

**A High-Order, Adaptive, Discontinuous Galerkin Finite
Element Method for the Reynolds-Averaged Navier-Stokes
Equations**

by

Todd A. Oliver

S.M., Massachusetts Institute of Technology (2004)

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

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

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2008

© Massachusetts Institute of Technology 2008. All rights reserved.

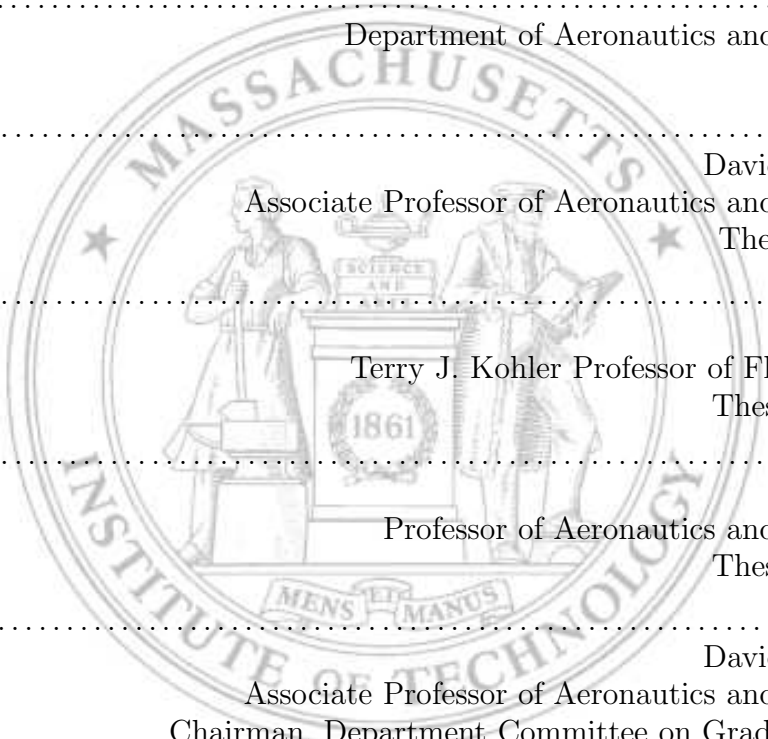
Author
Department of Aeronautics and Astronautics
July 3, 2008

Certified by
David L. Darmofal
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by
Mark Drela
Terry J. Kohler Professor of Fluid Dynamics
Thesis Committee

Certified by
Jaime Peraire
Professor of Aeronautics and Astronautics
Thesis Committee

Accepted by
David L. Darmofal
Associate Professor of Aeronautics and Astronautics
Chairman, Department Committee on Graduate Students

The seal of the Massachusetts Institute of Technology is a large, faint watermark in the background. It features a circular border with the text "MASSACHUSETTS INSTITUTE OF TECHNOLOGY" and "1861". In the center, there is a figure of a person standing at a podium with a book, and another figure standing to the right. The motto "MENS ET MANUS" is written on a banner at the bottom.

A High-Order, Adaptive, Discontinuous Galerkin Finite Element Method for the Reynolds-Averaged Navier-Stokes Equations

by

Todd A. Oliver

Submitted to the Department of Aeronautics and Astronautics
on July 3, 2008, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis presents high-order, discontinuous Galerkin (DG) discretizations of the Reynolds-Averaged Navier-Stokes (RANS) equations and an output-based error estimation and mesh adaptation algorithm for these discretizations. In particular, DG discretizations of the RANS equations with the Spalart-Allmaras (SA) turbulence model are examined. The dual consistency of multiple DG discretizations of the RANS-SA system is analyzed. The approach of simply weighting gradient dependent source terms by a test function and integrating is shown to be dual inconsistent. A dual consistency correction for this discretization is derived. The analysis also demonstrates that discretizations based on the popular mixed formulation, where dependence on the state gradient is handled by introducing additional state variables, are generally asymptotically dual consistent. Numerical results are presented to confirm the results of the analysis.

The output error estimation and output-based adaptation algorithms used here are extensions of methods previously developed in the finite volume and finite element communities. In particular, the methods are extended for application on the curved, highly anisotropic meshes required for boundary conforming, high-order RANS simulations. Two methods for generating such curved meshes are demonstrated. One relies on a user-defined global mapping of the physical domain to a straight meshing domain. The other uses a linear elasticity node movement scheme to add curvature to an initially linear mesh. Finally, to improve the robustness of the adaptation process, an “unsteady” algorithm, where the mesh is adapted at each time step, is presented. The goal of the unsteady procedure is to allow mesh adaptation prior to converging a steady state solution, not to obtain a time-accurate solution of an unsteady problem. Thus, an estimate of the error due to spatial discretization in the output of interest averaged over the current time step is developed. This error estimate is then used to drive an h -adaptation algorithm.

Adaptation results demonstrate that the high-order discretizations are more efficient than the second-order method in terms of degrees of freedom required to achieve a desired error tolerance. Furthermore, using the unsteady adaptation process, adaptive RANS simulations may be started from extremely coarse meshes, significantly decreasing the mesh generation burden to the user.

Thesis Supervisor: David L. Darmofal

Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

To begin, I would like to express my gratitude to my advisor, Prof. David Darmofal. Without his guidance and encouragement throughout my time as a graduate student, this work would not have been possible. His probing questions and insightful ideas contributed immeasurably to the research presented here, and his mentoring has been crucial to my development as a researcher.

In addition, I would like to thank my committee members, Prof. Mark Drela and Prof. Jaime Peraire, for their criticism and feedback, which led to many improvements in my research and this thesis. I would also like to thank Dr. Ralf Hartmann for his comments on the initial draft of the thesis, and I am particularly indebted to Dr. Steve Allmaras for his help with the SA model modifications and for his insightful comments on the initial thesis draft.

Of course, none of this work would have taken place without the entire Project X team. I would particularly like to thank Garrett Barter, Krzysztof Fidkowski, Mike Park, Robert Haines, Laslo Diosady, JM Modisette, and Josh Krakos for their contributions to the code as well as their input throughout many discussions of research, DG methods, software practices, and Project X. In addition to those already mentioned, Project X would not be what it is now without the contributions of Mathieu Serrano, Michael Brasher, James Lu, Paul Nicholson, Eric Liu, Eleanor Lin, Peter Whitney, Shannon Cheng, Jean-Baptiste Brachet, Haufei Sun, and Masayuki Yano, and I am looking forward to following the evolution of the project in the capable hands of Laslo, JM, Josh, Haufei, and Masa.

On a more personal note, I have sincerely enjoyed the many friendships I developed through both my undergraduate and graduate years at MIT. In particular, Dave Bennett, Garrett Barter, Mike Brasher, Mark Monroe, and Shana Diez have all helped to make MIT more fun than it might otherwise have been.

I would also like to thank my family—Mom, Dad, Lee, and Lauren—for their love, support, and encouragement. Lauren, I have especially enjoyed having you living in Boston for the past three years, and I will miss you. However, I am sincerely looking forward to leaving for Austin to be close to the rest of the family.

I must thank my wife, Christy, whose love and support have been most important of all. I do not know how to fully express my gratitude. I can only say that you have made the past two years the best of my life, and I am looking forward to continuing our journey together.

Finally, I would like to acknowledge the financial support I have received throughout my graduate studies. In particular, my work was supported by the National Defense Science and Engineering Graduate Fellowship, the U. S. Air Force Research Laboratory (USAF-3306-03-SC-0001), and The Boeing Company.

Contents

1	Introduction	19
1.1	Objective	20
1.2	Previous Work	20
1.2.1	High-Order Methods	20
1.2.2	Discontinuous Galerkin Methods	21
1.2.3	Error Estimation and Adaptation	23
1.3	Thesis Overview	26
2	Discretization of the RANS Equations	29
2.1	The RANS Equations	29
2.2	The SA Turbulence Model	30
2.2.1	Baseline Model	30
2.2.2	Negative $\tilde{\nu}$ Modifications	32
2.2.3	Comparison of SA Model Versions	35
2.3	Spatial Discretization	39
2.3.1	Inviscid Discretization	41
2.3.2	Viscous Discretization	41
2.3.3	Source Discretization	42
2.4	Temporal Discretization	43
2.5	Solution Technique	45
3	Dual Consistency Analysis	47
3.1	Review of Dual Consistency	47
3.1.1	Definition	47
3.1.2	Importance	49
3.2	Analysis of DG Discretizations of Source Terms	50
3.2.1	The Standard Weighting and Dual Consistent Methods	51
3.2.2	The Mixed Formulation	54

3.3	Model Problem Results	59
3.4	RANS Results	66
3.4.1	Flat Plate Adjoint Results	66
3.4.2	NACA 0012 Refinement Study	71
3.4.3	RAE 2822 Refinement Study	75
4	Output Error Estimation	81
4.1	Motivation	81
4.2	Implementation	82
4.3	Numerical Results	85
4.3.1	Flat Plate	85
4.3.2	NACA 0012	87
4.3.3	RAE 2822	93
5	Output-Based Adaptation	99
5.1	Algorithm	99
5.1.1	Anisotropy Detection	100
5.1.2	Mesh Optimization	104
5.1.3	Curved Mesh Generation	106
5.2	Numerical Results	109
5.2.1	Flat Plate	109
5.2.2	Ellipse	112
5.2.3	NACA 0012	121
6	Unsteady Adaptation Algorithm	129
6.1	Motivation	129
6.2	Unsteady Algorithm	129
6.2.1	Error Estimation	130
6.2.2	Comparison with Standard Algorithm	131
6.2.3	Extension to Variable Time Step	133
6.3	Numerical Results	133
6.3.1	Flat Plate	134
6.3.2	Ellipse	137
6.3.3	Advanced Energy Efficient Transport Three-Element Airfoil	137
7	Conclusions and Outlook	147
7.1	Summary and Conclusions	147

7.2	Future Work	148
7.2.1	Turbulence Model Improvements	149
7.2.2	Discretization Robustness Improvement	149
7.2.3	Standard Adaptation Algorithm Modifications	149
7.2.4	Unsteady Adaptation Refinements	150
A	Derivation of the RANS Equations	153
B	Laminar NACA 0012 Results	157
C	<i>A Priori</i> Output Error Estimation	161
C.1	Linear Analysis	161
C.2	Nonlinear Analysis	163
D	Error Convergence Rates for Adaptation	167
E	Boundary Conditions	171
E.1	Farfield, Full State Boundary	171
E.2	Subsonic Inflow: $T_t, p_t, \alpha, \rho\tilde{\nu}$	171
E.3	Subsonic Outflow: p	172
E.4	No Slip, Adiabatic Wall	172
E.5	Symmetry Plane	173

List of Figures

2-1	SA model f_{v2} closure function versus χ	32
2-2	Comparison of baseline and modified SA model diffusion terms	36
2-3	Turbulence model working variable profiles for flow over a flat plate, computed using three versions of the SA model on the coarse mesh	37
2-4	Turbulence model working variable profiles for flow over a flat plate, computed using three versions of the SA model on the fine mesh	38
2-5	Velocity profiles for flow over a flat plate at $Re_x = 5 \times 10^6$, computed using three versions of the SA model on the coarse mesh	39
2-6	Skin friction distributions for flow over a flat plate, computed using three versions of the SA model on the coarse mesh	40
2-7	Illustration of the map f_κ from the reference element to the element κ in physical space	40
3-1	Primal error in the broken H^1 norm for the scalar model problem	61
3-2	Primal error in the L^2 norm for the scalar model problem	62
3-3	Adjoint error in the broken H^1 norm for the scalar model problem	63
3-4	Adjoint error in the L^2 norm for the scalar model problem	64
3-5	Functional output error for the scalar model problem	65
3-6	Comparison of x -momentum adjoint solution profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$ for flow over a flat plate	67
3-7	Comparison of turbulence model adjoint solution profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$	68
3-8	Comparison of x -momentum adjoint y -derivative profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$ for flow over a flat plate	69

3-9	Comparison of turbulence model adjoint y -derivative profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$ for flow over a flat plate	70
3-10	Coarse (1280 elements), linear mesh of the NACA 0012 airfoil	72
3-11	Leading edge of the $q = 1$ and $q = 3$ versions of the coarse mesh of the NACA 0012 airfoil	72
3-12	Order of accuracy obtained for $p = 1, 2, 3, 4$ versus order of accuracy assumed for $p = 5$ for drag error for flow over a NACA 0012 airfoil, computed using the dual consistent discretization	73
3-13	Drag error for flow over a NACA 0012 computed using the dual consistent discretization	74
3-14	Drag error for flow over a NACA 0012 computed using the standard weighting discretization	76
3-15	Coarse (4000 elements), linear mesh of the RAE 2822 airfoil	78
3-16	Drag error for flow over a RAE 2822 computed using the dual consistent discretization	79
4-1	The patch of elements centered at κ , denoted $\mathcal{P}(\kappa)$, highlighted in blue . . .	83
4-2	Drag error for the dual consistent discretization versus uniform grid refinement for the flat plate test case	86
4-3	Primal and dual residual drag error estimates for the dual consistent discretization versus uniform grid refinement for the flat plate test case	86
4-4	Maximum error estimate and effectivity for the dual consistent discretization versus uniform grid refinement for the flat plate test case	87
4-5	Drag error for the standard weighting discretization versus uniform grid refinement for the flat plate test case	88
4-6	Primal and dual residual drag error estimates for the standard weighting discretization versus uniform grid refinement for the flat plate test case . . .	88
4-7	Maximum error estimate and effectivity for the standard weighting discretization versus uniform grid refinement for the flat plate test case	89
4-8	Primal and dual residual drag error estimates for the dual consistent discretization versus uniform grid refinement for the NACA 0012 test case . .	90
4-9	Maximum error estimate and effectivity for the dual consistent discretization versus uniform grid refinement for the NACA 0012 test case	91
4-10	Elemental error estimate (left) and Mach number (right) for the $p = 2$, dual consistent discretization on the coarse mesh for the NACA 0012 test case .	92

4-11	Elemental error estimate (left) and Mach number (right) for the $p = 4$, dual consistent discretization on the coarse mesh for the NACA 0012 test case	92
4-12	Primal and dual residual drag error estimates for the standard weighting discretization versus uniform grid refinement for the NACA 0012 test case	93
4-13	Maximum error estimate and effectivity for the standard weighting discretization versus uniform grid refinement for the NACA 0012 test case	94
4-14	Primal and dual residual drag error estimates for the dual consistent discretization versus uniform grid refinement for the RAE 2822 test case	95
4-15	Maximum error estimate and effectivity for the dual consistent discretization versus uniform grid refinement for the RAE 2822 test case	95
4-16	Elemental error estimate for the $p = 3$, dual consistent discretization on the coarse and fine meshes for the RAE 2822 test case	97
4-17	Elemental error estimate at the trailing edge for the $p = 3$, dual consistent discretization on the fine mesh for the RAE 2822 test case	97
5-1	Illustration of transfer of desired mesh anisotropy from reference to physical space for two dimensional case	103
5-2	Mappings from the reference element to the unit equilateral triangle and physical space	105
5-3	Illustration of the global map, g , from meshing to physical domain	107
5-4	Estimated and actual drag error versus DOF for adaptation on the drag on a flat plate, comparison of $p = 1, 2, 3$ results	110
5-5	Estimated and actual drag error versus DOF for adaptation on the drag on a flat plate, comparison of standard weighting (SW), dual consistent (DC) and mixed formulation (MF) discretizations	111
5-6	Skin friction distributions for flow over a flat plate computed using the standard weighting discretization	113
5-7	Skin friction distributions for flow over a flat plate computed using the dual consistent discretization	114
5-8	Skin friction distributions for flow over a flat plate computed using the mixed formulation discretization	115
5-9	Initial mesh (1600 elements) for the ellipse adaptation test case, shown in both the meshing and physical spaces	116
5-10	Estimated and actual drag error versus DOF for adaptation on the drag on an ellipse	117

5-11	Pressure coefficient distributions on initial and final meshes for flow over an ellipse	118
5-12	Skin friction distributions on initial and final meshes for flow over an ellipse	119
5-13	Initial (left) and final (right) meshes for the $p = 3$ adaptation on the ellipse test case	120
5-14	Estimated and actual drag error versus degrees DOF for adaptation on the drag on a NACA 0012 in $Re_c = 1 \times 10^6$ flow	121
5-15	Pressure coefficient distributions on initial and final meshes for $Re_c = 1 \times 10^6$ flow over a NACA 0012	123
5-16	Skin friction distributions on initial and final meshes for $Re_c = 1 \times 10^6$ flow over a NACA 0012	124
5-17	Estimated and actual drag error versus DOF for adaptation on the drag on a NACA 0012 in $Re_c = 1 \times 10^7$ flow	125
5-18	Pressure coefficient distributions on initial and final meshes for $Re_c = 1 \times 10^7$ flow over a NACA 0012	126
5-19	Skin friction distributions on initial and final meshes for $Re_c = 1 \times 10^7$ flow over a NACA 0012	127
6-1	Initial mesh for unsteady adaptation applied to flow over a flat plate	134
6-2	Unsteady adaptation history for the flat plate test case	135
6-3	Finest and final meshes for unsteady adaptation using uniform time step applied to flow over a flat plate	136
6-4	Final mesh (252 elements) for unsteady adaptation using variable time step applied to flow over a flat plate	136
6-5	Initial mesh for unsteady adaptation applied to flow over an ellipse	138
6-6	Unsteady adaptation history for the ellipse test case	139
6-7	Finest and final meshes for unsteady adaptation applied to flow over an ellipse	140
6-8	Initial mesh for unsteady adaptation applied to flow over the EET 3-element airfoil	141
6-9	Mesh obtained by unsteady adaptation applied to flow over the EET 3-element airfoil	142
6-10	Mesh obtained by a standard adaptation iteration for flow over the EET 3-element airfoil	142
6-11	Boundary layer mesh on slat before and after standard adaptation for EET 3-element airfoil	144
6-12	Finest solution ($p = 3$, 11620 elements) for the EET 3-element airfoil	145

6-13	Surface pressure coefficient distribution for finest solution ($p = 3$, 11620 elements) for the EET 3-element airfoil	146
B-1	Drag error for laminar flow over a NACA 0012	158
B-2	Elemental error estimate for the $p = 4$ discretization on the coarse and fine meshes for the laminar NACA 0012 test case in the near field	159
B-3	Elemental error estimate for the $p = 4$ discretization on the coarse and fine meshes for the laminar NACA 0012 test case for the entire domain	159

List of Tables

D.1	Error convergence versus DOF for the ellipse test case	169
D.2	Error convergence versus DOF for the NACA 0012 ($Re_c = 1 \times 10^6$) test case	170
D.3	Error convergence versus DOF for the NACA 0012 ($Re_c = 1 \times 10^7$) test case	170

Chapter 1

Introduction

In recent decades, Computational Fluid Dynamics (CFD) technology has achieved significant maturity. In particular, CFD is widely used throughout industry, academia, and government for analysis and design of aerospace vehicles. Despite this widespread use, many challenging problems remain. For example, when high accuracy simulations are necessary, computational costs using current industry standard techniques are very large. Furthermore, it is unclear if the grid generation practices and second-order finite volume methods typically used by the aerospace community are adequate given the stringent accuracy requirements of aerodynamic design.

To assess the current state of the art in CFD for applied aerodynamics, one can examine the results of the recent American Institute of Aeronautics and Astronautics (AIAA) Drag Prediction Workshops (DPW) [70, 65, 75]. These workshops, held in June 2001, June 2003, and June 2006, were convened with the explicit goals of assessing CFD as a practical tool for computation of aerodynamic forces for industry relevant geometries and identifying areas for additional research and development.

Over the three workshops, increasing effort has been directed to determining the effects of spatial discretization error. In the most recent workshop, four geometries were studied: two wing-body configurations and two wing-alone configurations [75, 101]. The wing-alone geometries were specifically included to minimize the complexity of the flow physics and enable grid convergence studies. The results show that the uncertainties associated with standard CFD methods are unacceptably high. In particular, the standard deviation of the submitted values of the total drag increases with the number of mesh points for both wing-alone geometries [75]. Also, the magnitude of the standard deviation, 5-7 drag counts, is substantially larger than the uncertainty desired by airframe designers [75, 100]. Furthermore, restricting comparison to only those submissions that used the SA turbulence model, the spread in the drag results extrapolated to continuum is more than 20 counts, or more

than ten percent of the total drag [101].

Additional evidence that discretization error plays a significant role in the errors in current CFD methods is presented by Mavriplis [73]. In particular, he demonstrates that, even for very large grids, asymptotic results appear to be different for different families of self-similar grids. This behavior has been attributed to the large range of scales present in high speed, turbulent aerodynamic flows. This large range of scales makes it difficult to adequately resolve all regions of the flow via global uniform refinement starting from an arbitrary mesh. Thus, even “fine” meshes may produce inaccurate results if important regions of the flow are not well resolved.

These results show that, not only is discretization error a significant contributor to the error in current CFD solutions, it can be very difficult to detect, even for expert practitioners. Thus, there exists a need in the CFD community for additional research and development aimed at detecting and reducing errors associated with spatial discretization. High-order, adaptive techniques have significant promise to accomplish this aim.

1.1 Objective

The objective of this work is to develop a high-order, adaptive method for the simulation of high Reynolds number, turbulent flows and to demonstrate the performance of the method for two-dimensional aerodynamic test cases.

1.2 Previous Work

1.2.1 High-Order Methods

High-order methods have significant potential to decrease the impact of discretization error on the accuracy of CFD solutions. Most CFD methods in widespread use in the aerospace industry achieve, at best, $E \propto h^2$, where E is a measure of the error in the solution and h is a measure of the mesh spacing. Thus, in the context of this work, high-order methods are those that achieve $E \propto h^r$, where $r > 2$, for sufficiently smooth problems. While second-order finite volume discretizations are popular, high-order methods have been extensively studied. These efforts have led to many types of high-order schemes, including finite difference, finite volume, and finite element methods.

In the context of finite difference methods, Lele [68] introduced up to tenth-order compact finite difference schemes. This work was extended and applied by Visbal and Gaitonde [104], who used high-order compact difference methods to solve the compressible Navier-Stokes equations on curvilinear meshes. Additional work in high-order finite

difference methods for aerodynamics was conducted by Zingg *et al.* [113], who showed that high-order compact difference methods are more efficient, in terms of number of nodes required to accurately compute drag, than typical second-order finite differences.

While these high-order finite difference techniques have been successfully used in many cases, their application is limited to structured meshes. To minimize mesh generation effort for complex geometries, unstructured meshes are of interest. For unstructured meshes, Barth [11] pioneered high-order finite volume methods using the least-squares k -exact reconstruction method. For this method, additional nodes and control volumes are added to each element to enable high-order reconstruction without extending the reconstruction support outside of the element. More recently, a similar idea, known as the spectral volume method, has been developed by Wang [105]. In this scheme, each mesh cell—i.e. spectral volume—is divided into sub-cells. The state averages on these sub-cells are then used to build a high-order reconstruction of the solution within the spectral volume.

In the context of finite element methods, researchers in the early 1980s pioneered the so-called p -type finite element method. In the p -type method, the grid spacing, h , remains fixed while the interpolation order, p , is increased to improve the accuracy of the solution. Babuska *et al.* [8] applied the p -type method to the elasticity equations in 1981 and concluded that the p -type method achieved superior performance in terms of the degrees of freedom required to achieve a desired accuracy. Later, Patera [81, 64] introduced a variant of the p -type method, known as the spectral element method, and used it to solve the incompressible Navier-Stokes equations.

