. T. Patera. A spectral element method for fluid dynamics: laminar flow in a channel expansion. *J. Comput. Phys.*, 54:468–488, 1984.

[41] Niles A. Pierce and Michael B. Giles. Preconditioned multigrid methods for compressible flow calculations on stretched meshes. *J. Comput. Phys.*, 136:425–445, 1997.

[42] F. Brezzi; G. Manzini; D. Marini; P. Pietra; and A. Russo. Discontinuous Galerkin approximations for elliptic problems. *Numer. Methods for Partial Differential Eqns.*, 16:365–378, 2000.

[43] R. Radespiel and R.C. Swanson. An investigation of cell centered and cell vertex multigrid schemes for the Navier-Stokes equations. AIAA Paper Number 89-0453, 1989.

[44] W. H. Reed and T. R. Hill. Triangular mesh methods for the neutron transport equation. Technical Report Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[45] G. R. Richter. An optimal-order error estimate for the discontinuous Galerkin method. *Math. Comp.*, 50:75–88, 1988.

[46] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *J. Comput. Phys.*, 43:357–372, 1981.

[47] P. L. Roe. Characteristic-based schemes for the Euler equations. *Ann. Rev. Fluid Mech.*, 18:337–65, 1986.

[48] E. M. Rønquist and A. T. Patera. Spectral element multigrid I. Formulation and numerical results. *J. Sci. Comput.*, 2(4):389–406, 1987.

[49] C. W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *J. Comput. Phys.*, 77:439–471, 1988.

[50] B. van Leer. Towards the ultimate conservative difference scheme. IV. a new approach to numerical convection. *J. Comput. Phys.*, 23:276–299, 1977.

[51] B. Van Leer. Flux-vector splitting for the Euler equations. Technical Report 81-11, ICASE, 1981.

[52] B. Van Leer. Upwind-difference methods for aerodynamic problems governed by the Euler equations. *Lectures in Applied Mathematics*, 22, 1985.

[53] B. Van Leer, J. Thomas, P. Roe, and R. Newsome. A comparison of numerical flux formulas for the Euler and Navier-Stokes equations. AIAA Paper 87-1104, 1987.

[54] John C. Vassberg, Mark A. DeHaan, and Tony J. Sclafani. Grid generation requirements for accurate drag predictions based on OVERFLOW calculations. AIAA-2003-4124, 2003.

[55] V. Venkatakrishnan, S. R. Allmaras, D. S. Kamenetskii, and F. T. Johnson. Higher order schemes for the compressible Navier-Stokes equations. AIAA Paper 2003-3987, 2003.

[56] M. R. Visbal and D. V. Gaitonde. On the use of higher-order finite-difference schemes of curvilinear and deforming meshes. *J. Comput. Phys.*, 181:155–185, 2002.

[57] Z. J. Wang. Spectral (finite) volume method for conservation laws on unstructured grids. Basic formulation. *J. Comput. Phys.*, 178:210–251, 2002.

[58] M. Wheeler. An elliptic collocation-finite element method with interior penalties. *SIAM J. Numer. Anal.*, 15:152–161, 1978.

[59] D. W. Zingg, S. De Rango, M. Nemec, and T. H. Pulliam. Comparison of several spatial discretizations for the Navier-Stokes equations. *J. Comput. Phys.*, 160:683–704, 2000.