While finite element methods offer a conceptually simple path to high-order accuracy, it is well known that the standard, continuous Galerkin method is inappropriate for use on convection-dominated problems. This drawback stems from the fact that the basic continuous Galerkin discretization is unstable for convection. Thus, even for subsonic flows, the discretization of the Euler or Navier-Stokes equations with finite element methods requires the addition of stabilization. One popular technique is the Streamline-Upwind Petrov Galerkin method [61]. Another method that has received significant attention is the discontinuous Galerkin (DG) method.

1.2.2 Discontinuous Galerkin Methods

For achieving high-order accuracy, DG is attractive for two reasons. First, it allows the development of stable, high-order accurate discretizations of convection-dominated problems using upwinding methods developed in the finite difference and finite volume communities. Second, the resulting discretizations have element-wise compact stencils. These compact

stencils simplify the task of achieving high-order accuracy on unstructured meshes and near boundaries, and they allow the development of efficient solution methods.

In 1973, Reed and Hill [90] introduced the DG method for the neutron transport equation. Since that time, there has been a rapid proliferation of new DG techniques as well as analyses and applications of those techniques. Cockburn *et al.* [26] provides an excellent review of work in DG methods through the year 2000. Highlights of this review as well as relevant recent advances are summarized here.

In 1974, LeSaint and Raviart [69] proved the first *a priori* error estimates of the DG method for linear hyperbolic problems. They showed that, assuming a smooth exact solution, the error measured in the L^2 -norm is $O(h^p)$. Later, Johnson and Pitkaranta [63] and Richter [92] improved upon the estimate of LeSaint and Raviart. Specifically, Johnson and Pitkaranta proved that, in the most general case, $O(h^{p+1/2})$ is the optimal convergence rate in L^2 , while Richter showed that, assuming the characteristic direction is not exactly aligned with the grid, $O(h^{p+1})$ can be obtained.

The extension from the method of Reed and Hill for linear problems to nonlinear hyperbolic problems is accomplished by the use of a Riemann solver to evaluate the flux across element boundaries. Riemann solvers have been extensively developed in the finite volume community [93, 97]. The first application of the DG method to a nonlinear hyperbolic problem was accomplished by Chavent and Salzano [24] using Godunov's flux.

A breakthrough in the application of DG methods to nonlinear hyperbolic problems using explicit time integration was made by Cockburn, Shu, and co-authors [28, 27, 25, 30], 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 [94]. This method was later generalized to be high-order accurate in time as well as space.

Independent of the above work, Allmaras [2] and Allmaras and Giles [4] developed a second-order DG scheme for the 2-D Euler equations. Their method is the extension of van Leer's method of moments [98] 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 [49] later extended this technique to be higher-order accurate.

For elliptic operators, in the late 1970s and early 1980s, Arnold [6] and Wheeler [107] introduced discontinuous finite element methods known as penalty methods. More recently, many researchers [14, 78, 29, 15, 16, 31, 20] have applied DG methods to diffusion problems. One procedure, pioneered by Bassi and Rebay [14, 15] and generalized by Cockburn and Shu [29, 31], is to rewrite a second-order equation as a first-order system and then discretize

the first-order system using the DG formulation. Methods derived in this fashion will be referred to as mixed formulations. The first mixed formulation developed by Bassi and Rebay (BR1) is not coercive and produces an extended stencil. However, a slight modification of BR1 leads to a coercive scheme with a nearest-neighbor stencil, referred to as the second method of Bassi and Rebay (BR2). The generalization of the first-order system idea by Cockburn and Shu leads to the so-called Local Discontinuous Galerkin methods (LDG). In multiple dimensions, LDG has an extended stencil, but a recent modification of LDG by Persson and Peraire [82] known as Compact Discontinuous Galerkin (CDG) recovers a compact stencil while retaining the attractive properties of LDG.

The penalty-type and mixed formulation DG methods have been brought into a single analysis framework. This framework, introduced by Arnold *et al.* [7], provides for a unified analysis, including error estimates, of these schemes. One recently introduced scheme which has not been incorporated into this framework is that of van Leer [99], who uses a patch reconstruction of the solution to evaluate the flux across element boundaries.

This work examines three DG discretizations of the RANS equations with the Spalart-Allmaras (SA) turbulence model. The application of DG discretizations to the RANS equations has been somewhat limited. At the time of this writing, the author is aware of three DG implementations of the RANS equations. Specifically, Bassi and Rebay [12] used the BR2 method to discretize the RANS equations coupled with the $k - \omega$ turbulence model, and Nguyen, Persson, and Peraire [76] used CDG for the RANS equations coupled with the SA turbulence model. Most recently, Landmann [66] has applied both LDG and BR2 to discretize the RANS equations coupled with the SA and $k - \omega$ turbulence models. All three of these methods are based on mixed formulations, which are examined in detail in Chapter 3.

1.2.3 Error Estimation and Adaptation

It is widely recognized that error estimation and adaptation increase the usefulness of CFD computations. Estimating the error in a CFD solution and generating appropriate high-quality meshes are difficult tasks, even for experts in CFD and aerodynamics. By enabling the user to set the error tolerance of the computation and automatically adapting the discretization to satisfy that tolerance, the confidence in the computed solution increases and the user is freed from the onerous task of generating a high-quality mesh.

Given the potential impact of error estimation and adaptation on the usefulness of CFD, many researchers have studied error estimation and adaptation algorithms. A brief summary of relevant advances is given here.

Error Estimation

This section gives a brief review of error estimation techniques used to drive mesh adaptation algorithms. Thus, the techniques discussed include both rigorous *a posteriori* error estimation algorithms as well as more ad hoc ideas used to construct “error indicators” to target mesh adaptation.

A common technique for driving adaptation in the CFD community is feature detection. This method aims to identify the dominant flow features, such as shock waves and boundary layers, by finding regions with large gradients [9, 86, 106]. The underlying assumption is that the error in these regions dominates the error in the solution. Thus, the identified large gradient regions are targeted for grid refinement. Such methods have been successfully applied to some flow problems, but they are clearly ad hoc. Furthermore, it has been demonstrated that simply refining the dominant features of the flow can lead to incorrect results [106]. For example, small errors in the upstream flow can affect shock or separation locations, leading to large errors in computed outputs. These errors are not reduced by continually refining the shock or boundary layer.

A less ad hoc procedure is offered by Zienkiewicz and Zhu [110, 111, 112], who propose an error estimation algorithm based on recovery. The idea underlying these techniques is to use the current discrete solution to reconstruct a better approximation of the exact solution. Then, the difference between the discrete solution and the reconstruction can be used to assess the local error in the solution. Many researchers have used similar schemes based on interpolation error estimates [21, 48, 83, 106]. However, such schemes have many drawbacks. For example, Ainsworth and Oden [1] present a second-order ODE case where the recovery-based error estimate is zero while the actual error can be arbitrarily large. Furthermore, given the local nature of such error estimates, they may fail to correctly capture propagation of error for convection-dominated problems. This failure can lead to similar problems as those described for purely feature-based algorithms.

Another type of error estimation technique relies on the residual. These estimates are motivated by the observation that it is often possible to bound the error by an appropriately defined residual norm. For example, consider a linear PDE of the form

$$\mathcal{L}u = f.$$

Then, given an approximate solution, u_h , the error, $e_h \equiv u - u_h$, is governed by

$$\mathcal{L}e_h = r_h$$

where $r_h \equiv f - \mathcal{L}u_h$. Furthermore, this provides an error bound of the form

$$\|e_h\|_1 \leq C\|r_h\|_2,$$

where $\|\cdot\|_1$ and $\|\cdot\|_2$ are appropriate norms, and C is a constant.

Certainly, the analysis is more difficult for nonlinear problems, but in many cases, it is possible to bound the error using an appropriate residual and norm [109, 87]. Thus, it is possible to construct error indicators based on the residual. An interesting study is provided by Zhang *et al.* [109]. They find that, for driving mesh adaptation in one-dimensional subsonic flow, both recovery-based and residual-based error estimation schemes are adequate, but that the residual-based method is more efficient for fine meshes. For one-dimensional transonic flow with shocks, they conclude that the residual-based method is superior because it is able to account for the transport of error due to convection. However, neither estimate is adequate for driving adaptation in two dimensions.

A significant drawback of both recovery- and residual-based procedures discussed thus far is that they aim to estimate some global solution error—e.g. the L^2 or H^1 norm of the error over the entire domain. Alternatively, the most important errors are generally those in quantities of engineering interest—e.g. lift, drag, etc. Hence, another class of methods has been developed to estimate the error in the output of interest directly. Such algorithms are known as output-based or goal-oriented error estimation schemes. These schemes are generally divided into two types: Type I methods, which depend explicitly on the solution of an appropriate dual problem, and Type II methods, where the dependence on the dual problem is eliminated. While both types of methods can be used to construct error indicators to target adaptation, Type I methods have been shown to be superior [55, 54] for efficient and accurate computation of functional outputs. Thus, Type I methods are examined here. In particular, the method used in this work is based on the Dual Weighted Residual (DWR) method due to Becker and Rannacher [17, 18]. The DWR method uses the property of Galerkin orthogonality of finite element discretizations, combined with duality concepts, to express the output error in terms of weighted residuals.

Many implementations, variations, and extensions of the DWR methods appear in the literature. For example, Pierce and Giles [85, 43, 45] and Venditti and Darmofal [102, 103] have developed duality based error correction methods that extend the superconvergence properties of finite element methods to more general discretizations. Venditti and Darmofal [103] also used an estimate of the error remaining after correction to drive a mesh adaptation procedure. This procedure was the first output-based, anisotropic adaptation method applied to the RANS equations. The results show that, for a second-order finite

volume discretization, the output-based approach is significantly more reliable than a purely feature-based adaptation scheme.

For DG finite element methods, Hartmann and Houston [55, 56, 57], Hartmann [52], Houston and Süli [59, 60], Lu [72], and Fidkowski [42, 41] have all used techniques based directly on or fundamentally similar to the DWR method. Similar ideas are also used in this work. They are presented in detail in Chapter 4.

Adaptation

Given an error indicator or estimate, the goal of the adaptation procedure is to modify the discretization to decrease the error. For finite element methods, this can be done in one of three ways: p -adaptation, where the order of the elements is modified on a constant mesh; h -adaptation, where the mesh is changed but the order of the elements remains fixed; or hp -adaptation, where both h and p are allowed to change. While p -adaptation is known to be more efficient for sufficiently regular solutions, h -adaptation easily allows for the generation of highly anisotropic meshes, which are critical for efficiency in high Re flows. While hp -adaptation might be the most efficient, in that case, one must contend with the additional difficulty of deciding between h and p adaptation. Some researchers have proposed algorithms for making this choice [60], but it is not a solved problem. Thus, for simplicity, h -adaptation alone is chosen for this work.

The h -adaptation is driven by an output-based error estimate. However, as noted earlier, appropriately anisotropic meshes are crucial for efficient simulation of high Re flows. In this work, the desired anisotropy calculation is based on equidistributing interpolation error. This approach is similar to that of Peraire *et al.* [83], who use the Hessian of the density to determine the mesh spacing request. Castro-Diaz *et al.* [21] and Habashi [48] also use the Hessian of a scalar quantity to determine the desired mesh spacing. Alternatively, Venditti [102, 103] uses an output error estimate to determine the absolute mesh size request and only uses the Hessian to determine the desired anisotropy. Fidkowski [42, 41] extended Venditti's approach to higher-order discretization methods. A similar technique based on higher-order derivatives has been applied by Leicht and Hartmann [67], who also consider an anisotropic adaptation indicator based on the inter-element jumps in the DG solution. The method of Fidkowski, described in Chapter 5, is used here.

1.3 Thesis Overview

This thesis describes the development of a high-order, h -adaptive, DG discretization for the RANS equations coupled with the SA turbulence model. In particular, the thesis makes

the following contributions:

- an analysis of the dual consistency of DG discretizations of source terms depending on state gradients, like those appearing in the SA model,
- extension of the output-based error estimation implementation to curved meshes,
- development of curved mesh generation techniques, including a global mapping approach and a linear elasticity mesh movement approach,
- application of the output-based error estimation and adaptation algorithms to high-order discretizations of the RANS equations on curved meshes, and
- development of an unsteady adaptation algorithm to improve the robustness of adaptation for steady state problems.

The thesis begins with a review of the RANS equations and the SA turbulence model in Chapter 2. Chapter 2 also describes modifications to the SA model made to improve robustness. Test results demonstrate that, as the mesh is refined, the modifications to the model have a diminishing effect on the computed solution. Finally, the chapter concludes by detailing the discretizations of the RANS-SA system used in this work.

An analysis of the dual consistency of these discretizations is given in Chapter 3. The analysis shows that, for source terms depending on the gradient of the state, the straightforward method of weighting the source term by a test function and integrating leads to a dual inconsistent scheme. A dual consistency correction to the standard weighting scheme is derived. Further analysis demonstrates that discretizations based on the popular mixed formulation are, in general, asymptotically dual consistent. Model problem and RANS results confirm the conclusions of the analysis.

Chapter 4 details the error estimation algorithm, including implementation modifications used here to improve the performance of the algorithm on curved meshes. Numerical results show that the error estimate is quite accurate for two high Re , two-dimensional test flows. Chapter 5 discusses the adaptation algorithm, including two curved mesh generation techniques. The first of these techniques relies on a global mapping from a straight meshing space to the curved physical domain. While this method is successfully demonstrated in two dimensions, it is quite restrictive. Thus, a more general technique, using a mesh movement algorithm based on a linear elasticity analogy, is developed. This technique is demonstrated on two-dimensional test problems, but it can be extended to three-dimensions in a straightforward manner. Chapter 5 concludes with multiple two-dimensional test cases demonstrating the performance of the adaptive algorithm.

Chapter 6 provides an unsteady adaptation algorithm for improving the robustness of the algorithm from Chapter 5. A major weakness of the standard algorithm is the need to obtain a steady state solution prior to adaptation. Especially when starting from coarse meshes, this need places an unacceptably high robustness requirement on the flow solver. Thus, the goal of the new adaptation approach is to allow mesh adaptation prior to obtaining a converged steady state solution. The new approach is referred to as the unsteady algorithm because the mesh is adapted at every time step while marching to the steady state solution. However, the goal is not a time accurate solution. Simple test cases demonstrate that the unsteady algorithm is capable of obtaining accurate steady state RANS solutions starting from very coarse, inviscid-style meshes.

The thesis ends with Chapter 7, which gives conclusions and suggestions for future work.

Chapter 2

Discretization of the RANS Equations

This chapter describes the RANS equations, the SA turbulence model, and the discretization of the RANS-SA system. It begins with a brief review of the RANS equations and the SA turbulence model in Sections 2.1 and 2.2. Sections 2.3 and 2.4 show the spatial and temporal discretizations used in this work. An in-depth analysis of the spatial discretizations is given in Chapter 3. Finally, Section 2.5 provides a brief overview of the solution method used to solve the discrete system.

2.1 The RANS Equations

The RANS equations are derived by averaging the Navier-Stokes equations. Specifically, for compressible flows, the Favre averaging procedure is used. This procedure as well as simplifying assumptions used here are described in Appendix A. The form of the RANS equations used in this work is given by

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial}{\partial x_i}(\bar{\rho} \tilde{u}_i) = 0, \quad (2.1)$$

$$\frac{\partial}{\partial t}(\bar{\rho} \tilde{u}_i) + \frac{\partial}{\partial x_j}(\bar{\rho} \tilde{u}_j \tilde{u}_i + \bar{p} \delta_{ji}) = \frac{\partial}{\partial x_j} \left[2(\mu + \mu_t) \left(\tilde{s}_{ji} - \frac{1}{3} \frac{\partial \tilde{u}_k}{\partial x_k} \delta_{ji} \right) \right], \quad (2.2)$$

$$\begin{aligned} & \frac{\partial}{\partial t} \left[\bar{\rho} \left(\tilde{e} + \frac{1}{2} \tilde{u}_i \tilde{u}_i \right) \right] + \frac{\partial}{\partial x_j} \left[\bar{\rho} \tilde{u}_j \left(\tilde{h} + \frac{1}{2} \tilde{u}_i \tilde{u}_i \right) \right] \\ & = \frac{\partial}{\partial x_j} \left[c_p \left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t} \right) \frac{\partial \tilde{T}}{\partial x_j} \right] + \frac{\partial}{\partial x_j} \left[\tilde{u}_i 2(\mu + \mu_t) \left(\tilde{s}_{ij} - \frac{1}{3} \frac{\partial \tilde{u}_k}{\partial x_k} \delta_{ij} \right) \right], \quad (2.3) \end{aligned}$$

where ρ denotes the density, u_i are the velocity components, p is the pressure, e is internal energy, h is the enthalpy, T is the temperature, $s_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$ is the strain-rate tensor, μ is the dynamic viscosity, μ_t is the dynamic eddy viscosity, Pr is the Prandtl number, and Pr_t is the turbulent Prandtl number. The $(\bar{\cdot})$ and $(\tilde{\cdot})$ notation indicates Reynolds-averaging and Favre-averaging, respectively. These averages are defined in Appendix A.

Equations (2.1) through (2.3) contain more unknowns than equations. In particular, the eddy viscosity, μ_t , which relates the mean flow viscous stresses to the apparent stresses due to turbulent fluctuations, cannot yet be computed. Thus, to close the system, (2.1) through (2.3) are augmented by the SA turbulence model, described in Section 2.2.

To avoid confusion, the $(\bar{\cdot})$ and $(\tilde{\cdot})$ notation is dropped for the remainder of the thesis. All uses of the standard flow variables refer to the appropriate mean flow quantities—e.g. ρ is the Reynolds-average density and u_i is the Favre-average velocity.

2.2 The SA Turbulence Model

The closure of the RANS system is accomplished by the addition of a turbulence model. In this work, the SA turbulence model [95] is used. The model is widely used in the aerospace industry and is generally regarded as robust. Moreover, it has been shown to be accurate for most attached and mildly separated aerodynamic flows [46, 89, 108, 23].

2.2.1 Baseline Model

The model takes the form of a PDE for a working variable, $\tilde{\nu}$, which is algebraically related to the eddy viscosity, μ_t . In particular, the eddy viscosity is given by

$$\mu_t = \rho \tilde{\nu} f_{v1},$$

where

$$f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad \chi = \frac{\tilde{\nu}}{\nu},$$

and $\nu = \mu/\rho$ is the kinematic viscosity. Then, $\rho \tilde{\nu}$ is governed by

$$\begin{aligned} \frac{\partial}{\partial t}(\rho \tilde{\nu}) + \frac{\partial}{\partial x_j}(\rho u_j \tilde{\nu}) &= c_{b1} \tilde{S} \rho \tilde{\nu} \\ + \frac{1}{\sigma} \left[\frac{\partial}{\partial x_j} \left((\mu + \rho \tilde{\nu}) \frac{\partial \tilde{\nu}}{\partial x_j} \right) + c_{b2} \rho \frac{\partial \tilde{\nu}}{\partial x_j} \frac{\partial \tilde{\nu}}{\partial x_j} \right] &- c_{w1} f_w \frac{\rho \tilde{\nu}^2}{d^2}, \end{aligned} \quad (2.4)$$

where

$$\tilde{S} = \begin{cases} S + \bar{S}, & \bar{S} \geq -c_{v2}S \\ S + \frac{S(c_{v2}^2S + c_{v3}\bar{S})}{(c_{v3} - 2c_{v2})S - \bar{S}}, & \bar{S} < -c_{v2}S, \end{cases}$$

S is the magnitude of the vorticity, and

$$\bar{S} = \frac{\tilde{\nu}f_{v2}}{\kappa^2d^2}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}.$$

The remaining closure functions are

$$f_w = g \left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6} \right)^{1/6}, \quad g = r + c_{w2}(r^6 - r), \quad r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2d^2},$$

where d is the distance to the nearest wall, $c_{b1} = 0.1355$, $\sigma = 2/3$, $c_{b2} = 0.622$, $\kappa = 0.41$, $c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma$, $c_{w2} = 0.3$, $c_{w3} = 2$, $c_{v1} = 7.1$, $c_{v2} = 0.7$, $c_{v3} = 0.9$, and $Pr_t = 0.9$.

Some clarifying remarks are in order. To begin, the laminar suppression and trip terms from the original model are omitted because they are not used in this work. All cases are run fully turbulent with no effort to model transition or force the flow to transition at a desired location.

In addition, the form of the SA model shown in (2.4) has been modified in two ways from that given in [95]. First, it is a straightforward generalization of the original model to the compressible flow case. For incompressible flows, the two models are exactly the same. No effort was made in this work to optimize the form of the model for flows where compressibility effects are highly important. Thus, it is likely that another form would be more appropriate for such cases. For example, Catris and Aupoix [22] propose a form of the model that is compatible with density variations in the log layer of a compressible boundary layer.

Second, the form of the production term has been changed. In the original production term, \tilde{S} is given by simply $\tilde{S} = S + \bar{S}$. This form can cause robustness problems because the production can be negative, even for positive values of $\tilde{\nu}$. Negative values can occur because the f_{v2} closure function is negative for approximately $1.00 \leq \chi \leq 18.40$, as shown in Figure 2-1.

The new form of the production was developed by Johnson and Allmaras to avoid the possibility of negative \tilde{S} for positive $\tilde{\nu}$ [3]. In fact, \tilde{S} is non-negative for all $\tilde{\nu}$, and as $\bar{S} \rightarrow -\infty$, $\tilde{S} \rightarrow (1 - c_{v3})S$. Furthermore, the function is C^1 continuous, with the value and derivative matching at $\bar{S} = -c_{v2}S$, where $\tilde{S} = (1 - c_{v2})S$.

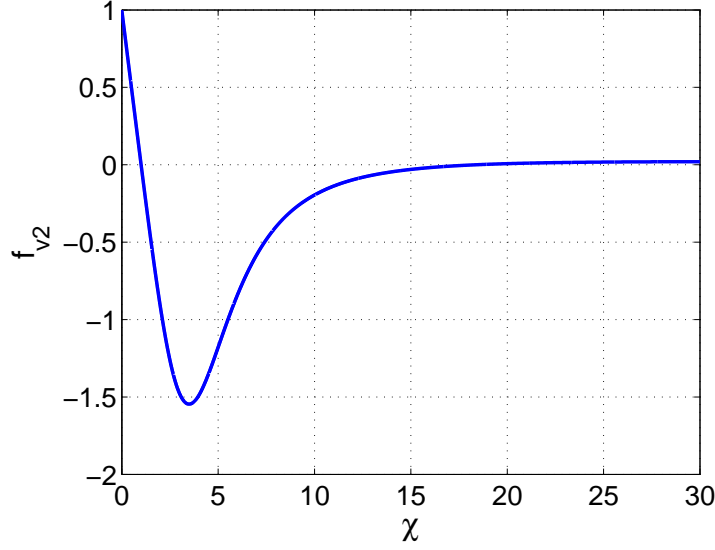


Figure 2-1: SA model f_{v2} closure function versus χ

2.2.2 Negative $\tilde{\nu}$ Modifications

The exact solution of (2.4) is non-negative, which agrees with the physical intuition that $\mu_t \geq 0$. However, the discrete solution of (2.4) may not share this property. More importantly, negative $\tilde{\nu}$ values can adversely impact the iterative convergence of the discrete solution, even causing it to diverge in some cases.

To ameliorate this behavior, changes can be made to the model for negative $\tilde{\nu}$ values. The most obvious change is to the definition of the eddy viscosity. The modified definition is given by

$$\mu_t = \begin{cases} \rho \tilde{\nu} f_{v1} & \tilde{\nu} > 0 \\ 0 & \tilde{\nu} \leq 0. \end{cases} \quad (2.5)$$

Clearly, this modification ensures that the eddy viscosity is non-negative. Furthermore, the definition is continuous and has continuous first and second derivatives.

The remaining changes are motivated by examining the energy of the turbulence model working variable, $e_{\tilde{\nu}} \equiv \frac{1}{2}\tilde{\nu}^2$. One can derive a governing equation for $e_{\tilde{\nu}}$ by multiplying (2.4) by $\tilde{\nu}$. The resulting equation is

$$\frac{\partial}{\partial t}(\rho e_{\tilde{\nu}}) + \frac{\partial}{\partial x_j}(\rho u_j e_{\tilde{\nu}}) = \frac{1}{\sigma} \left[\frac{\partial}{\partial x_j} \left(\eta \frac{\partial e_{\tilde{\nu}}}{\partial x_j} \right) + (c_{b2}\rho\tilde{\nu} - \eta) \frac{\partial \tilde{\nu}}{\partial x_j} \frac{\partial \tilde{\nu}}{\partial x_j} \right] + \tilde{\nu}(P - D), \quad (2.6)$$

where $\eta = \mu + \rho\tilde{\nu}$, $P = c_{b1}\tilde{S}\rho\tilde{\nu}$, $D = c_{w1}f_w\rho\tilde{\nu}^2/d^2$.

For $\tilde{\nu} < 0$, the right-hand side of (2.6) should act to dissipate $e_{\tilde{\nu}}$. This property ensures

that, in regions where $\tilde{\nu} < 0$, the energy of the turbulence model working variable will decrease with time.

To make this notion more precise, define the integrated energy by

$$E_{\tilde{\nu}}(t) = \int_{\Omega} \rho e_{\tilde{\nu}}(\mathbf{x}, t) d\mathbf{x},$$

where the bounded set $\Omega \subset \mathbb{R}^n$ is the domain of interest. Furthermore, define the following sub-domains:

$$\begin{aligned} \Omega_t^+ &= \{\mathbf{x} \in \Omega \mid \tilde{\nu}(\mathbf{x}, t) > 0\}, \\ \Omega_t^- &= \{\mathbf{x} \in \Omega \mid \tilde{\nu}(\mathbf{x}, t) < 0\}. \end{aligned}$$

Then, $E_{\tilde{\nu}} = E_{\tilde{\nu}}^+ + E_{\tilde{\nu}}^-$, where

$$\begin{aligned} E_{\tilde{\nu}}^+(t) &= \int_{\Omega_t^+} \rho e_{\tilde{\nu}}(\mathbf{x}, t) d\mathbf{x}, \\ E_{\tilde{\nu}}^-(t) &= \int_{\Omega_t^-} \rho e_{\tilde{\nu}}(\mathbf{x}, t) d\mathbf{x}. \end{aligned}$$

To bound the solution in regions where $\tilde{\nu} < 0$, consider the integrated energy contained in Ω_t^- , $E_{\tilde{\nu}}^-$. Given an initial value at time t_0 , one can determine if this energy will grow by examining the derivative $\frac{dE_{\tilde{\nu}}^-}{dt}$. In particular,

$$\begin{aligned} \frac{dE_{\tilde{\nu}}^-}{dt} &= \frac{d}{dt} \int_{\Omega_t^-} \rho e_{\tilde{\nu}}(\mathbf{x}, t) d\mathbf{x}, \\ &= \int_{\partial\Omega_t^-} \rho e_{\tilde{\nu}} \vec{v} \cdot \vec{n} ds + \int_{\Omega_t^-} \frac{\partial}{\partial t} (\rho e_{\tilde{\nu}}) d\mathbf{x}, \end{aligned}$$

where \vec{v} denotes the velocity of the movement of the boundary of Ω_t^- and \vec{n} is the outward pointing unit normal vector.

Consider the case where $\partial\Omega_t^- \cap \partial\Omega = \emptyset$. Then, assuming that $\tilde{\nu}$ is continuous, it is clear that $\tilde{\nu}|_{\partial\Omega_t^-} = 0$, which implies that $e_{\tilde{\nu}}|_{\partial\Omega_t^-} = 0$. Thus,

$$\int_{\partial\Omega_t^-} \rho e_{\tilde{\nu}} \vec{v} \cdot \vec{n} ds = 0.$$

Furthermore, using (2.6),

$$\begin{aligned} \int_{\Omega_t^-} \frac{\partial}{\partial t} (\rho e_{\tilde{\nu}}) d\mathbf{x} &= \int_{\Omega_t^-} \left[-\frac{\partial}{\partial x_j} (\rho u_j e_{\tilde{\nu}}) + \frac{1}{\sigma} \frac{\partial}{\partial x_j} \left(\eta \frac{\partial e_{\tilde{\nu}}}{\partial x_j} \right) \right] \\ &\quad + \int_{\Omega_t^-} \left[\frac{(c_{b2} \rho \tilde{\nu} - \eta)}{\sigma} \frac{\partial \tilde{\nu}}{\partial x_j} \frac{\partial \tilde{\nu}}{\partial x_j} + \tilde{\nu} (P - D) \right] \end{aligned}$$

Applying the divergence theorem and recalling that $\tilde{\nu}|_{\partial\Omega_t^-} = 0$ gives

$$\frac{dE_{\tilde{\nu}}^-}{dt} = \int_{\Omega_t^-} \left[\frac{(c_{b2} \rho \tilde{\nu} - \eta)}{\sigma} \frac{\partial \tilde{\nu}}{\partial x_j} \frac{\partial \tilde{\nu}}{\partial x_j} + \tilde{\nu} (P - D) \right]. \quad (2.7)$$

If the right-hand side of (2.7) is negative or zero, then $E_{\tilde{\nu}}^-$ is a non-increasing function of time. Thus, $E_{\tilde{\nu}}^-$ is bounded by its initial value, which provides a bound on the size of the solution in regions where $\tilde{\nu} < 0$.

However, for the baseline model, $\frac{dE_{\tilde{\nu}}^-}{dt}$ can be positive. The contribution of the term $\frac{1}{\sigma} (c_{b2} \rho \tilde{\nu} - \eta) \frac{\partial \tilde{\nu}}{\partial x_j} \frac{\partial \tilde{\nu}}{\partial x_j}$ is positive whenever $(c_{b2} \rho \tilde{\nu} - \eta)$ is positive, which occurs for $\chi < 1/(c_{b2} - 1) \approx -2.65$. Furthermore, the contribution of the production term, given by

$$\tilde{\nu} P = c_{b1} \tilde{S} \rho \tilde{\nu}^2,$$

is non-negative regardless of $\tilde{\nu}$. Finally, the contribution of the destruction term is

$$-\tilde{\nu} D = -c_{w1} f_w \frac{\rho \tilde{\nu}^3}{d^2}.$$

The sign of this term is not entirely determined by $\tilde{\nu}$ because the function f_w changes sign at $r \approx -1.18$, and r depends on $\tilde{\nu}$, \tilde{S} , and d . However, for $r < -1.18$, f_w is positive, which, for $\tilde{\nu} < 0$, implies that $-\tilde{\nu} D$ is positive.

To fix these problems, one can change the model when $\tilde{\nu} < 0$ to ensure that

$$(c_{b2} \rho \tilde{\nu} - \eta) < 0, \quad (2.8)$$

$$\tilde{\nu} (P - D) < 0. \quad (2.9)$$

To satisfy these properties, changes to the diffusion coefficient, η , as well as the production, P , and destruction, D , terms have been made. These changes were suggested by Allmaras [3].

To satisfy (2.8) the diffusion coefficient is re-defined as

$$\eta = \begin{cases} \mu(1 + \chi), & \chi \geq 0 \\ \mu(1 + \chi + \frac{1}{2}\chi^2), & \chi < 0. \end{cases} \quad (2.10)$$

The diffusion coefficient and the quantity $c_{b2}\chi - \eta/\mu$ are plotted in Figure 2-2. Clearly, the modification ensures that the diffusion coefficient is always positive and $c_{b2}\chi - \eta/\mu$ is always negative. Also, η is continuous and has continuous first derivatives.

The modified production is given by

$$P = \begin{cases} c_{b1}\tilde{S}\rho\tilde{\nu}, & \chi \geq 0 \\ c_{b1}S\rho\tilde{\nu}g_n, & \chi < 0, \end{cases} \quad (2.11)$$

where

$$g_n = 1 - \frac{1000\chi^2}{1 + \chi^2}.$$

The function g_n is chosen such that P has continuous first derivatives at $\tilde{\nu} = 0$. This choice implies that $g_n > 0$, and thus $\tilde{\nu}P > 0$, for some negative $\tilde{\nu}$. However, $g_n < 0$ for $\chi < -\sqrt{1/999} \approx -0.032$, which implies that $\tilde{\nu}P < 0$ for $\chi < -0.032$.

Finally, the modified destruction is given by

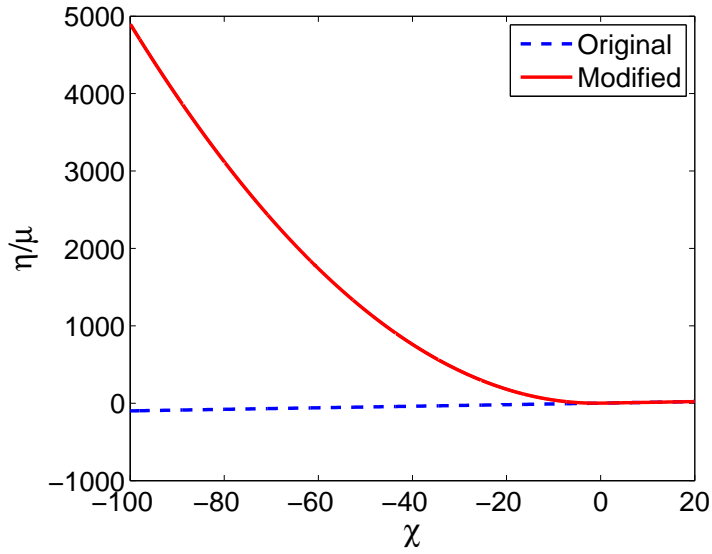
$$D = \begin{cases} c_{w1}f_w\frac{\rho\tilde{\nu}^2}{d^2}, & \chi \geq 0 \\ -c_{w1}\frac{\rho\tilde{\nu}^2}{d^2}, & \chi < 0. \end{cases} \quad (2.12)$$

Clearly, D is negative for $\tilde{\nu} < 0$. Thus, $-\tilde{\nu}D < 0$ for $\tilde{\nu} < 0$. Furthermore, D is continuous and has continuous first derivatives.

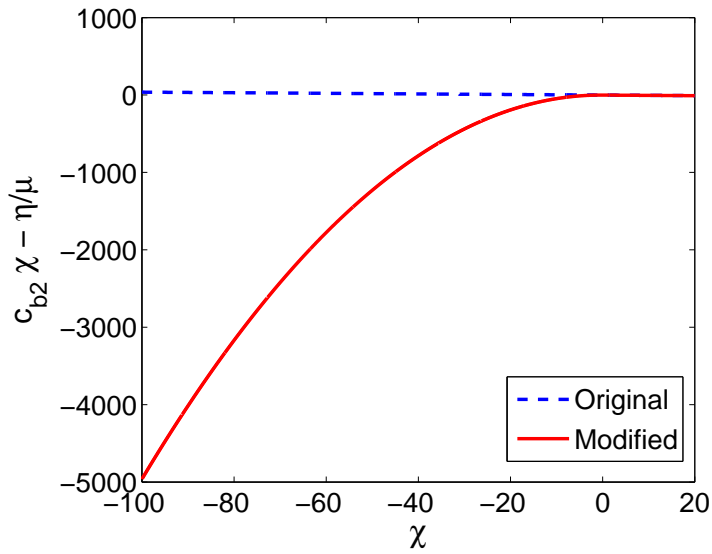
2.2.3 Comparison of SA Model Versions

To demonstrate the effects of the modifications to the SA turbulence model on computed flow solutions, an example case is solved using three versions of the model: the original model as presented in [95], the baseline model presented in Section 2.2.1, and the modified model presented in Section 2.2.2. For all the models, the non-negative form of μ_t , given in (2.5), is used, as this modification was required to obtain converged solutions for the original and baseline models.

The case is $M_\infty = 0.25$, $Re_c = 1 \times 10^7$ flow over a flat plate with zero pressure gradient. This test case will be examined further in Chapters 3, 4, and 5. Here, the only interest is the dependence of the solution on the modifications to the turbulence model. For each model, the RANS-SA system is discretized using the standard weighting scheme, which is



(a) Original and modified η/μ versus χ



(b) Original and modified $c_{b2}\chi - \eta/\mu$ versus χ

Figure 2-2: Comparison of baseline and modified SA model diffusion terms

described in Section 2.3. All results shown were computed using $p = 3$ polynomials.

Figures 2-3 and 2-4 show the turbulence model working variable profiles computed on coarse (234 elements) and fine (866 elements) meshes, respectively. Two stations are shown: $x/c = 0.5$ and $x/c = 1.0$. The profiles are very similar except in the boundary layer edge

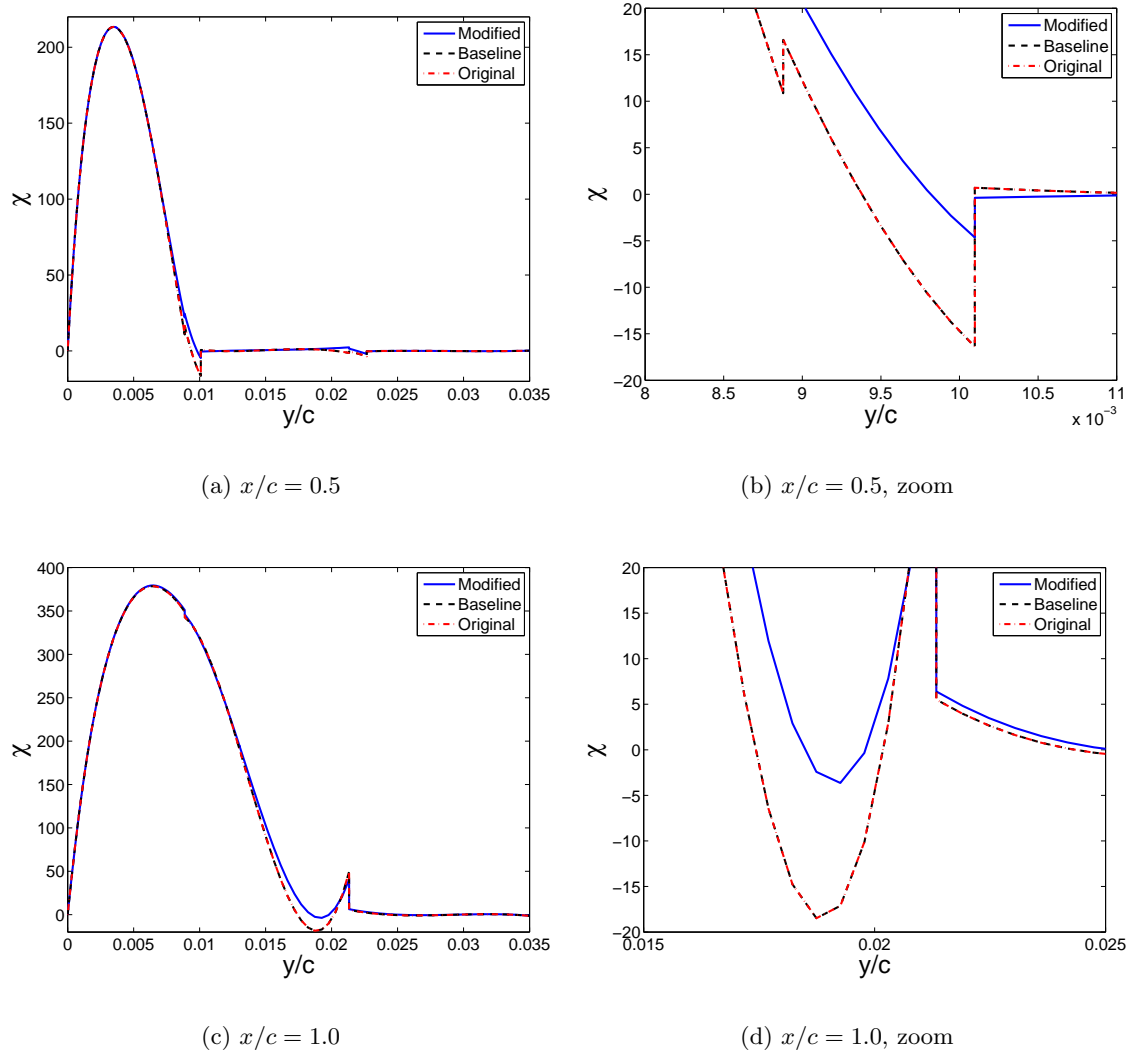
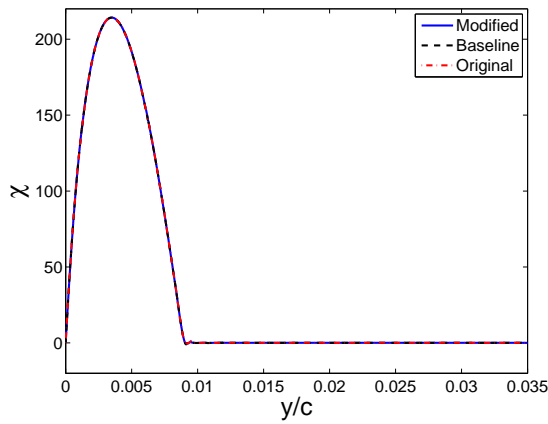
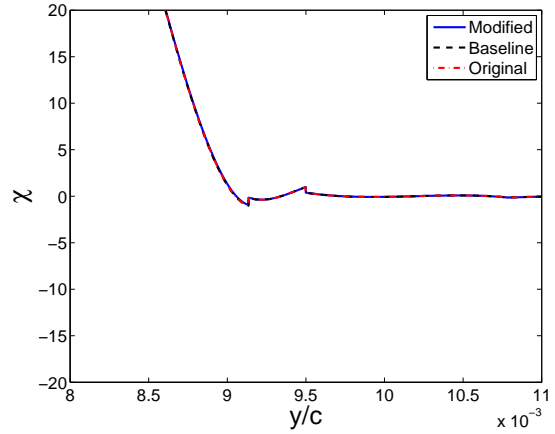


Figure 2-3: Turbulence model working variable profiles for flow over a flat plate, computed using three versions of the SA model on the coarse mesh

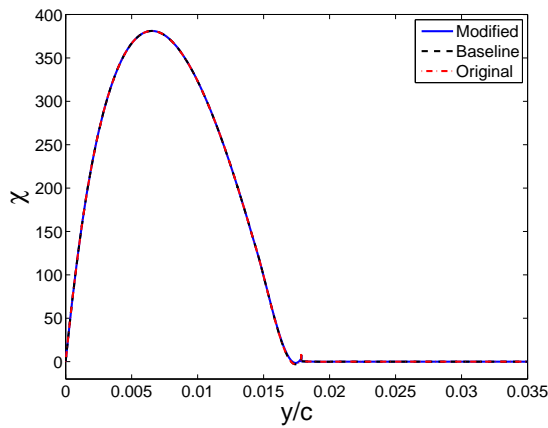
region. In this region, the original and baseline models are still very similar. However, while all three models predict $\tilde{\nu} < 0$, the modified model gives negative $\tilde{\nu}$ values with smaller magnitude. Furthermore, the differences between the solutions from the two models are smaller on the fine mesh. Thus, as the mesh is refined, the solutions from the models appear to converge.



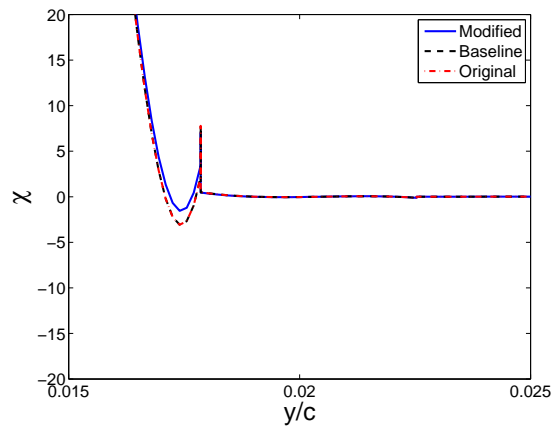
(a) $x/c = 0.5$



(b) $x/c = 0.5$, zoom



(c) $x/c = 1.0$



(d) $x/c = 1.0$, zoom

Figure 2-4: Turbulence model working variable profiles for flow over a flat plate, computed using three versions of the SA model on the fine mesh

More importantly, the differences in the $\tilde{\nu}$ solution have very little impact on the other flow variables. Figure 2-5 shows the velocity profile at $x/c = 0.5$, computed on the coarse mesh. The three models produce virtually the same profile. Figure 2-6 shows the skin fric-

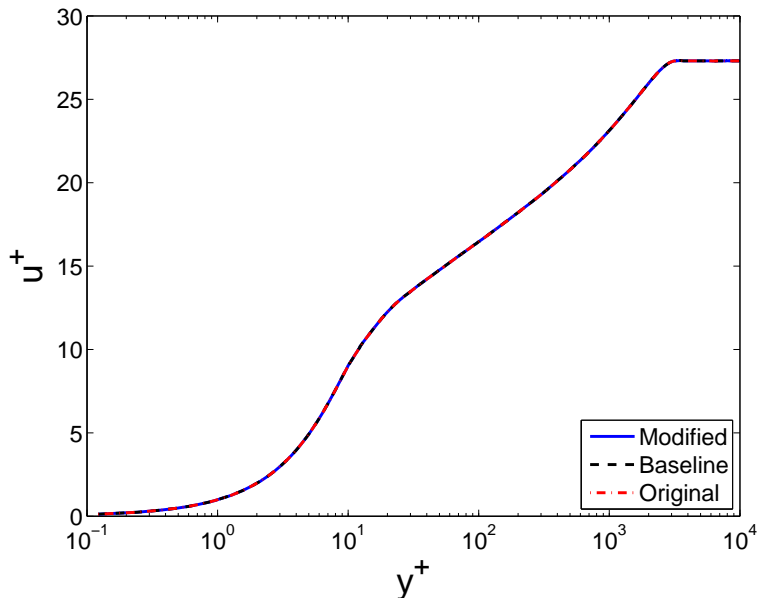


Figure 2-5: Velocity profiles for flow over a flat plate at $Re_x = 5 \times 10^6$, computed using three versions of the SA model on the coarse mesh

tion distribution for all three models, also computed on the coarse mesh. The distributions are practically indistinguishable. Furthermore, the drag computed using the three models differs by only 0.02 percent. On the fine mesh, the spread of the drag is less than 5×10^{-5} percent.

Thus, as expected, the modifications to the turbulence model have little effect on the converged solution. However, as desired, the negative $\tilde{\nu}$ modifications tend to increase $\tilde{\nu}$ in regions where $\tilde{\nu} < 0$, leading to a more robust scheme.

2.3 Spatial Discretization

To simplify the notation for the discretization, rewrite the RANS equations as

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathcal{F}(\mathbf{u}) - \nabla \cdot (\mathcal{A}(\mathbf{u})\nabla \mathbf{u}) = \mathbf{S}(\mathbf{u}, \nabla \mathbf{u}), \quad (2.13)$$

where $\mathbf{u} = [\rho, \rho u_i, \rho E, \rho \tilde{\nu}]^T$ is the conservative state vector, \mathcal{F} is the inviscid flux, $\mathcal{A}\nabla \mathbf{u}$ is the viscous flux, and \mathbf{S} is the source term.

Let T_h be a triangulation of the domain $\Omega \subset \mathbb{R}^n$ into non-overlapping elements κ . Define

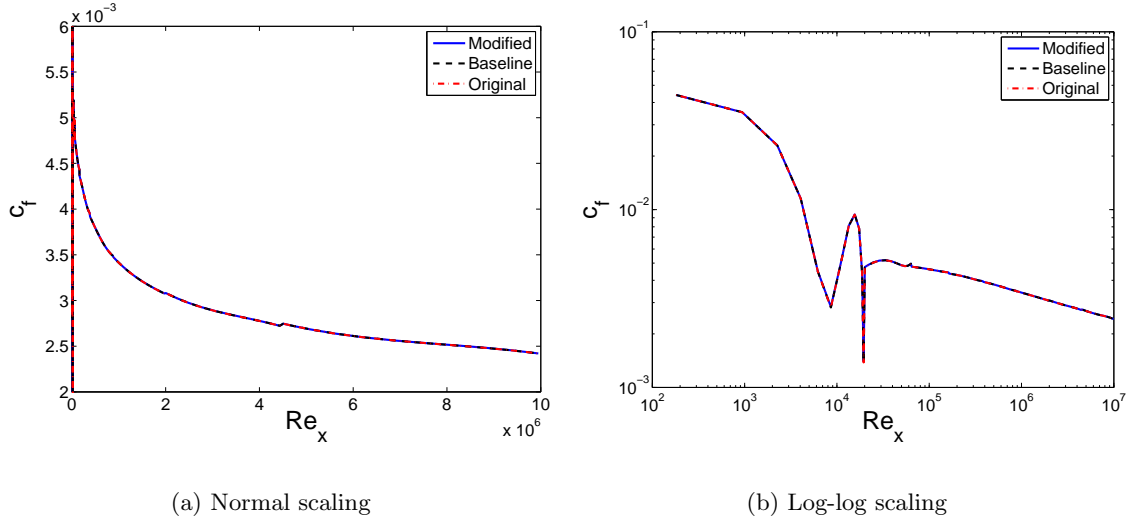


Figure 2-6: Skin friction distributions for flow over a flat plate, computed using three versions of the SA model on the coarse mesh

the function space \mathcal{V}_h^p by

$$\mathcal{V}_h^p \equiv \{\mathbf{v} \in [L^2(\Omega)]^r \mid \mathbf{v} \circ f_\kappa \in [P^p(\kappa_{ref})]^r, \forall \kappa \in T_h\},$$

where r is the length of the state vector, P^p denotes the space of polynomials of order p , and f_κ denotes the mapping from the reference element to physical space for the element κ , as illustrated in two dimensions by Figure 2-7.

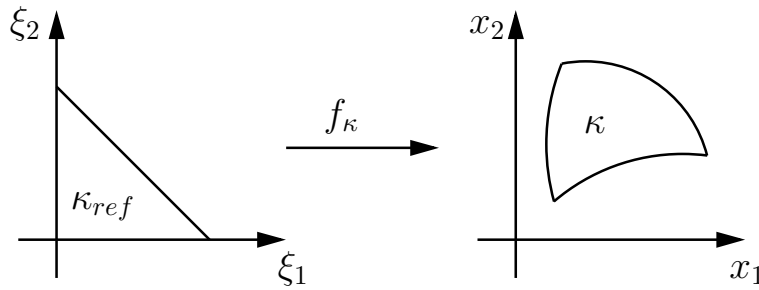


Figure 2-7: Illustration of the map f_κ from the reference element to the element κ in physical space

Then, the spatially discrete problem is as follows: find $\mathbf{u}_h(\cdot, t) \in \mathcal{V}_h^p$ such that

$$\sum_{\kappa \in T_h} \int_{\kappa} \mathbf{v}_h^T \frac{\partial \mathbf{u}_h}{\partial t} + R_h(\mathbf{u}_h, \mathbf{v}_h) = 0, \quad \forall \mathbf{v}_h \in \mathcal{V}_h^p, \quad (2.14)$$

where

$$R_h(\mathbf{w}_h, \mathbf{v}_h) = R_{h,I}(\mathbf{w}_h, \mathbf{v}_h) + R_{h,V}(\mathbf{w}_h, \mathbf{v}_h) + R_{h,S}(\mathbf{w}_h, \mathbf{v}_h),$$

and $R_{h,I}$, $R_{h,V}$, and $R_{h,S}$ denote the discretizations of the inviscid, viscous, and source terms, respectively.

2.3.1 Inviscid Discretization

The discretization of the inviscid terms is given by

$$\begin{aligned} R_{h,I}(\mathbf{w}_h, \mathbf{v}_h) &\equiv - \sum_{\kappa \in T_h} \int_{\kappa} \nabla \mathbf{v}_h^T \cdot \mathcal{F}(\mathbf{w}_h) \\ &\quad + \int_{\Gamma_i} (\mathbf{v}_h^+ - \mathbf{v}_h^-)^T \mathbf{H}(\mathbf{w}_h^+, \mathbf{w}_h^-, \vec{n}^+) + \int_{\partial\Omega} \mathbf{v}_h^T \mathcal{F}^b \cdot \vec{n}, \end{aligned}$$

where $(\cdot)^+$ and $(\cdot)^-$ denote trace values taken from opposite sides of a face, \vec{n}^+ is the normal vector pointing from $+$ to $-$, and \mathbf{H} is a numerical flux function for interior faces. In this work, \mathbf{H} is the Roe flux [93]. The inviscid boundary flux, \mathcal{F}^b , is computed by evaluating the flux at a boundary state, \mathbf{u}^b , which, in general, depends on both the interior state and the boundary conditions. The boundary condition specification is discussed in more detail in Appendix E.

2.3.2 Viscous Discretization

The viscous terms are discretized using the second method of Bassi and Rebay [15, 16]. To write the discretization in a compact form, the jump, $[[\cdot]]$, and average, $\{\cdot\}$, operators are used. For interior faces, let

$$\begin{aligned} \{s\} &= \frac{1}{2}(s^+ + s^-), & \{\vec{v}\} &= \frac{1}{2}(\vec{v}^+ + \vec{v}^-), \\ [[s]] &= (s^+ \vec{n}^+ + s^- \vec{n}^-), & [[\vec{v}]] &= (\vec{v}^+ \cdot \vec{n}^+ + \vec{v}^- \cdot \vec{n}^-), \end{aligned}$$

where s is a scalar and \vec{v} is a vector. Then, the viscous discretization is given by

$$\begin{aligned} R_{h,V}(\mathbf{w}_h, \mathbf{v}_h) &\equiv \sum_{\kappa \in T_h} \int_{\kappa} \nabla \mathbf{v}_h^T \cdot (\mathcal{A}(\mathbf{w}_h) \nabla \mathbf{w}_h) \\ &\quad - \int_{\Gamma_i} [[\mathbf{w}_h]]^T \cdot \{\mathcal{A}^T(\mathbf{w}_h) \nabla \mathbf{v}_h\} - [[\mathbf{v}_h]]^T \cdot (\{\mathcal{A}(\mathbf{w}_h) \nabla \mathbf{w}_h\} + \eta_f \{\vec{\mathbf{r}}_f(\mathbf{w}_h)\}) \\ &\quad - \int_{\partial\Omega} [(\mathbf{w}_h^+ - \mathbf{u}^b)^T (\mathcal{A}^T(\mathbf{u}^b) \nabla \mathbf{v}_h^+) \cdot \vec{n}^+ + \mathbf{v}_h^T (\mathcal{F}_v^b - \eta_f \vec{\mathbf{r}}_f^b(\mathbf{w}_h)) \cdot \vec{n}], \end{aligned}$$

where \mathcal{F}_v^b denotes the viscous boundary flux function, $\vec{\mathbf{r}}_f$ and $\vec{\mathbf{r}}_f^b$ are auxiliary variables, and η_f is a stabilization parameter. The stabilization parameter is set to $\eta_f = 6$ for all cases shown in this work. Given that $\eta_f > 3$ implies stability of the BR2 discretization for triangular meshes ($\eta_f > 4$ for quadrilateral meshes), this value is somewhat conservative. The auxiliary variables are defined by the following auxiliary problems: for each face, σ_f , find $\vec{\mathbf{r}}_f \in [\mathcal{V}_h^p]^n$ such that

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h^T \cdot \vec{\mathbf{r}}_f(\mathbf{w}_h) = \int_{\sigma_f} \llbracket \mathbf{w}_h \rrbracket^T \cdot \{\mathcal{A}^T(\mathbf{w}_h) \vec{\tau}_h\}, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n,$$

for interior faces, and find $\vec{\mathbf{r}}_f^b \in [\mathcal{V}_h^p]^n$

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h^T \cdot \vec{\mathbf{r}}_f^b(\mathbf{w}_h) = \int_{\sigma_f} (\mathbf{w}_h^+ - \mathbf{u}^b)^T (\mathcal{A}^T(\mathbf{u}^b) \vec{\tau}_h^+) \cdot \vec{n}^+, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n,$$

for boundary faces.

2.3.3 Source Discretization

Three options for the discretization of the source terms are considered. The first option is referred to as the standard weighting approach. In this approach, the source term is simply multiplied by the test function and integrated over the domain. Thus,

$$R_{h,S,SW}(\mathbf{w}_h, \mathbf{v}_h) \equiv - \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{v}_h^T \mathbf{S}(\mathbf{w}_h, \nabla \mathbf{w}_h). \quad (2.15)$$

The second approach is a modification of the standard weighting approach that ensures the source term discretization is dual consistent. The dual consistent discretization is given by

$$\begin{aligned} R_{h,S,DC}(\mathbf{w}_h, \mathbf{v}_h) &\equiv - \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{v}_h^T \mathbf{S}(\mathbf{w}_h, \nabla \mathbf{w}_h) \\ &+ \int_{\Gamma_i} \llbracket \mathbf{w}_h \rrbracket^T \cdot \{\vec{\beta}(\mathbf{w}_h, \nabla \mathbf{w}_h, \mathbf{v}_h)\} + \int_{\partial\Omega} (\mathbf{u}^b - \mathbf{w}_h^+) \vec{\beta}_b(\mathbf{w}_h, \nabla \mathbf{w}_h, \mathbf{v}_h) \cdot \vec{n}. \end{aligned}$$

The requirements on the “dual fluxes” $\vec{\beta}$ and $\vec{\beta}_b$ will be specified in Chapter 3, where a full derivation and analysis of this discretization are given.

The third approach is based on mixed formulations—e.g. BR2 [15, 16] and LDG [29, 31]—where an auxiliary state variable is solved for the gradient of the state. Specifically,

the discretization is given by

$$R_{h,S,MF}(\mathbf{w}_h, \mathbf{v}_h) \equiv - \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{v}_h^T \mathbf{S}(\mathbf{w}_h, \bar{\mathbf{q}}_h). \quad (2.16)$$

where $\bar{\mathbf{q}}_h \in [\mathcal{V}_h^p]^n$ satisfies

$$\begin{aligned} \sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \bar{\mathbf{q}}_h &= - \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{w}_h \nabla \cdot \vec{\tau}_h + \int_{\Gamma_i} ([\hat{\mathbf{u}}] \cdot \{\vec{\tau}_h\} + \{\hat{\mathbf{u}}\} [[\vec{\tau}_h]]) \\ &\quad + \int_{\partial\Omega} \mathbf{u}^b \vec{\tau}_h \cdot \vec{n}, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n, \end{aligned} \quad (2.17)$$

and $\hat{\mathbf{u}} = \hat{\mathbf{u}}(\mathbf{w}_h^+, \mathbf{w}_h^-)$ is a numerical flux function. In this work, $\hat{\mathbf{u}}(\mathbf{w}_h^+, \mathbf{w}_h^-) = \{\mathbf{w}_h\}$ is used. However, other valid choices are available. See Chapter 3 for a complete analysis.

The additional state variable $\bar{\mathbf{q}}_h$ can be eliminated by rewriting it in terms of $\nabla \mathbf{w}_h$ and lifting operators. Begin by integrating by parts on (2.17):

$$\begin{aligned} \sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \bar{\mathbf{q}}_h &= \sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \nabla \mathbf{w}_h + \int_{\Gamma_i} [[\hat{\mathbf{u}} - \mathbf{w}_h]] \cdot \{\vec{\tau}_h\} + \int_{\Gamma_i} \{\hat{\mathbf{u}} - \mathbf{w}_h\} [[\vec{\tau}_h]] \\ &\quad + \int_{\partial\Omega} (\mathbf{u}^b - \mathbf{w}_h) \vec{\tau}_h \cdot \vec{n}, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n. \end{aligned} \quad (2.18)$$

Define the lifting operators $\vec{\mathbf{r}}_h$ and $\vec{\mathbf{l}}_h$ (see e.g. [7]) by the following problems: find $\vec{\mathbf{r}}_h(\mathbf{w}_h) \in [\mathcal{V}_h^p]^n$ and $\vec{\mathbf{l}}_h(\mathbf{w}_h) \in [\mathcal{V}_h^p]^n$ such that

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \vec{\mathbf{r}}_h(\mathbf{w}_h) = - \int_{\Gamma_i} [[\hat{\mathbf{u}} - \mathbf{w}_h]] \cdot \{\vec{\tau}_h\} - \int_{\partial\Omega} (\mathbf{u}^b - \mathbf{w}_h) \vec{\tau}_h \cdot \vec{n}, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n, \quad (2.19)$$

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \vec{\mathbf{l}}_h(\mathbf{w}_h) = - \int_{\Gamma_i} \{\hat{\mathbf{u}} - \mathbf{w}_h\} [[\vec{\tau}_h]], \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n. \quad (2.20)$$

Then, using (2.18) gives

$$\bar{\mathbf{q}}_h = \nabla \mathbf{w}_h - \vec{\mathbf{r}}_h(\mathbf{w}_h) - \vec{\mathbf{l}}_h(\mathbf{w}_h). \quad (2.21)$$

Thus, substituting (2.21) into (2.16), the discretization can be written as

$$R_{h,S,MF}(\mathbf{w}_h, \mathbf{v}_h) \equiv - \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{v}_h^T \mathbf{S}(\mathbf{w}_h, \nabla \mathbf{w}_h - \vec{\mathbf{r}}_h(\mathbf{w}_h) - \vec{\mathbf{l}}_h(\mathbf{w}_h)).$$

2.4 Temporal Discretization

The spatially discrete problem (2.14) can be written as a system of ODEs. In particular, let $\{\mathbf{v}_i\}$ for $i = 1, \dots, N$ be a basis of the space \mathcal{V}_h^p . Then, for all $\mathbf{w}_h \in \mathcal{V}_h^p$, there exists a

vector $W \in \mathbb{R}^N$ such that

$$\mathbf{w}_h(\mathbf{x}) = \sum_{i=1}^N W_i \mathbf{v}_i(\mathbf{x}).$$

Thus, the spatially discrete problem can be written as the following initial value problem: given $U(0)$, find $U(t)$ such that

$$\mathcal{M} \frac{dU}{dt} + R(U) = 0, \quad (2.22)$$

where R is the spatial residual vector,

$$R_i(W) = R_h(\mathbf{w}_h, \mathbf{v}_i),$$

and \mathcal{M} is the mass matrix,

$$\mathcal{M}_{ij} = \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{v}_i^T \mathbf{v}_j.$$

To fully discretize the problem, any number of standard ODE integration techniques can be applied. The interest of this work is steady problems. Thus, the temporal discretization will be used to march the solution in time to a steady state. In this case, temporal accuracy is not a primary concern. Thus, the temporal discretization need not be higher-order accurate. However, to avoid time step restrictions, an implicit method is used. In particular, the backward Euler method is applied. Thus, one time step of the fully discrete problem is as follows: given $U^m \in \mathbb{R}^N$, find $U^{m+1} \in \mathbb{R}^N$ such that

$$\frac{1}{\Delta t} \mathcal{M} (U^{m+1} - U^m) + R(U^{m+1}) = 0, \quad (2.23)$$

where m is the iteration number and Δt is the current time step length. The current time step is set based on an input CFL number. In particular, for each element, define

$$\Delta t_{\kappa} = CFL \frac{h_{\kappa}}{\lambda_{\kappa}},$$

where h_{κ} is a measure of the element grid spacing and λ_{κ} is the maximum convective wave speed over the element. Then,

$$\Delta t = \min_{\kappa \in T_h} \Delta t_{\kappa}.$$

At each successful iteration, the CFL number is increased by a user specified factor, and a new time step is computed.

2.5 Solution Technique

At each time step, the discrete, nonlinear problem (2.23) can be solved using Newton's method. For a solution tolerance of ϵ , the algorithm is composed of the following steps:

1. Set $k = 0$, $V^k = U^m$.
2. While $\|\frac{1}{\Delta t}\mathcal{M}(V^k - U^m) + R(V^k)\| > \epsilon$,

$$V^{k+1} = V^k - \left(\frac{1}{\Delta t}\mathcal{M} + \frac{\partial R}{\partial U}\Big|_{V^k} \right)^{-1} \left(\frac{1}{\Delta t}\mathcal{M}(V^k - U^m) + R(V^k) \right),$$

$$k = k + 1.$$

3. Set $U^{m+1} = V^k$.

Clearly, this algorithm requires the solution of the linear system

$$\left(\frac{1}{\Delta t}\mathcal{M} + \frac{\partial R}{\partial U}\Big|_{V^k} \right) (V^{k+1} - V^k) = - \left(\frac{1}{\Delta t}\mathcal{M}(V^k - U^m) + R(V^k) \right).$$

This system is inverted using the GMRES algorithm with element-block ILU0 or multigrid preconditioning with line reordering. See [35, 34] for the details of this procedure.

As noted in Section 2.4, steady problems are the primary interest of this work. Thus, given that temporal accuracy is not crucial, the nonlinear system (2.23) is generally not fully solved at each time step. Instead, only one sub-iteration is taken. In this case, U^{m+1} is given by

$$U^{m+1} = U^m - \left(\frac{1}{\Delta t}\mathcal{M} + \frac{\partial R}{\partial U}\Big|_{U^m} \right)^{-1} R(U^m).$$

Note that this statement is not true when the unsteady adaptation algorithm is used. For more details, see Chapter 6.

In either case, the solution is marched forward in time until $\|R(U^m)\|$ is less than the desired tolerance.

Chapter 3

Dual Consistency Analysis

The focus of this chapter is the impact of dual consistency on discretizations of source terms depending on the gradient of the state, like those seen in the SA turbulence model in Chapter 2. Dual consistency provides a connection between the continuous and discrete dual problems. This concept is made rigorous in Section 3.1, which gives a review of dual consistency and its importance. Section 3.2 analyzes the dual consistency of the source term discretizations shown in Chapter 2. The results of the analysis are confirmed by numerical experiments on a model problem, shown in Section 3.3. Section 3.4 shows results for the RANS-SA system.

3.1 Review of Dual Consistency

3.1.1 Definition

Consider the following primal problem: compute $\mathcal{J}(u)$, where $\mathcal{J} : \mathcal{V} \rightarrow \mathbb{R}$ is a functional of interest, \mathcal{V} is an appropriate function space, and $u \in \mathcal{V}$ solves

$$R(u, v) = 0, \quad \forall v \in \mathcal{V},$$

where $R : \mathcal{V} \times \mathcal{V} \rightarrow \mathbb{R}$ is the weak form of a PDE of interest. For simplicity, it is assumed that both the primal and dual problems are well-posed.

Let $\mathcal{W}_h^p \equiv \mathcal{V}_h^p + \mathcal{V}$, where

$$\mathcal{V}_h^p + \mathcal{V} \equiv \{h = f + g \mid f \in \mathcal{V}_h^p, g \in \mathcal{V}\}.$$

Then, consider a general DG discretization of the primal problem: find $u_h \in \mathcal{V}_h^p$ such

that

$$R_h(u_h, v_h) = 0, \quad \forall v_h \in \mathcal{V}_h^p,$$

where $R_h : \mathcal{W}_h^p \times \mathcal{W}_h^p \rightarrow \mathbb{R}$ is a semi-linear form derived from the weak form of the primal problem. It is assumed that this discrete problem is also well-posed.

Let $\mathcal{J}_h : \mathcal{W}_h^p \rightarrow \mathbb{R}$ be the discrete version of the functional of interest. Then, the discrete dual problem is given by the following statement: find $\psi_h \in \mathcal{V}_h^p$ such that

$$R'_h[u_h](v_h, \psi_h) = \mathcal{J}'_h[u_h](v_h), \quad \forall v_h \in \mathcal{V}_h^p.$$

Here, $R'_h[u_h](\cdot, \psi_h) : \mathcal{W}_h^p \rightarrow \mathbb{R}$ is the linear functional given by evaluating the Frechét derivative of the functional $N_{\psi_h} : \mathcal{W}_h^p \rightarrow \mathbb{R}$ at u_h , where, for fixed $\psi_h \in \mathcal{W}_h^p$,

$$N_{\psi_h}(w_h) = R_h(w_h, \psi_h), \quad \forall w_h \in \mathcal{W}_h^p.$$

Similarly, $\mathcal{J}'_h[u_h](\cdot) : \mathcal{W}_h^p \rightarrow \mathbb{R}$ is the linear functional given by evaluating the Frechét derivative of \mathcal{J}_h at u_h .

Dual consistency has been used in the analysis of DG methods by multiple authors. For example, [51, 50, 7] define dual consistency for linear problems. A general definition of dual consistency for nonlinear problems is provided by both Lu [72] and Hartmann [53]. Lu also defines asymptotic dual consistency for general nonlinear problems. The definitions used here follow [72].

Definition 1. *The discretization defined by the semi-linear form, R_h , and functional, \mathcal{J}_h , is said to be dual consistent if, given exact solutions $u \in \mathcal{V}$ and $\psi \in \mathcal{V}$ of the continuous primal and dual problems, respectively,*

$$R'_h[u](v, \psi) = \mathcal{J}'_h[u](v), \quad \forall v \in \mathcal{W}_h^p.$$

Definition 2. *The discretization defined by the semi-linear form, R_h , and functional, \mathcal{J}_h , is said to be asymptotically dual consistent if, given exact solutions $u \in \mathcal{V}$ and $\psi \in \mathcal{V}$ of the continuous primal and dual problems, respectively,*

$$\lim_{h \rightarrow 0} \left(\sup_{v \in \mathcal{W}_h^p, \|v\|_{\mathcal{W}_h^p} = 1} |R'_h[u](v, \psi) - \mathcal{J}'_h[u](v)| \right) = 0.$$

Clearly, all dual consistent discretizations are automatically asymptotically dual consistent. In this work, a discretization will be referred to as asymptotically dual consistent only if it is not also dual consistent.

3.1.2 Importance

For many types of discretization, algorithms involving the dual problem have become popular for design optimization, error estimation, and grid adaptation [62, 77, 36, 44, 18, 103]. It is well known that dual consistency can significantly impact the performance of these algorithms. For example, Collis and Heinkenschloss [33] showed that when applying a dual inconsistent SUPG method for linear advection-diffusion to an optimal control problem, superior results are obtained using a direct discretization of the continuous dual problem as opposed to the discrete dual problem derived from the primal discretization. Specifically, both the control function and the adjoint solution converge at a higher rate when the continuous dual problem is discretized directly.

For DG discretizations, Harriman *et al.* [51, 50] examined symmetric and non-symmetric interior penalty—SIPG and NIPG, respectively—DG methods for the solution of Poisson’s equation. They showed that to achieve optimal convergence rates for a linear functional output, the dual consistent method, i.e. SIPG, must be used. Lu [72] considered the impact of dual consistency on the accuracy of functional outputs computed using DG discretizations of the Euler and Navier-Stokes equations. He demonstrated the importance of implementing the boundary conditions on the primal problem in a dual consistent manner. In particular, when using dual consistent boundary conditions, super-convergent functional output results are obtained, while, when using a dual inconsistent boundary condition treatment, significant degradation of the output convergence rates is observed. More recently, Hartmann [53] has developed a framework for analyzing the dual consistency of DG discretizations. He uses the framework to analyze DG discretizations of many equations, including the compressible Euler and Navier-Stokes equations. For example, applying the framework to the SIPG discretization of the Navier-Stokes equations, a modification of the original SIPG method to make the boundary conditions dual consistent is derived. Similar to the results shown by Lu, Hartmann’s modification of the SIPG scheme produces superior results to the original, dual inconsistent boundary condition treatment.

In addition, it is well known that dual consistency can impact the convergence of the error in the primal solution [7]. For example, for many DG discretizations of Poisson’s equation, standard proofs of order of accuracy of the solution error in the L^2 norm exist. Typically these proofs rely on the Aubin-Nitsche “duality trick” [96, 88] to obtain an optimal estimate in the L^2 norm given an optimal estimate in the energy norm [20, 7]. The use of this duality argument requires that the scheme be dual consistent. Thus, some dual inconsistent methods—e.g. NIPG and the Baumann-Oden method—do not achieve optimal accuracy in the L^2 norm, and dual inconsistent methods that do achieve optimal accuracy in the L^2

norm are typically super-penalized [7].

3.2 Analysis of DG Discretizations of Source Terms

This section considers DG discretizations of source terms depending on the state and first derivatives of the state. For simplicity of notation, the analysis will be done on a scalar model problem. Despite this simplification, the results easily generalize to the RANS-SA case.

Let $u \in \mathcal{V} \equiv H^1(\Omega)$ be the solution of the following scalar problem:

$$\begin{aligned} -\nabla \cdot (\nu \nabla u) &= f(u, \nabla u) \quad \text{for } x \in \Omega \subset \mathbb{R}^n \\ u &= 0 \quad \text{for } x \in \partial\Omega, \end{aligned} \tag{3.1}$$

where ν is constant and $f \in \mathcal{C}^1(\mathbb{R}^{n+1})$ is the source term of interest. As before, it is assumed that this problem and its dual are well-posed.

Let $\mathcal{J} : \mathcal{V} \rightarrow \mathbb{R}$ be a functional of interest defined as

$$\mathcal{J}(w) = \int_{\Omega} J(w), \tag{3.2}$$

where $J \in \mathcal{C}(\mathbb{R})$. For simplicity, the functional output analyzed here does not include boundary integrals. The inclusion of boundary integrals would slightly alter the boundary conditions on the dual problem and the analysis of the boundary terms. However, these modifications are not central to the analysis of the source term discretization. For an analysis of dual consistency for equations without source terms that includes functionals with boundary integrals, see [72].

For the output defined in (3.2), the dual PDE is given by

$$-\nabla \cdot (\nu \nabla \psi) - D_1 f(u, \nabla u) \psi + \nabla \cdot (D_{\nabla u} f(u, \nabla u) \psi) = J'[u] \quad \text{for } x \in \Omega, \tag{3.3}$$

where ψ is the adjoint state, $D_1 f(u, \nabla u)$ is the partial derivative of f with respect to u evaluated at $(u, \nabla u)$ and

$$D_{\nabla u} f(u, \nabla u) = [D_2 f(u, \nabla u), \dots, D_{n+1} f(u, \nabla u)]^T$$

where $D_i f(u, \nabla u)$ is the partial derivative of f with respect to $\frac{\partial u}{\partial x_{i-1}}$ for $2 \leq i \leq n+1$, evaluated at $(u, \nabla u)$. The boundary conditions on the dual problem can be written in the

following weak form:

$$-\int_{\partial\Omega} \psi q = 0, \quad \forall q \in H^{-1/2}(\partial\Omega).$$

The analysis in Sections 3.2.1 and 3.2.2 considers whether the discrete dual problems for the source discretizations defined in Section 2.3.3 are consistent with (3.3) and the associated boundary conditions.

3.2.1 The Standard Weighting and Dual Consistent Methods

As will be shown, the simple approach of weighting by a test function and integrating leads to a dual inconsistent scheme. However, a dual consistent discretization can be constructed by adding terms to the discretization.

Consider the following DG discretization: find $u_h \in \mathcal{V}_h^p$ such that

$$R_h(u_h, v_h) \equiv B_h(u_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} v_h f(u_h, \nabla u_h) = 0, \quad \forall v_h \in \mathcal{V}_h^p, \quad (3.4)$$

where B_h is a consistent and dual consistent bilinear form for the diffusion operator (e.g. BR2 [15] or LDG [29, 31]). Clearly, the source term discretization in (3.4) is equivalent to that given for the RANS-SA case in (2.15). Furthermore, define the discrete functional of interest, $\mathcal{J}_h : \mathcal{W}_h^p \rightarrow \mathbb{R}$, as

$$\mathcal{J}_h(w_h) = \sum_{\kappa \in T_h} \int_{\kappa} J(w_h). \quad (3.5)$$

In the following dual consistency analysis, additional smoothness is assumed for the exact primal and dual solutions, u and ψ . In particular, it is assumed that $u, \psi \in H^2(\Omega)$. This smoothness assumption is common in the analysis of dual consistency for discretizations of second-order operators [7, 72].

Proposition 1. *The discretization defined by the semi-linear form R_h , defined in (3.4), together with the discrete functional \mathcal{J}_h , defined in (3.5), is dual inconsistent.*

Proof. Linearizing R_h about the exact solution and integrating by parts gives

$$\begin{aligned} R'_h[u](w_h, v_h) &= B_h(w_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} w_h (D_1 f(u, \nabla u) v_h - \nabla \cdot (D_{\nabla u} f(u, \nabla u) v_h)) \\ &\quad - \int_{\Gamma_i} (\llbracket w_h \rrbracket \cdot \{D_{\nabla u} f(u, \nabla u) v_h\} + \{w_h\} \llbracket D_{\nabla u} f(u, \nabla u) v_h \rrbracket) \\ &\quad - \int_{\partial\Omega} w_h (D_{\nabla u} f(u, \nabla u) v_h) \cdot \vec{n}. \end{aligned}$$

The assumptions $\psi \in H^2(\Omega)$, $u \in H^2(\Omega)$, and $f \in \mathcal{C}^1(\mathbb{R}^{n+1})$, imply that $\{D_{\nabla u} f(u, \nabla u) \psi\} =$

$D_{\nabla u}f(u, \nabla u)\psi$ and $\llbracket D_{\nabla u}f(u, \nabla u)\psi \rrbracket = 0$. Thus,

$$\begin{aligned} R'_h[u](v_h, \psi) &= B_h(v_h, \psi) - \sum_{\kappa \in T_h} \int_{\kappa} v_h (D_1 f(u, \nabla u)\psi - \nabla \cdot (D_{\nabla u}f(u, \nabla u)\psi)) \\ &\quad - \int_{\Gamma_i} \llbracket v_h \rrbracket \cdot (D_{\nabla u}f(u, \nabla u)\psi) - \int_{\partial\Omega} v_h (D_{\nabla u}f(u, \nabla u)\psi) \cdot \vec{n}. \end{aligned}$$

Evaluating the dual consistency using the discrete functional as defined in (3.5) gives

$$R'_h[u](v_h, \psi) - \mathcal{J}'_h[u](v_h) = (\mathcal{L}_{h,I}(u, \psi))(v_h) + (\mathcal{L}_{h,B}(u, \psi))(v_h),$$

where

$$(\mathcal{L}_{h,I}(u, \psi))(v_h) \equiv - \int_{\Gamma_i} \llbracket v_h \rrbracket \cdot (D_{\nabla u}f(u, \nabla u)\psi), \quad (3.6)$$

$$(\mathcal{L}_{h,B}(u, \psi))(v_h) \equiv - \int_{\partial\Omega} v_h (D_{\nabla u}f(u, \nabla u)\psi) \cdot \vec{n}. \quad (3.7)$$

In general, there exists $v_h \in \mathcal{V}_h^p$ such that at least $(\mathcal{L}_{h,I}(u, \psi))(v_h)$ does not vanish. Thus, the scheme is dual inconsistent. Due to the boundary condition on the dual problem, the boundary term, $(\mathcal{L}_{h,B}(u, \psi))(v_h)$, will vanish if $(D_{\nabla u}f(u, \nabla u)v_h \cdot \vec{n})|_{\partial\Omega} \in H^{-1/2}(\partial\Omega)$. However, if this condition does not hold, the boundary term will also contribute to the dual inconsistency. \square

Remark 1. It is possible to construct a dual consistent discretization by adding terms to the semi-linear form R_h . In particular, define a new bilinear form,

$$R_{h,DC}(w_h, v_h) \equiv R_h(w_h, v_h) + A_{h,I}(w_h, v_h) + A_{h,B}(w_h, v_h),$$

where $A_{h,I}$ will serve to eliminate the interior face dual inconsistency term, \mathcal{L}_I , and $A_{h,B}$ will serve to eliminate the boundary face dual inconsistency, \mathcal{L}_B . Furthermore, to maintain consistency, $A_{h,I}$ and $A_{h,B}$ must vanish when evaluated at u :

$$\begin{aligned} A_{h,I}(u, v_h) &= 0, \quad \forall v_h \in \mathcal{V}_h^p, \\ A_{h,B}(u, v_h) &= 0, \quad \forall v_h \in \mathcal{V}_h^p. \end{aligned}$$

The interior face and boundary face contributions to the dual inconsistency are examined separately. To eliminate the dual inconsistency from the interior faces, the following term

is added to the semi-linear form R_h :

$$A_{h,I}(w_h, v_h) = \int_{\Gamma_i} \llbracket w_h \rrbracket \cdot \{\vec{\beta}_i(w_h, v_h)\},$$

where dual consistency requires that $\{\vec{\beta}_i(u, \psi)\} = D_{\nabla u} f(u, \nabla u) \psi$.

To eliminate the boundary dual inconsistency, the following term is added to R_h :

$$A_{h,B}(w_h, v_h) = \int_{\partial\Omega} w_h \vec{\beta}_b(w_h, v_h) \cdot \vec{n},$$

where dual consistency requires $\vec{\beta}_b(u, \psi) = D_{\nabla u} f(u, \nabla u) \psi$.

Proposition 2. *Let B_h be a dual consistent bilinear form corresponding to the diffusion operator. Let $\vec{\beta}_i$ be such that $\vec{\beta}_i(w, v) = D_{\nabla u} f(w, \nabla w) v$ for all $w, v \in H^2(\Omega)$. Let $\vec{\beta}_b$ be such that $\vec{\beta}_b(w, v) = D_{\nabla u} f(w, \nabla w) v$ for all $w, v \in H^2(\Omega)$. Then, the semi-linear form given by*

$$\begin{aligned} R_{h,DC}(w_h, v_h) &= B_h(w_h, v_h) - \sum_{\kappa \in \mathcal{T}_h} \int_{\kappa} v_h f(w_h, \nabla w_h) \\ &\quad + \int_{\Gamma_i} \llbracket w_h \rrbracket \cdot \{\vec{\beta}_i(w_h, v_h)\} + \int_{\partial\Omega} w_h \vec{\beta}_b(w_h, v_h) \cdot \vec{n}, \end{aligned}$$

together with the discrete functional \mathcal{J}_h , defined in (3.5), is dual consistent.

Proof. Linearizing $R_{h,DC}$ gives

$$\begin{aligned} R'_{h,DC}[u](v_h, \psi) &= R'_h[u](v_h, \psi) + \int_{\Gamma_i} \llbracket v_h \rrbracket \cdot (D_{\nabla u} f(u, \nabla u) \psi) \\ &\quad + \int_{\partial\Omega} v_h (D_{\nabla u} f(u, \nabla u) \psi) \cdot \vec{n}, \quad \forall v_h \in \mathcal{W}_h^p. \end{aligned}$$

Thus,

$$R'_{h,DC}[u](v_h, \psi) - \mathcal{J}_h[u](v_h) = 0, \quad \forall v_h \in \mathcal{W}_h^p.$$

□

Remark 2. The choices of $\vec{\beta}_i$ and $\vec{\beta}_b$ are not fully determined by requiring dual consistency. One valid choice is given by

$$\vec{\beta}_i(w_h, v_h) = \{D_{\nabla u} f(w_h, \nabla w_h) v_h\}; \quad \vec{\beta}_b(w_h, v_h) = D_{\nabla u} f(w_h, \nabla w_h) v_h.$$

Then,

$$\begin{aligned}
R_{h,DC}(w_h, v_h) &= B_h(w_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} v_h f \\
&+ \int_{\Gamma_i} \llbracket w_h \rrbracket \cdot \{D_{\nabla u} f(w_h, \nabla w_h) v_h\} + \int_{\partial\Omega} w_h (D_{\nabla u} f(w_h, \nabla w_h) v_h) \cdot \vec{n}. \quad (3.8)
\end{aligned}$$

However, if necessary, one can construct more complex functions that satisfy the dual consistency requirement as well as add stability to the discretization. For example, given that the dual problem is a convection-diffusion-reaction problem, one option to add stability is to construct a dual flux that results in an upwind discretization of convective term of the dual problem.

Remark 3. In addition to being dual consistent, if B_h is a consistent bilinear form for the diffusion operator, the discretization given in Proposition 2 is *consistent* for any choice of $\vec{\beta}_i$ and $\vec{\beta}_b$ because, for the exact solution u , $\llbracket u \rrbracket = 0$ and $u|_{\partial\Omega} = 0$. Thus,

$$R_{h,DC}(u, v_h) = B_h(u, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} v_h f(u, \nabla u) = 0, \quad \forall v_h \in \mathcal{V}_h^p.$$

3.2.2 The Mixed Formulation

The mixed formulation discretization of the model problem is as follows: find $u_h \in \mathcal{V}_h^p$ such that

$$B_h(u_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} v_h f(u_h, \nabla u_h - \vec{r}_h(u_h) - \vec{\ell}_h(u_h)) = 0, \quad \forall v_h \in \mathcal{V}_h^p, \quad (3.9)$$

where \vec{r}_h and $\vec{\ell}_h$ are lifting operators analogous to those defined in Section 2.3.3. In particular, they are defined by the following problems: find $\vec{r}_h(u_h) \in [\mathcal{V}_h^p]^n$ and $\vec{\ell}_h(u_h) \in [\mathcal{V}_h^p]^n$ such that

$$\begin{aligned}
\sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \vec{r}_h(u_h) &= - \int_{\Gamma_i} \llbracket \hat{u}(u_h) - u_h \rrbracket \cdot \{\vec{\tau}_h\} \\
&- \int_{\partial\Omega} (u^b(u_h) - u_h) \vec{\tau}_h \cdot \vec{n}, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n, \quad (3.10)
\end{aligned}$$

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\tau}_h \cdot \vec{\ell}_h(u_h) = - \int_{\Gamma_i} \{\hat{u}(u_h) - u_h\} \llbracket \vec{\tau}_h \rrbracket, \quad \forall \vec{\tau}_h \in [\mathcal{V}_h^p]^n. \quad (3.11)$$

To complete the scheme, one must define the numerical flux functions \hat{u} and u^b . For the remainder of the chapter, it is assumed that these fluxes have the following properties:

1. for each interior edge, e , there exists a constant vector \vec{d}_e such that

$$\hat{u}(w_h) = \{w_h\} + \vec{d}_e \cdot \llbracket w_h \rrbracket,$$

2. $u^b = 0$.

Remark 4. While the assumptions on the numerical fluxes may seem restrictive, to the best of the author's knowledge, all existing mixed formulations that are consistent use fluxes satisfying these assumptions for problems with homogeneous Dirichlet boundary conditions [7].

The dual consistency of this discretization is considered in two parts. First, a restrictive condition is assumed which implies that the scheme is dual consistent. Then, the proof is extended to the general case, where only asymptotic dual consistency can be shown.

Proposition 3. *Let $D_{\nabla u}f(u, \nabla u)\psi \in [\mathcal{V}_h^p]^n$. Then, the DG formulation defined in (3.9) together with the functional \mathcal{J}_h defined in (3.5) forms a dual consistent discretization.*

Proof. Noting that the lifting operators \vec{r}_h and $\vec{\ell}_h$ are linear functionals and that $\vec{r}_h(u) = \vec{\ell}_h(u) = 0$, linearizing R_h about the exact solution gives

$$\begin{aligned} R'_h[u](w_h, v_h) &= B_h(w_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} v_h D_1 f(u, \nabla u) w_h \\ &\quad - \sum_{\kappa \in T_h} \int_{\kappa} v_h D_{\nabla u} f(u, \nabla u) \cdot (\nabla w_h - \vec{r}_h(w_h) - \vec{\ell}_h(w_h)). \end{aligned}$$

Thus, integrating by parts gives

$$\begin{aligned} R'_h[u](w_h, v_h) &= B_h(w_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} w_h (D_1 f(u, \nabla u) v_h - \nabla \cdot (D_{\nabla u} f(u, \nabla u) v_h)) \\ &\quad - \int_{\Gamma_i} (\llbracket w_h \rrbracket \cdot \{D_{\nabla u} f(u, \nabla u) v_h\} + \{w_h\} \llbracket D_{\nabla u} f(u, \nabla u) v_h \rrbracket) \\ &\quad - \int_{\partial\Omega} w_h (D_{\nabla u} f(u, \nabla u) v_h) \cdot \vec{n} \\ &\quad + \sum_{\kappa \in T_h} \int_{\kappa} v_h D_{\nabla u} f(u, \nabla u) \cdot (\vec{r}_h(w_h) + \vec{\ell}_h(w_h)). \end{aligned} \tag{3.12}$$

Using the assumptions on \hat{u} and u^b combined with the hypothesis that $D_{\nabla u}f(u, \nabla u)\psi \in$

$[\mathcal{V}_h^p]^n$,

$$\begin{aligned} \sum_{\kappa \in T_h} \int_{\kappa} (D_{\nabla u} f(u, \nabla u) \psi) \cdot \vec{r}_h(w_h) &= \int_{\Gamma_i} \llbracket w_h \rrbracket \cdot \{D_{\nabla u} f(u, \nabla u) \psi\} \\ &\quad + \int_{\partial \Omega} w_h (D_{\nabla u} f(u, \nabla u) \psi) \cdot \vec{n}, \end{aligned} \quad (3.13)$$

$$\sum_{\kappa \in T_h} \int_{\kappa} (D_{\nabla u} f(u, \nabla u) \psi) \cdot \vec{\ell}_h(w_h) = - \int_{\Gamma_i} (\vec{d}_e \cdot \llbracket w_h \rrbracket) \llbracket D_{\nabla u} f(u, \nabla u) \psi \rrbracket. \quad (3.14)$$

Substituting (3.13) and (3.14) into (3.12) gives

$$\begin{aligned} R'_h[u](w_h, v_h) &= B_h(w_h, v_h) - \sum_{\kappa \in T_h} \int_{\kappa} w_h (D_1 f(u, \nabla u) v_h - \nabla \cdot (D_{\nabla u} f(u, \nabla u) v_h)) \\ &\quad - \int_{\Gamma_i} (\hat{u}(w_h) \llbracket D_{\nabla u} f(u, \nabla u) v_h \rrbracket). \end{aligned}$$

Furthermore, by the assumptions on the smoothness of u , ψ , and f , $\llbracket D_{\nabla u} f(u, \nabla u) \psi \rrbracket = 0$.

Thus,

$$R'_h[u](v_h, \psi) = B_h(v_h, \psi) - \sum_{\kappa \in T_h} \int_{\kappa} v_h (D_1 f(u, \nabla u) \psi - \nabla \cdot (D_{\nabla u} f(u, \nabla u) \psi)).$$

Finally,

$$R'_h[u](v_h, \psi) - \mathcal{J}'_h[u](v_h) = 0, \quad \forall v_h \in \mathcal{W}_h^p,$$

which implies that the scheme is dual consistent. \square

Of course, in general, the assumption of $D_{\nabla u} f(u, \nabla u) \psi \in [\mathcal{V}_h^p]^n$ is not realistic. For $D_{\nabla u} f(u, \nabla u) \psi \notin [\mathcal{V}_h^p]^n$, Proposition 3 does not hold. In this case, the mixed formulation is only asymptotically dual consistent.

Proposition 4. *If $D_{\nabla u} f(u, \nabla u) \psi \in [H^{k+1}(\Omega)]^n$, where $1 \leq k \leq p$, then the DG discretization defined in (3.9) together with the functional \mathcal{J}_h defined in (3.5) forms an asymptotically dual consistent discretization.*

To simplify the proof, some preliminary notation and lemmas are required. In particular, let \mathcal{E}_h denote the set of all faces in the triangulation T_h . Define the jump operator, $\llbracket \cdot \rrbracket$, on boundary faces by $\llbracket s \rrbracket = s \vec{n}$ for scalar quantities and $\llbracket \vec{v} \rrbracket = \vec{v} \cdot \vec{n}$ for vector quantities. Define the average operator, $\{ \cdot \}$, on boundary faces by $\{ \vec{v} \} = \vec{v}$.

For the following lemmas, it is assumed that $\mathcal{W}_h^p = (\mathcal{V}_h^p + H^2(\Omega)) \cap H_0^1(\Omega)$. Furthermore, it is assumed that the set of triangulations, $[T_h]_{h>0}$, is quasi-uniform (see [38, 88] for

definition).

Lemma 1. *There exists a norm, $\|\cdot\|_* : \mathcal{W}_h^p \rightarrow \mathbb{R}$, and a constant, c , such that*

$$h^{-1/2} \sum_{e \in \mathcal{E}_h} \|\llbracket v \rrbracket\|_{0,e} \leq \sum_{e \in \mathcal{E}_h} h_{\kappa_e}^{-1/2} \|\llbracket v \rrbracket\|_{0,e} \leq c \|v\|_*, \quad \forall v \in \mathcal{W}_h^p,$$

where κ_e is such that $e \subset \partial\kappa_e$, $h = \max_{\kappa \in \mathcal{T}_h} h_\kappa$, and $h_\kappa = \sup_{x,y \in \kappa} |x - y|$.

Proof. An example of such a norm is used by Arnold *et al.* [7]. In particular,

$$\|v\|_*^2 \equiv \sum_{\kappa \in \mathcal{T}_h} (|v|_{1,\kappa}^2 + h_\kappa^2 |v|_{2,\kappa}^2) + \sum_{e \in \mathcal{E}_h} \|r_e(\llbracket v \rrbracket)\|_{0,\Omega}^2,$$

where

$$\int_{\Omega} r_e(\llbracket v \rrbracket) \cdot \vec{\tau} = - \int_e \llbracket v \rrbracket \cdot \{\vec{\tau}\} \quad \forall \vec{\tau} \in [\mathcal{V}_h^p]^n.$$

For the proof of the lemma for this norm, see [7], Section 4.1, or [20], Lemma 2. Note that only the existence of such a norm, not its particular form, is important here. \square

Lemma 2. *For a face, $e \in \mathcal{E}_h$, such that $e \subset \partial\kappa$, there exists a constant, c , such that, for all $v \in H^1(\kappa)$ and $w \in L^2(e)$,*

$$\int_e |vw| \leq ch_\kappa^{-1/2} (\|v\|_{0,\kappa} + h_\kappa |v|_{1,\kappa}) \|w\|_{0,e}.$$

Proof. Apply the Cauchy-Schwarz inequality, and then use

$$\|v\|_{0,e} \leq ch_\kappa^{-1/2} (\|v\|_{0,\kappa} + h_\kappa |v|_{1,\kappa}), \quad \forall v \in H^1(\kappa),$$

which is a standard trace theorem [7]. \square

Lemma 3. *For all $w \in H^{k+1}(\Omega)$, where $1 \leq k \leq p$, there exists a constant, c , such that*

$$\left(\sum_{\kappa \in \mathcal{T}_h} \|w - \Pi_h^p(w)\|_{1,\kappa}^2 \right)^{1/2} \leq ch^k |w|_{k+1,\Omega},$$

where $\Pi_h^p : L^2(\Omega) \rightarrow \mathcal{V}_h^p$ is the $L^2(\Omega)$ -orthogonal projection onto \mathcal{V}_h^p .

Proof. If $\Pi_\kappa^p : L^2(\kappa) \rightarrow P^p$ is the $L^2(\kappa)$ -orthogonal projection onto P^p , then

$$\Pi_h^p(v)|_\kappa = \Pi_\kappa^p(v|_\kappa), \quad \forall v \in L^2(\Omega).$$

To complete the proof, apply Proposition 1.134(iii) from [38] to each element κ and sum over the elements. \square

Proof (Proposition 4). Define $\vec{\pi} \in [\mathcal{V}_h^p]^n$ by

$$\pi_j = \Pi_h^p((D_{\nabla u} f(u, \nabla u)\psi)_j), \quad \text{for } j = 1, \dots, n.$$

Furthermore, define $\vec{\epsilon} \in [L^2(\Omega)]^n$ by

$$\epsilon_j = (D_{\nabla u} f(u, \nabla u)\psi)_j - \pi_j, \quad \text{for } j = 1, \dots, n.$$

By assumption, $D_{\nabla u} f(u, \nabla u)\psi \in [H^1(\Omega)]^n$. Thus, $\vec{\epsilon}|_{\kappa} \in [H^1(\kappa)]^n$, for all $\kappa \in T_h$. From the proof of Proposition 3, it is clear that

$$\begin{aligned} E_h(v_h) \equiv R'_h[u](v_h, \psi) - \mathcal{J}'_h[u](v_h) &= \sum_{\kappa \in T_h} \int_{\kappa} (D_{\nabla u} f(u, \nabla u)\psi) \cdot (\vec{r}_h(v_h) + \vec{\ell}_h(v_h)) \\ &\quad - \int_{\Gamma} \llbracket v_h \rrbracket \cdot (D_{\nabla u} f(u, \nabla u)\psi), \quad \forall v_h \in \mathcal{W}_h^p, \end{aligned}$$

where $\Gamma \equiv \Gamma_i \cup \partial\Omega$. Thus,

$$\begin{aligned} E_h(v_h) &= \sum_{\kappa \in T_h} \int_{\kappa} (\vec{\pi} + \vec{\epsilon}) \cdot (\vec{r}_h(v_h) + \vec{\ell}_h(v_h)) \\ &\quad - \int_{\Gamma} \llbracket v_h \rrbracket \cdot (\vec{\pi} + \vec{\epsilon}), \quad \forall v_h \in \mathcal{W}_h^p. \end{aligned} \tag{3.15}$$

From (2.19), (2.20), and the assumptions on \hat{u} and u^b ,

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\pi} \cdot \vec{r}_h(v_h) = \int_{\Gamma} \llbracket v_h \rrbracket \cdot \{\vec{\pi}\}, \tag{3.16}$$

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\pi} \cdot \vec{\ell}_h(v_h) = - \int_{\Gamma_i} (\vec{d}_e \cdot \llbracket v_h \rrbracket) \llbracket \vec{\pi} \rrbracket. \tag{3.17}$$

Substituting (3.16) and (3.17) into (3.15) gives

$$E_h(v_h) = \sum_{\kappa \in T_h} \int_{\kappa} \vec{\epsilon} \cdot (\vec{r}_h(v_h) + \vec{\ell}_h(v_h)) - \int_{\Gamma} \llbracket v_h \rrbracket \cdot \{\vec{\epsilon}\} - \int_{\Gamma_i} (\vec{d}_e \cdot \llbracket v_h \rrbracket) \llbracket \vec{\pi} \rrbracket.$$

By the definition of $\vec{\epsilon}$,

$$\sum_{\kappa \in T_h} \int_{\kappa} \vec{\epsilon} \cdot \vec{z}_h = 0, \quad \forall \vec{z}_h \in [\mathcal{V}_h^p]^n.$$

Furthermore, since $\llbracket D_{\nabla u} f(u, \nabla u) \psi \rrbracket = 0$, it is clear that $\llbracket \vec{\pi} \rrbracket = -\llbracket \vec{\epsilon} \rrbracket$. Thus,

$$E_h(v_h) = - \int_{\Gamma} \llbracket v_h \rrbracket \cdot \{\vec{\epsilon}\} + \int_{\Gamma_i} (\vec{d}_e \cdot \llbracket v_h \rrbracket) \llbracket \vec{\epsilon} \rrbracket.$$

Then, using the assumption on \hat{u} , $E_h(v_h)$ can be rewritten as

$$E_h(v_h) = - \int_{\partial\Omega} v_h \vec{\epsilon} \cdot \vec{n} - \int_{\Gamma_i} \llbracket v_h \rrbracket \cdot \left[\left(\frac{1}{2} - \vec{d}_e \cdot \vec{n}^+ \right) \vec{\epsilon}^+ + \left(\frac{1}{2} + \vec{d}_e \cdot \vec{n}^+ \right) \vec{\epsilon}^- \right],$$

where $(\cdot)^+$ and $(\cdot)^-$ refer to trace values taken from opposite sides of an interior face, and \vec{n}^+ is the outward pointing unit normal for κ^+ .

Applying Lemma 2 to each edge in the triangulation, one can show that

$$|E_h(v_h)| \leq \sum_{e \in \mathcal{E}_h} \sum_{j=1}^n \left[c_e h_{\kappa_e}^{-1/2} (\|\epsilon_j\|_{0, \kappa_e} + h_{\kappa_e} |\epsilon_j|_{1, \kappa_e}) \|\llbracket v_h \rrbracket\|_{0, e} \right].$$

Thus,

$$\begin{aligned} |E_h(v_h)| &\leq \sum_{e \in \mathcal{E}_h} \sum_{j=1}^n \left[c_e h_{\kappa_e}^{-1/2} (\|\epsilon_j\|_{1, \kappa_e}^2)^{1/2} \|\llbracket v_h \rrbracket\|_{0, e} \right], \\ &\leq \sum_{e \in \mathcal{E}_h} \sum_{j=1}^n \left[c_e h_{\kappa_e}^{-1/2} \left(\sum_{\kappa \in T_h} \|\epsilon_j\|_{1, \kappa}^2 \right)^{1/2} \|\llbracket v_h \rrbracket\|_{0, e} \right], \\ &\leq C \sum_{j=1}^n \left[\left(\sum_{\kappa \in T_h} \|\epsilon_j\|_{1, \kappa}^2 \right)^{1/2} \right] \times \left[\sum_{e \in \mathcal{E}_h} h_{\kappa_e}^{-1/2} \|\llbracket v_h \rrbracket\|_{0, e} \right]. \end{aligned}$$

Finally, applying Lemmas 1 and 3 gives

$$|E_h(v_h)| \leq C \|v_h\|_* h^k \sum_{j=1}^n |(D_{\nabla u} f(u, \nabla u) \psi)_j|_{k+1, \Omega}.$$

Thus, as $h \rightarrow 0$, $|E_h(v_h)| \rightarrow 0$ for all $v_h \in \mathcal{W}_h^p$, which implies that the scheme is asymptotically dual consistent. \square

3.3 Model Problem Results

As a demonstration of the effects of dual consistency, a simple test problem based on a nonlinear ODE is considered. The effect of dual consistency on the convergence rates of the solution and adjoint solution errors as well as a simple functional output is demonstrated.

Consider the following ODE:

$$\begin{aligned} -((\nu + u)u_x)_x - cu_xu_x &= g \quad \text{for } x \in (0, 1), \\ u(0) = u(1) &= 0, \end{aligned}$$

where $\nu = 1$ and $c = \frac{1}{2}$. Setting

$$g(x) = \pi^2((\nu + \sin(\pi x)) \sin(\pi x) - (1 + c) \cos^2(\pi x)),$$

it is easy to show that the exact solution is given by

$$u(x) = \sin(\pi x).$$

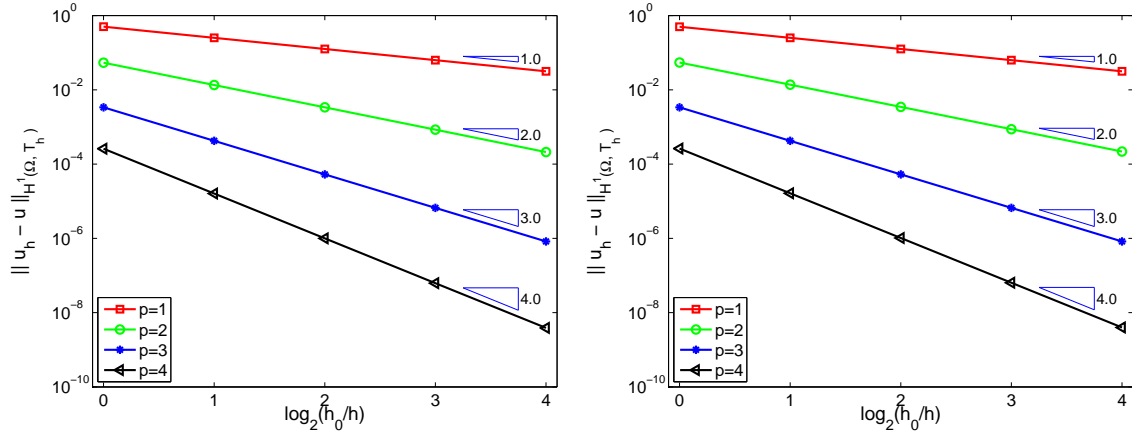
This nonlinear problem has been solved using three discretizations: the standard weighting method as shown in (3.4), a dual consistent method with a penalty term as shown in (3.8), and an asymptotically dual consistent mixed method with $\hat{u} = \{u\}$. In all cases, the BR2 scheme is used to discretize the nonlinear diffusion operator.

Figure 3-1 shows the error in the primal solution versus grid refinement. The error is measured in a broken H^1 norm defined by

$$\|v\|_{H^1(\Omega; T_h)}^2 = \sum_{\kappa \in T_h} \int_{\kappa} (v^2 + v_x^2).$$

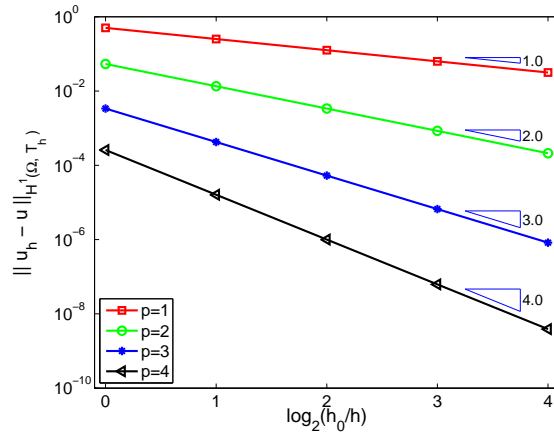
In this norm, all three schemes produce essentially the same error in the primal solution. However, as shown in Figure 3-2, the dual consistent and asymptotically dual consistent schemes produce superior results when the error is measured in the L^2 norm. In particular, for the dual inconsistent discretization, the L^2 norm of the error is proportional to h^p for even p and proportional to h^{p+1} for odd p . It is not clear why the even and odd p results show different asymptotic rates, but similar results have been observed for other dual inconsistent discretizations [51].

Alternatively, the dual consistent and asymptotically dual consistent discretizations give $O(h^{p+1})$ for all p tested. Furthermore, it is interesting to note that the asymptotically dual consistent method produces essentially exactly the same results as the dual consistent discretization. This fact shows that, for the asymptotically dual consistent scheme, the contribution of the dual inconsistency error to the total error is sufficiently high-order that it does not degrade the asymptotic rate convergence rate of the L^2 error. This result is not surprising given the form of the dual consistency error derived in Section 3.2.2.



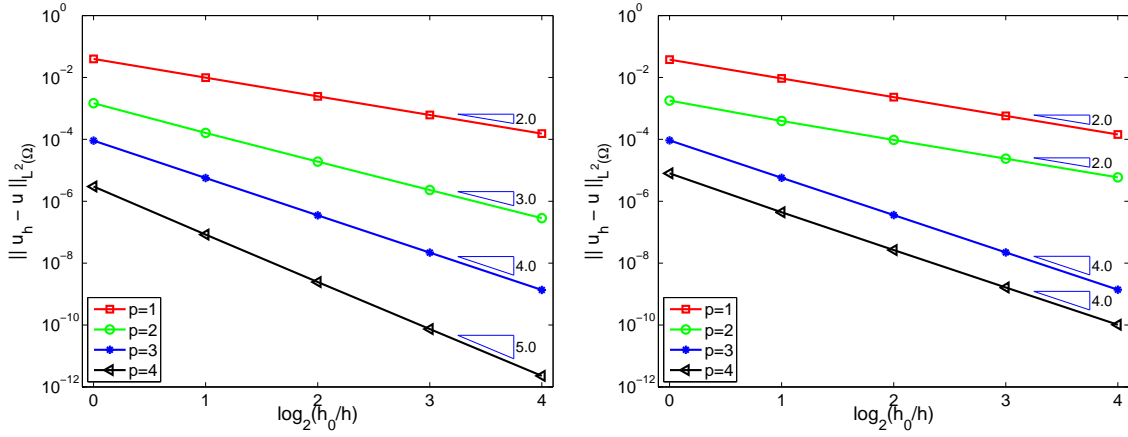
(a) Dual consistent

(b) Standard weighting



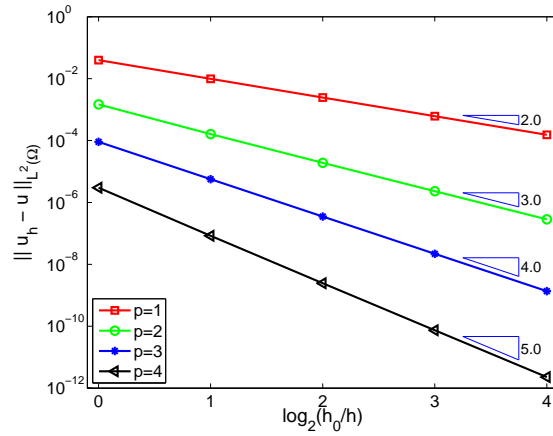
(c) Mixed formulation

Figure 3-1: Primal error in the broken H^1 norm for the scalar model problem



(a) Dual consistent

(b) Standard weighting



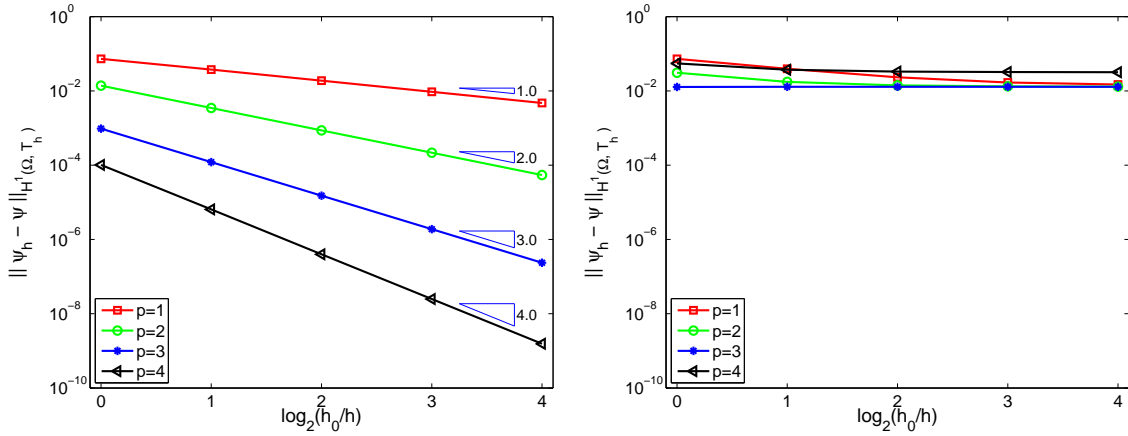
(c) Mixed formulation

Figure 3-2: Primal error in the L^2 norm for the scalar model problem

Let the output of interest be

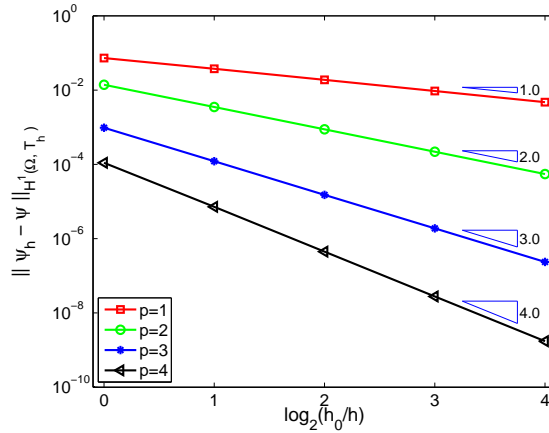
$$\mathcal{J}(u) = \frac{1}{2} \int_0^1 (w - u)^2,$$

where $w(x) = 2 \sin(\pi x)$. Then, examining the adjoint solution error, one can see that the dual consistent and asymptotically dual consistent schemes are superior for computing the adjoint. Figure 3-3 shows the adjoint error in the broken H^1 norm. The adjoint error



(a) Dual consistent

(b) Standard weighting

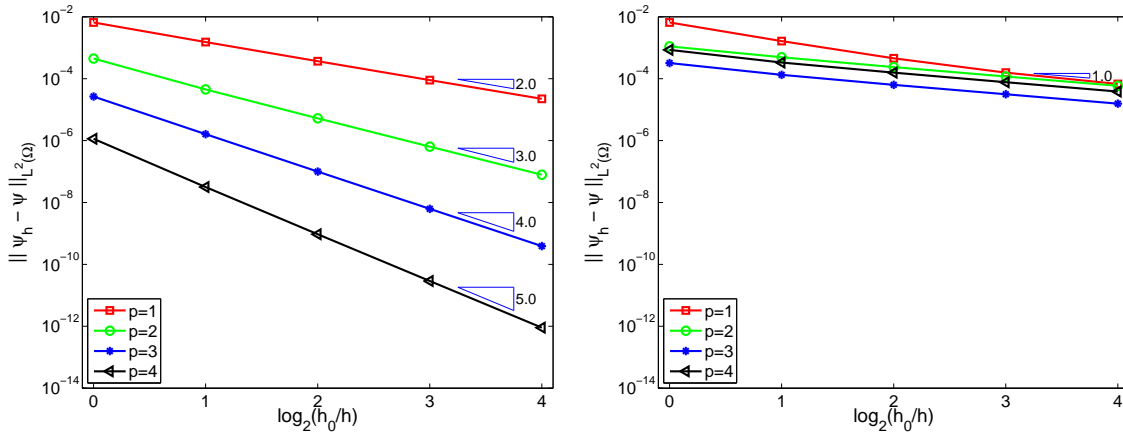


(c) Mixed formulation

Figure 3-3: Adjoint error in the broken H^1 norm for the scalar model problem

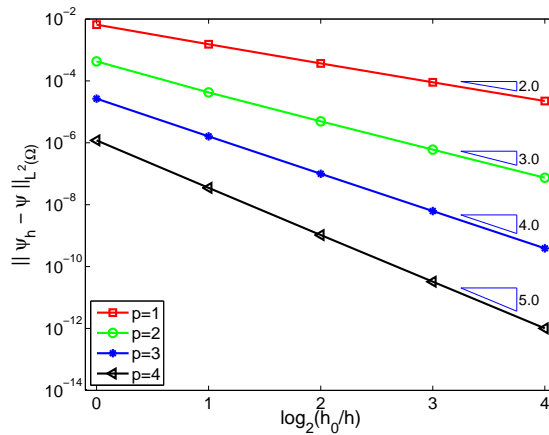
is computed relative to a 40th order solution of a Galerkin spectral discretization of the dual problem. When using the dual inconsistent discretization, the broken H^1 norm of the adjoint error does not converge to zero with grid refinement. For the dual consistent and

asymptotically dual consistent schemes, this error converges at $O(h^p)$. Similarly, Figure 3-4 shows that the L^2 norm of the adjoint error converges at $O(h)$ when using the dual inconsistent scheme, regardless of p , while, for the dual consistent and asymptotically dual consistent schemes, this error converges at $O(h^{p+1})$.



(a) Dual consistent

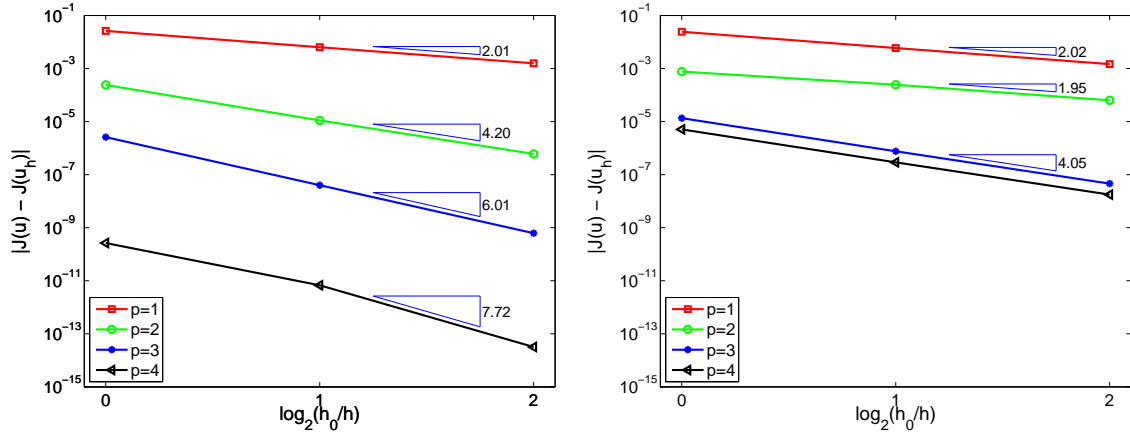
(b) Standard weighting



(c) Mixed formulation

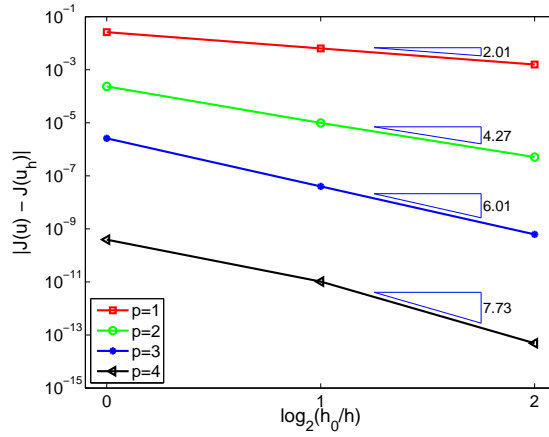
Figure 3-4: Adjoint error in the L^2 norm for the scalar model problem

Since the exact solution is known, computing the exact functional output is trivial, enabling comparison of the computed result with the exact value, $\mathcal{J}(u) = 1/4$. Figure 3-5 shows the error in the computed functional for the three discretizations considered. The figure shows that, as in the state and adjoint results, the performance of the dual consistent and asymptotically dual consistent schemes is very similar. Both schemes achieve $O(h^{2p})$ for



(a) Dual consistent

(b) Standard weighting



(c) Mixed formulation

Figure 3-5: Functional output error for the scalar model problem

$1 \leq p \leq 4$. However, for the dual inconsistent scheme, the convergence rate of the functional is $O(h^p)$ for even p and $O(h^{p+1})$ for odd p . Thus, the dual consistent and asymptotically dual consistent discretizations predict the functional with greater accuracy than the dual inconsistent discretization for similar numbers of degrees of freedom.

3.4 RANS Results

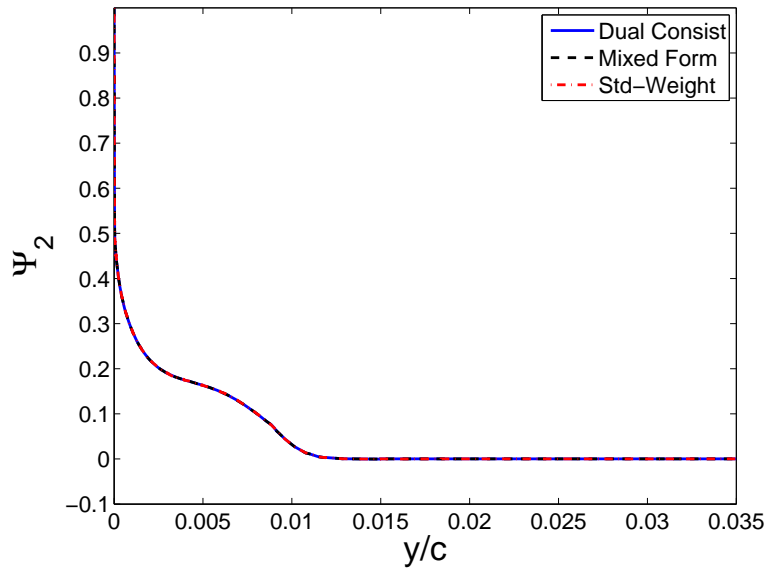
Three test cases are considered. First, the adjoint solution and its derivatives are examined for flow over a flat plate solved using all three source term discretizations. Second, a grid refinement study is conducted for flow around a NACA 0012 airfoil using both the standard weighting and dual consistent discretizations. Finally, a RAE 2822 airfoil case is solved using the dual consistent scheme.

3.4.1 Flat Plate Adjoint Results

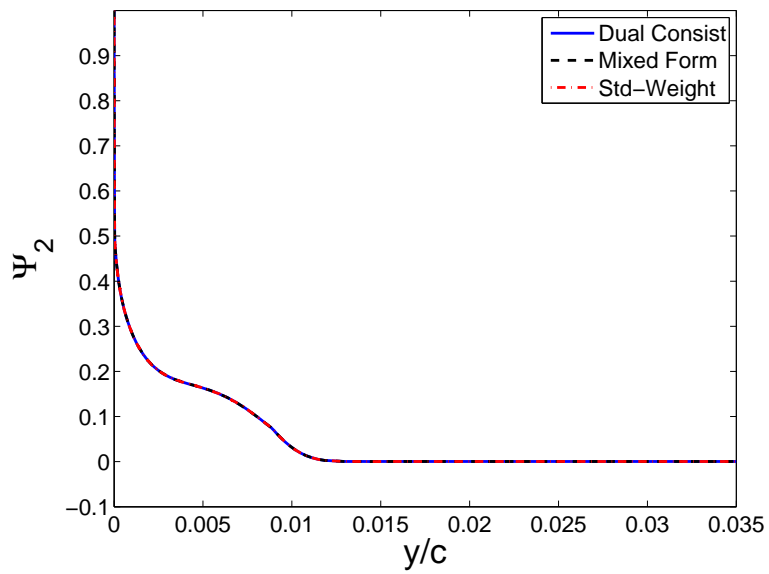
To begin, adjoint solutions for flow over a flat plate are examined. In particular, the test case is $M_\infty = 0.25$, $Re_c = 1 \times 10^7$ flow over a flat plate with zero pressure gradient. The output of interest is the drag force on the plate.

Figures 3-6 and 3-7 show $p = 3$ adjoint solution profiles computed on an 866 element mesh and on a 3464 element mesh generated by uniformly refining the 866 element mesh. Even on the coarser mesh, it is difficult to see a difference between the profiles. This fact is consistent with the adjoint results from the model problem. Specifically, for the model problem, the adjoint error measured in the L^2 norm converges for all three schemes but at different rates.

However, as can be seen in Figures 3-8 and 3-9, there are clearly differences in the derivatives of the adjoint. In particular, the figures show that the y derivatives of the adjoint computed using the standard weighting scheme are highly oscillatory, while the dual consistent and mixed formulation results are relatively smooth. Furthermore, the oscillations in the derivatives produced by the standard weighting scheme are not decreasing with grid refinement. This result is consistent with the results obtained for the adjoint error measured in the broken H^1 norm for the model problem. In that case, the adjoint error did not converge for the standard weighting scheme. Finally, the figure shows that, while the dual inconsistency in the standard weighting scheme is caused only by the discretization of the turbulence model—none of the other equations have gradient dependent source terms—it adversely affects the adjoint components corresponding to the other equations as well. This adverse effect is not surprising given the coupling between the turbulence model and the flow equations.

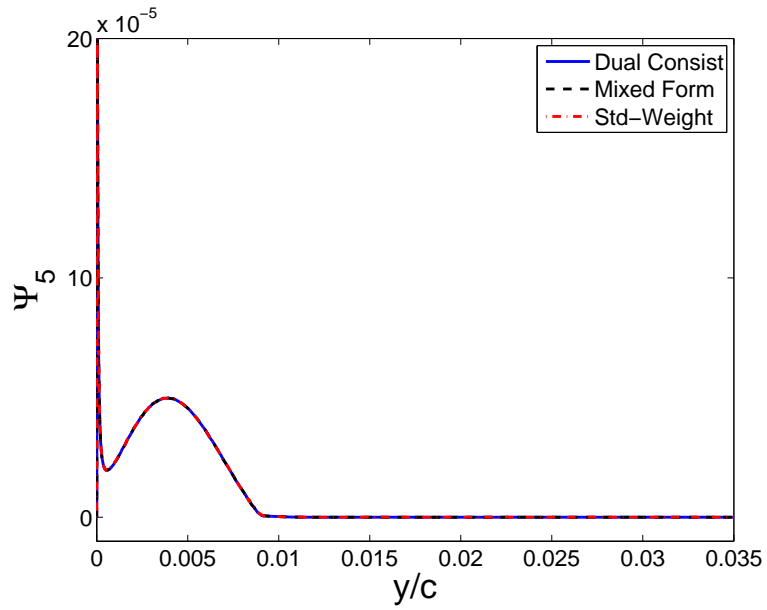


(a) 866 element mesh

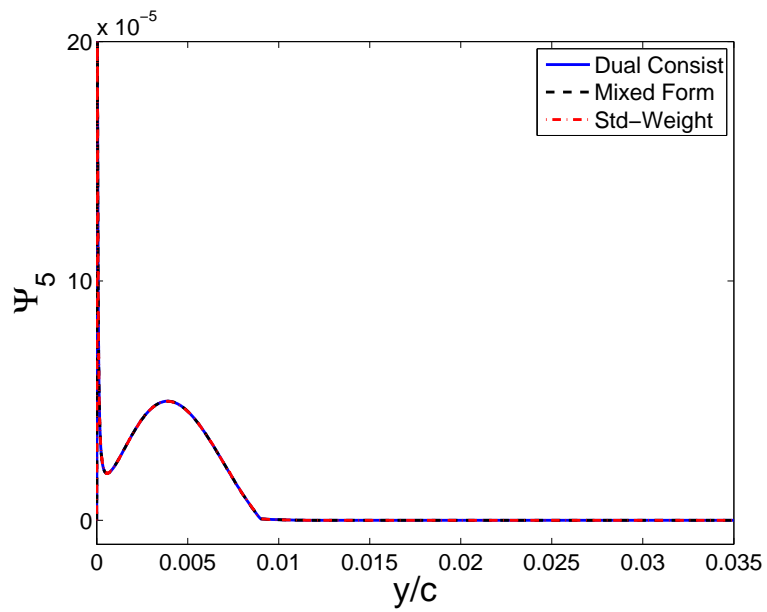


(b) 3464 element mesh

Figure 3-6: Comparison of x -momentum adjoint solution profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$ for flow over a flat plate

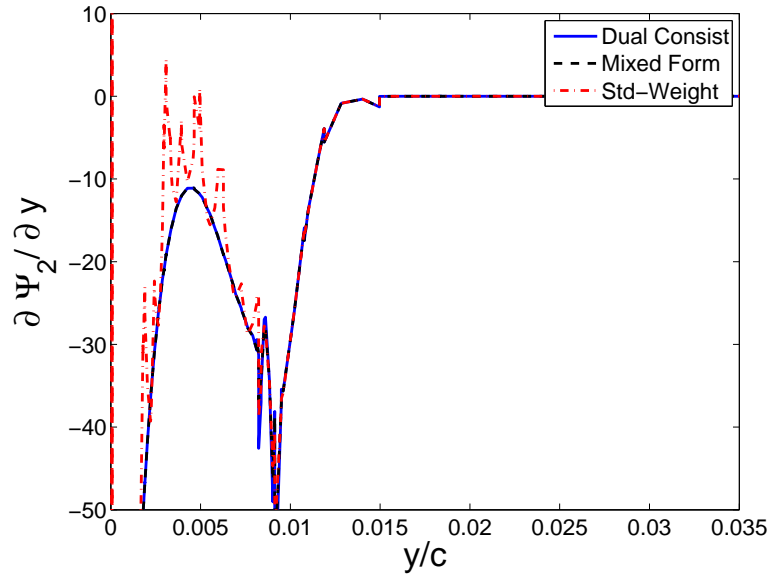


(a) 866 element mesh

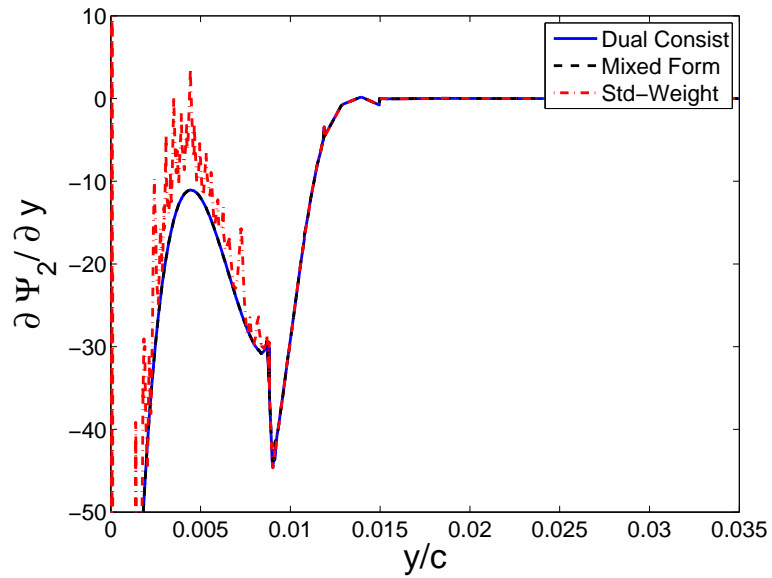


(b) 3464 element mesh

Figure 3-7: Comparison of turbulence model adjoint solution profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$

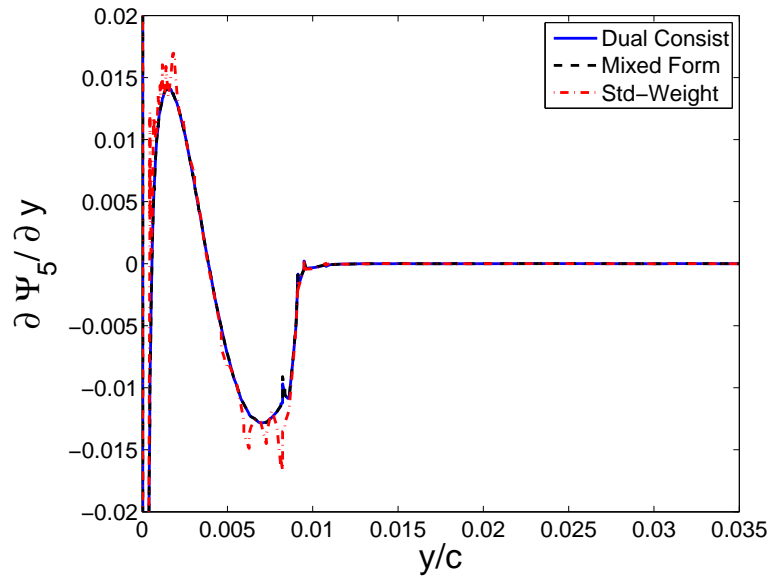


(a) 866 element mesh

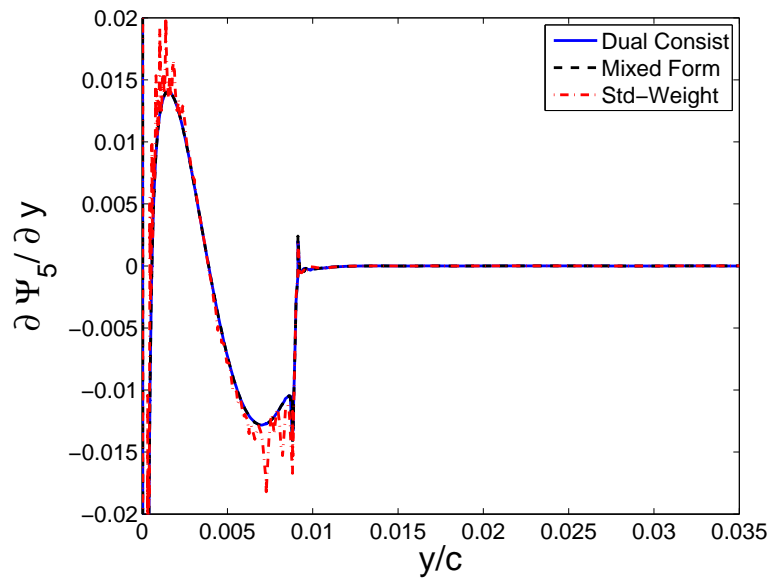


(b) 3464 element mesh

Figure 3-8: Comparison of x -momentum adjoint y -derivative profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$ for flow over a flat plate



(a) 866 element mesh



(b) 3464 element mesh

Figure 3-9: Comparison of turbulence model adjoint y -derivative profiles for dual consistent, mixed formulation, and standard weighting discretizations at $x/c = 0.5$ for flow over a flat plate

3.4.2 NACA 0012 Refinement Study

To examine the accuracy of the discretizations, a grid refinement study is conducted for $M_\infty = 0.25$, $Re_c = 1 \times 10^7$, $\alpha = 0$ flow over a NACA 0012 airfoil. The airfoil is modified slightly such that the trailing edge has zero thickness. Specifically, the airfoil is defined by

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

for $0 \leq x \leq 1$. The outer boundary of the computational domain is the circle of radius 40, centered at the origin. To generate a family of meshes, a coarse, 1280 element linear mesh of this domain was generated by meshing a rectangle and then mapping the mesh nodes to the physical domain using transfinite interpolation. The resulting mesh is shown in Figure 3-10. Given the coarse mesh, medium, fine, and ultra-fine linear meshes were generated by uniformly refining the coarse mesh.

Results are presented for $p = 1, 2, 3, 4$ for the dual consistent and standard weighting discretizations on the coarse, medium, and fine meshes. Only $p = 1, 2$ results have been obtained on the ultra-fine mesh, and no mixed formulation results are presented because this discretization has not been implemented in parallel.

All results are isoparametric. Thus, curved meshes are required. To generate these curved meshes, a linear elasticity based node movement algorithm, described in Section 5.1.3, was applied to the linear meshes. To illustrate the differences between the linear and curved meshes, the $q = 1$ and $q = 3$ —where q denotes the polynomial order of the mesh—versions of the coarse mesh are shown in Figure 3-11. The $q = 3$ mesh is plotted by linearly interpolating between the edge nodes. The figure shows the leading edge region of the airfoil, where the differences between the $q = 1$ and $q = 3$ meshes are the largest. Far away from the airfoil, the two meshes are essentially the same.

To examine the drag error, an exact value of the drag must be chosen. Since the exact solution is not known, the “exact” drag in this work is computed by extrapolating high-order results. Specifically, the error is computed relative to a drag value of 75.612866 counts, which is obtained by extrapolating the dual consistent, $p = 5$ drag results from the coarse and medium meshes, assuming the drag error for $p = 5$ is proportional to h^3 . This rate was chosen after examining the order of accuracy obtained by the $p = 1, 2, 3, 4$ results for different $p = 5$ rate assumptions. The results of this examination are shown in Figure 3-12. The figure shows the order of accuracy obtained for $p = 1, 2, 3, 4$ by refining from the coarse mesh to the fine mesh versus the assumed order of accuracy used to extrapolate the $p = 5$ drag results to the “exact” value. The $p = 1, 2$ results are relatively insensitive to the $p = 5$ rate assumption. This result gives confidence that the computed error and order of

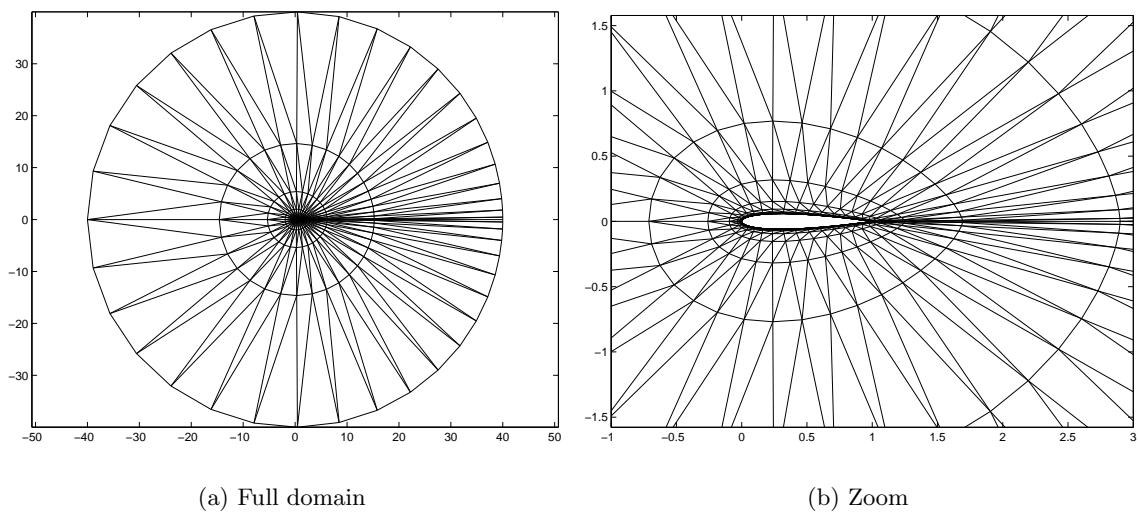


Figure 3-10: Coarse (1280 elements), linear mesh of the NACA 0012 airfoil

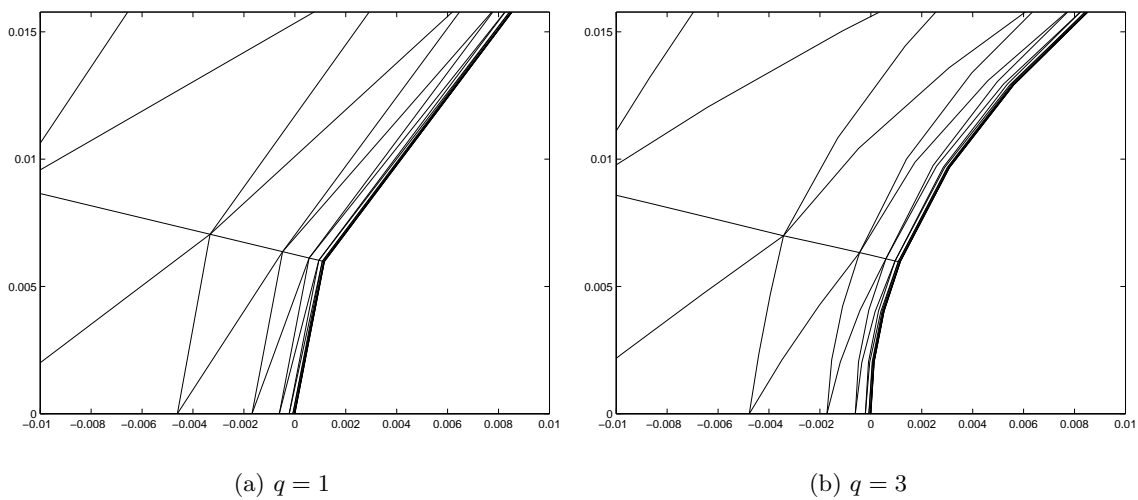


Figure 3-11: Leading edge of the $q = 1$ and $q = 3$ versions of the coarse mesh of the NACA 0012 airfoil

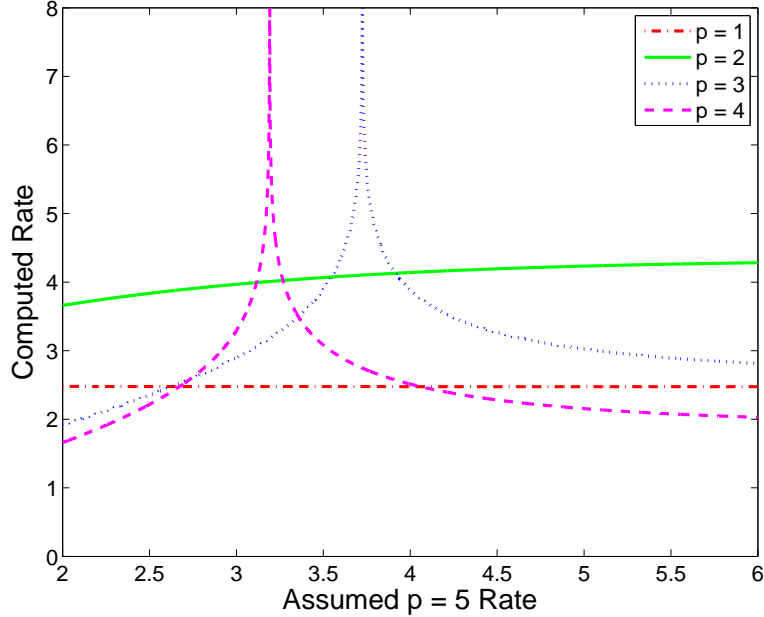
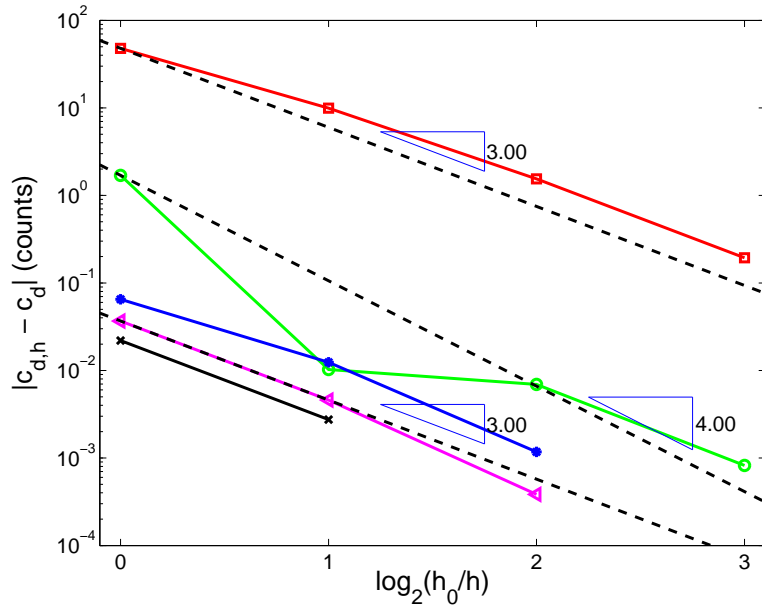


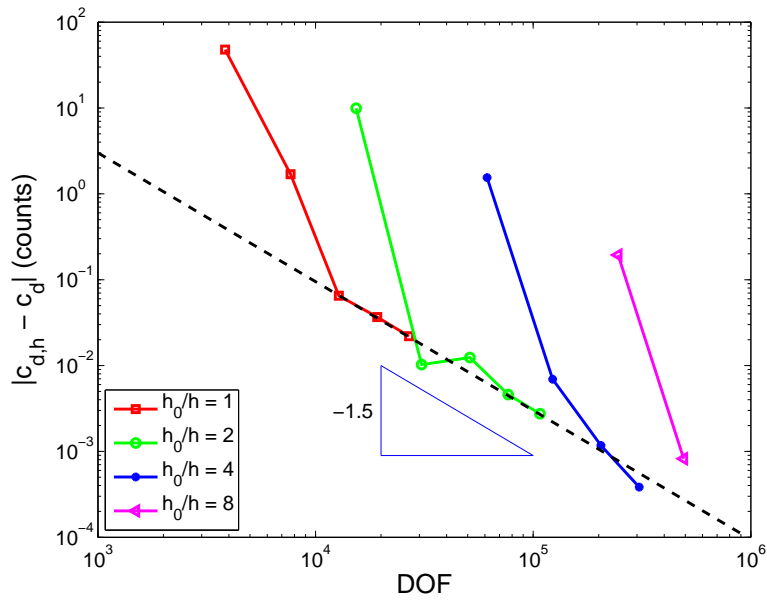
Figure 3-12: Order of accuracy obtained for $p = 1, 2, 3, 4$ versus order of accuracy assumed for $p = 5$ for drag error for flow over a NACA 0012 airfoil, computed using the dual consistent discretization

accuracy for $p = 1, 2$ are very close to the true values. Alternatively, the $p = 3, 4$ results are quite sensitive to the assumed rate. However, it appears that assumed rates between two and three give the most consistent results. Assuming $p = 5$ is converging faster than approximately $O(h^4)$ leads to a scenario where $p = 3$ obtains between $O(h^3)$ and $O(h^4)$ while $p = 4$ gives between $O(h^2)$ and $O(h^3)$. This result makes little sense. On the other hand, assuming $p = 5$ obtains $O(h^3)$ gives that $p = 3, 4$ both achieve roughly $O(h^3)$. While this rate is suboptimal, it can be reasonably explained if the solution is not smooth. Thus, $O(h^3)$ is chosen here.

Figure 3-13 shows the drag error results obtained using the dual consistent discretization. In particular, Figure 3-13(a) shows the drag error versus grid refinement. The $p = 1$ order of accuracy is slightly better than the expected optimal rate of $O(h^2)$, and the $p = 2$ accuracy appears optimal at approximately $O(h^4)$. Alternatively, $p = 3, 4$ do not achieve the asymptotic convergence rates expected for smooth problems. While the precise rates obtained are dependent on the assumed $p = 5$ rate, no assumed rate gives optimal performance for both $p = 3$ and $p = 4$, as shown in Figure 3-12. This suboptimal behavior is attributed to under-resolution in the boundary layer edge region. Neglecting the effect of the viscosity of the fluid, the RANS-SA system has discontinuous first derivatives at the boundary layer edge in both the primal and adjoint solutions. The discontinuities in the



(a) Drag error versus h for $p = 1$ through $p = 5$ ($p = 1 : \square$, $p = 2 : \circ$, $p = 3 : *$, $p = 4 : \triangleleft$, $p = 5 : \times$)



(b) Drag error versus DOF

Figure 3-13: Drag error for flow over a NACA 0012 computed using the dual consistent discretization

adjoint derivatives can be seen in Figures 3-8 and 3-9. While the addition of viscosity formally eliminates these discontinuities, given that the solution in this region is not resolved, the effect on the numerical solution is the same.

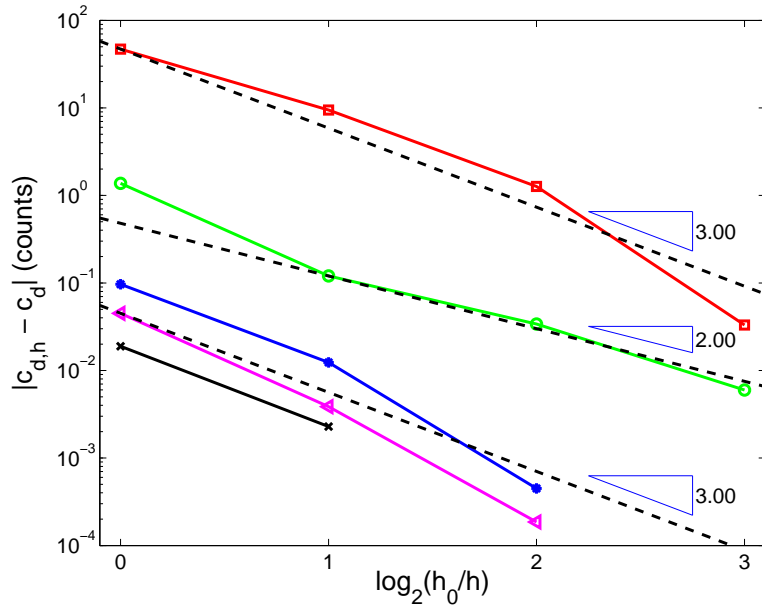
Figure 3-13(b) shows the drag error results again. In this figure, the drag error versus DOF is shown for p refinement on the coarse, medium, fine, and ultra-fine meshes. The figure shows that, on each mesh, as p is increased, the drag error eventually becomes proportional to $DOF^{-3/2}$. In this regime, the error is dominated by the boundary layer edge region, as will be confirmed by examining an error estimate in Section 4.3.2. To further confirm that the boundary layer edge singularity is the root cause of the suboptimal accuracy, a laminar case is examined using the same set of meshes used here. The laminar case has no boundary layer edge singularity, and, as expected, the order of accuracy does not asymptote to $O(h^3)$ as the resolution increases. The results of this study are shown in Appendix B.

The results of the refinement study for the standard weighting discretization are shown in Figure 3-14. As in the dual consistent case, the $p = 1$ discretization gives better than expected results. However, the effect of the dual inconsistency can be observed for the $p = 2$ scheme, which obtains only $O(h^2)$ accuracy. This rate is consistent with the suboptimal results observed for the standard weighting discretization applied to the model problem. For $p \geq 3$, it appears that errors due to the boundary layer edge singularity are dominant. Thus, no degradation of the accuracy due to the dual inconsistency is observed. In fact, in this regime, the standard weighting discretization appears to be slightly more accurate for this case.

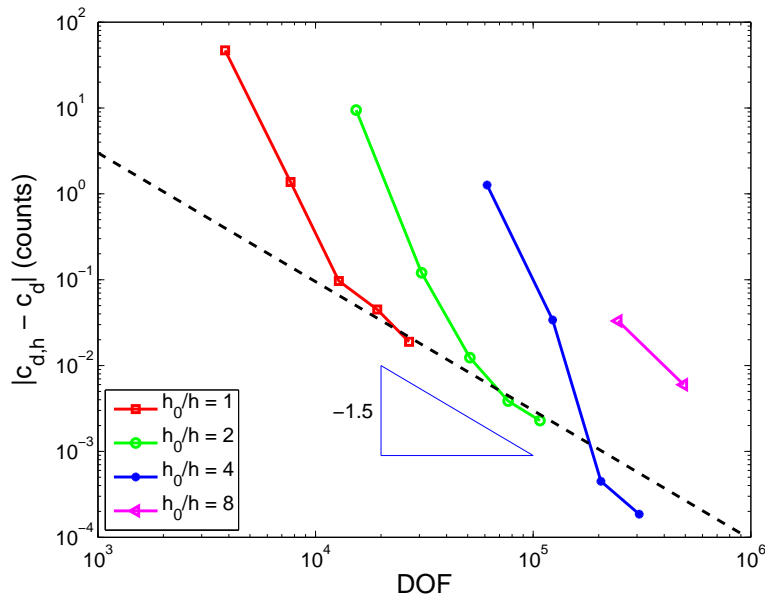
3.4.3 RAE 2822 Refinement Study

To demonstrate the performance of the dual consistent scheme on a slightly more difficult problem, a RAE 2822 airfoil case is considered. The freestream flow conditions are $M_\infty = 0.6$, $Re_c = 6.3 \times 10^6$, $\alpha = 2.57^\circ$. The computational domain is the circle of radius 40, centered at the origin, which is the leading edge of the airfoil. The meshes are generated analogously to those used for the NACA 0012 case. That is, a coarse, linear mesh containing 4000 elements was generated. Figure 3-15 shows this coarse mesh. The linear coarse mesh was uniformly refined twice to produce meshes of 16000 and 64000 elements. Then, high-order meshes were constructed using the linear elasticity node movement scheme. Results are presented for $p = 1, 2, 3, 4, 5$ for the coarse and medium meshes. For the fine mesh, only $p = 1, 2, 3$ solutions have been computed.

Figure 3-16 shows the drag error results obtained using the dual consistent discretization. The error is computed relative to an “exact” drag value, which is computed by extrapolating



(a) Drag error versus h for $p = 1$ through $p = 5$ ($p = 1 : \square$, $p = 2 : \circ$, $p = 3 : *$, $p = 4 : \triangleleft$, $p = 5 : \times$)

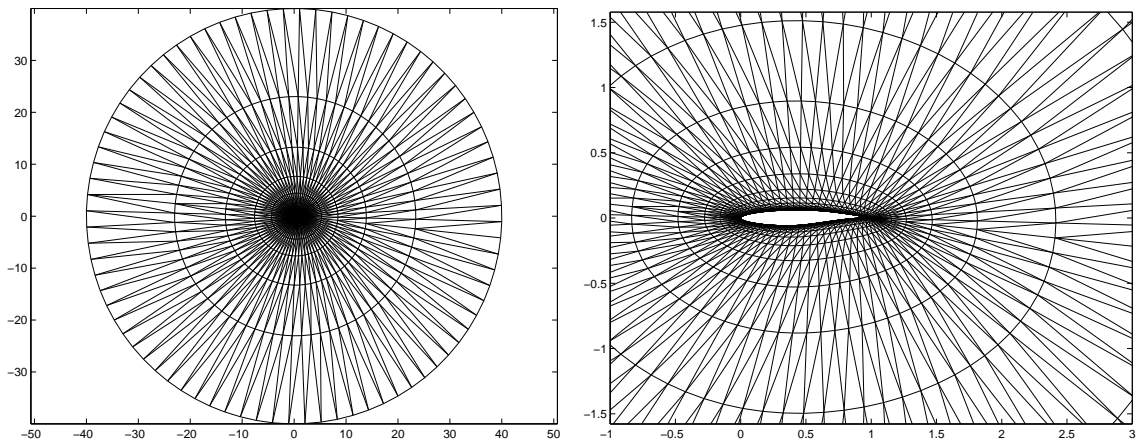


(b) Drag error versus DOF

Figure 3-14: Drag error for flow over a NACA 0012 computed using the standard weighting discretization

the $p = 5$ drag results, assuming that the error is proportional to h^3 . This rate is chosen to be consistent with the NACA 0012 case. The resulting value is 94.542604 counts.

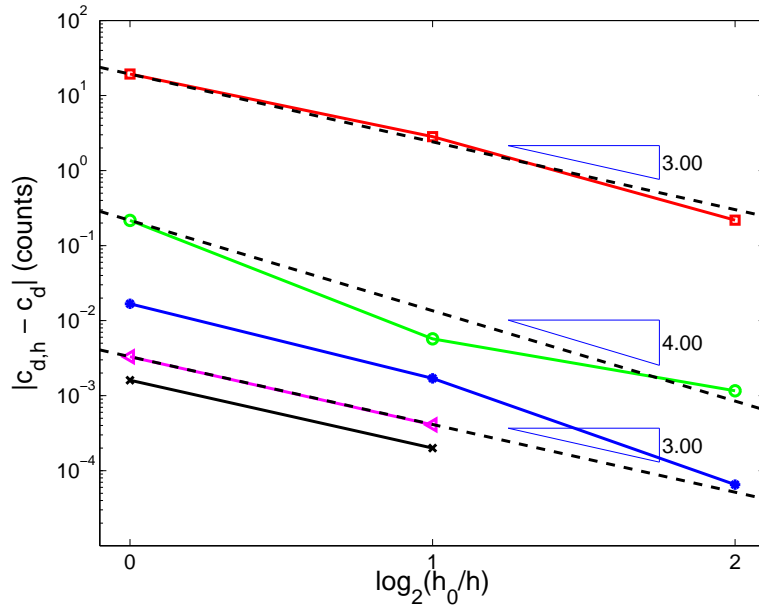
The figure shows the drag error versus grid refinement and DOF. The $p = 1$ order of accuracy is better than expected at approximately $O(h^3)$, and the $p = 2$ accuracy oscillates slightly about the optimal $O(h^4)$. However, as in the NACA 0012 case, the $p = 3, 4$ results do not achieve optimal accuracy. Unlike the NACA 0012 case, the suboptimal accuracy here is attributed to under-resolution of the trailing edge region. This under-resolution will be demonstrated by examining elemental error estimates, as shown in Section 4.3.3.



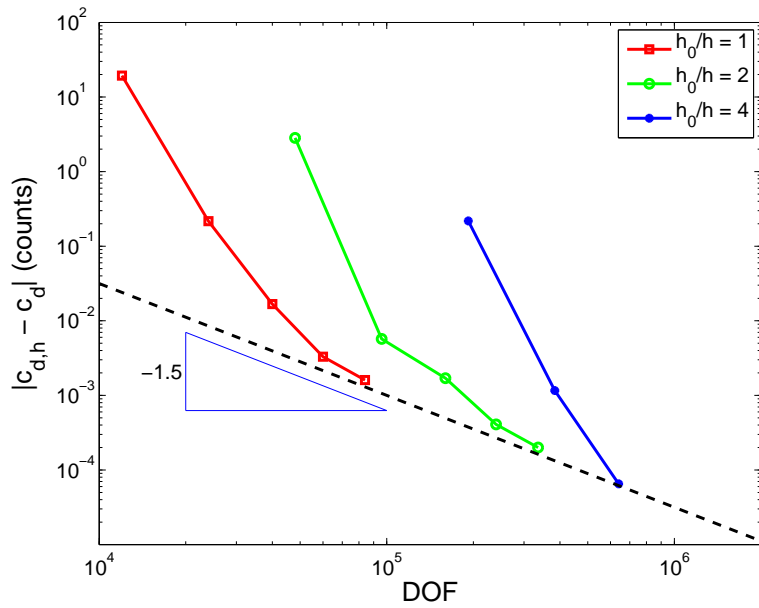
(a) Full domain

(b) Zoom

Figure 3-15: Coarse (4000 elements), linear mesh of the RAE 2822 airfoil



(a) Drag error versus h for $p = 1$ through $p = 5$ ($p = 1 : \square$, $p = 2 : \circ$, $p = 3 : *$, $p = 4 : \triangleleft$, $p = 5 : \times$)



(b) Drag error versus DOF

Figure 3-16: Drag error for flow over a RAE 2822 computed using the dual consistent discretization

Chapter 4

Output Error Estimation

This chapter details the error estimation approach applied in this work. Sections 4.1 and 4.2 show the error estimation algorithm, which is based on the method of Dual Weighted Residuals, due to Becker and Rannacher [17, 18], and draws heavily on recent work by Fidkowski and Darmofal [42, 39]. The main contribution of this chapter is the application of the algorithm to high-order discretizations of the RANS equations on curved, boundary conforming meshes. Numerical results, shown in Section 4.3, demonstrate that the error estimation algorithm produces acceptable estimates of the drag error for three high Re problems.

4.1 Motivation

Let $R_H(\cdot, \cdot)$ be a semi-linear form, and let $\mathbf{u}_H \in \mathcal{V}_H$ be such that

$$R_H(\mathbf{u}_H, \mathbf{v}_H) = 0, \quad \forall \mathbf{v}_H \in \mathcal{V}_H, \quad (4.1)$$

where \mathcal{V}_H is an appropriate finite dimensional function space—e.g. \mathcal{V}_h^p as defined in Section 2.3. Furthermore, let $\mathbf{u} \in \mathcal{V}$ be the exact solution of the PDE of interest. Then, for a general output of interest, $\mathcal{J}(\cdot)$, define the following dual problem: find $\boldsymbol{\psi} \in \mathcal{V}$ such that

$$\bar{R}_H(\mathbf{u}, \mathbf{u}_H; \mathbf{v}, \boldsymbol{\psi}) = \bar{\mathcal{J}}(\mathbf{u}, \mathbf{u}_H; \mathbf{v}), \quad \forall \mathbf{v} \in \mathcal{V}_H + \mathcal{V}, \quad (4.2)$$

where \bar{R}_H and $\bar{\mathcal{J}}$ are mean-value linearizations given by

$$\begin{aligned} \bar{R}_H(\mathbf{u}, \mathbf{u}_H; \mathbf{v}, \mathbf{w}) &\equiv \int_0^1 R'_H[\theta \mathbf{u} + (1 - \theta) \mathbf{u}_H](\mathbf{v}, \mathbf{w}) d\theta, \\ \bar{\mathcal{J}}(\mathbf{u}, \mathbf{u}_H; \mathbf{v}) &\equiv \int_0^1 \mathcal{J}'[\theta \mathbf{u} + (1 - \theta) \mathbf{u}_H](\mathbf{v}) d\theta. \end{aligned}$$

Then,

$$\begin{aligned}\bar{R}_H(\mathbf{u}, \mathbf{u}_H; \mathbf{u} - \mathbf{u}_H, \mathbf{w}) &= R_H(\mathbf{u}, \mathbf{w}) - R_H(\mathbf{u}_H, \mathbf{w}), \\ \bar{\mathcal{J}}(\mathbf{u}, \mathbf{u}_H; \mathbf{u} - \mathbf{u}_H) &= \mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{u}_H).\end{aligned}$$

Assuming that $R_H(\mathbf{u}, \mathbf{w}) = 0, \forall \mathbf{w} \in \mathcal{V}_H + \mathcal{V}$ and using (4.1) and (4.2), the error in the output may be written as

$$\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{u}_H) = -R_H(\mathbf{u}_H, \boldsymbol{\psi} - \boldsymbol{\psi}_H), \quad (4.3)$$

for arbitrary $\boldsymbol{\psi}_H \in \mathcal{V}_H$. Alternatively, by defining the adjoint residual,

$$\bar{R}_H^\psi(\mathbf{u}, \mathbf{u}_H; \mathbf{v}, \mathbf{w}) \equiv \bar{R}_H(\mathbf{u}, \mathbf{u}_H; \mathbf{v}, \mathbf{w}) - \bar{\mathcal{J}}(\mathbf{u}, \mathbf{u}_H; \mathbf{v}),$$

it is possible to express the error in terms of the exact primal solution. In particular, one can show that

$$\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{u}_H) = -\bar{R}_H^\psi(\mathbf{u}, \mathbf{u}_H; \mathbf{u} - \mathbf{u}_H, \boldsymbol{\psi}_H), \quad (4.4)$$

again for any $\boldsymbol{\psi}_H \in \mathcal{V}_H$.

4.2 Implementation

Given that (4.3) and (4.4) depend on the unknown exact solutions, it is necessary to make some approximations to compute error estimates. To begin, all mean-value linearizations are replaced by linearizations about the discrete solution, \mathbf{u}_H . Then, to minimize the error due to this approximation, $\boldsymbol{\psi}_H$ is set to the discrete adjoint solution [40, 42]. That is, $\boldsymbol{\psi}_H \in \mathcal{V}_H$ is chosen such that

$$R_H^\psi(\mathbf{u}_H; \mathbf{v}_H, \boldsymbol{\psi}_H) \equiv R'_H[\mathbf{u}_H](\mathbf{v}_H, \boldsymbol{\psi}_H) - \mathcal{J}'[\mathbf{u}_H](\mathbf{v}_H) = 0, \quad \forall \mathbf{v}_H \in \mathcal{V}_H.$$

Furthermore, the differences $\mathbf{u} - \mathbf{u}_H$ and $\boldsymbol{\psi} - \boldsymbol{\psi}_H$ are replaced by approximations $\mathbf{u}_h - \mathbf{u}_H$ and $\boldsymbol{\psi}_h - \boldsymbol{\psi}_H$, where \mathbf{u}_h and $\boldsymbol{\psi}_h$ are surrogates for \mathbf{u} and $\boldsymbol{\psi}$, that exist in a richer space, \mathcal{V}_h , than \mathbf{u}_H and $\boldsymbol{\psi}_H$. In this work, if \mathcal{V}_H is given by the space of polynomials of order p in reference space,

$$\mathcal{V}_H \equiv \{\mathbf{v} \in [L^2(\Omega)]^r \mid \mathbf{v} \circ f_\kappa \in [P^p(\kappa_{ref})]^r, \forall \kappa \in T_H\},$$

then \mathcal{V}_h is taken to be the space of polynomials of order $p + 1$. In particular,

$$\mathcal{V}_h \equiv \{\mathbf{v} \in [L^2(\Omega)]^r \mid \mathbf{v} \circ f_\kappa \in [P^{p+1}(\kappa_{ref})]^r, \forall \kappa \in T_H\}.$$

For linear geometry elements (i.e. f_κ is affine), one common approach is to define \mathbf{u}_h and ψ_h by a local L^2 or H^1 patch reconstruction [72, 42]. For example, the L^2 patch reconstruction takes the following form: for each element κ , find $\mathbf{v} \in P^{p+1}(\mathcal{P}(\kappa))$ such that

$$\mathbf{v} = \arg \min \|\mathbf{v} - \mathbf{u}_H\|_{L^2(\mathcal{P}(\kappa))},$$

where $\mathcal{P}(\kappa)$ denotes the union of κ and the elements that share a face with κ , as depicted in Figure 4-1. Then, the reconstruction on κ is defined by restricting \mathbf{v} to κ : $\mathbf{u}_h|_\kappa = \mathbf{v}|_\kappa$.

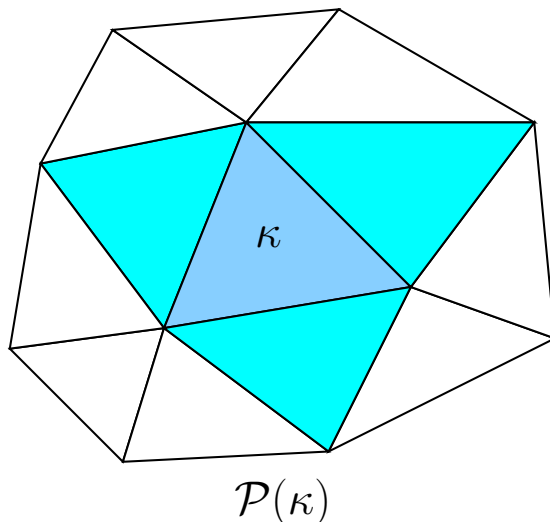


Figure 4-1: The patch of elements centered at κ , denoted $\mathcal{P}(\kappa)$, highlighted in blue

However, in the curved element case, since p th order polynomials in the reference space are generally not p th order polynomials in physical space, it is not likely that \mathbf{u}_h defined in this manner is in the space \mathcal{V}_h . Furthermore, the patch reconstruction fails to account for the physics of the problem. Specifically, the reconstruction does not properly account for the propagation of information in convection dominated problems. Thus, while it is possible to generalize the patch reconstruction procedure to generate a reconstruction in the desired space or to choose a different space \mathcal{V}_h , a different technique is applied here.

To motivate the procedure, one would expect that choosing \mathbf{u}_h and ψ_h to be the DG solutions of the primal and adjoint problems posed on the solution space \mathcal{V}_h would give accurate error estimates. However, for the purposes of error estimation, these solutions

are regarded as too expensive to compute. Thus, instead of computing these solutions, the exact solution surrogates are defined by injecting \mathbf{u}_H and $\boldsymbol{\psi}_H$ into \mathcal{V}_h and taking a small number of element-block Jacobi iterations on the $p + 1$ discrete problem. To be precise, the algorithm has the following steps:

1. Compute the injections of \mathbf{u}_H and $\boldsymbol{\psi}_H$ into \mathcal{V}_h . That is, find $U^0 \in \mathbb{R}^N$ and $\Psi^0 \in \mathbb{R}^N$ such that

$$\mathbf{u}_H(\mathbf{x}) = \sum_{i=1}^N U_i^0 \phi_i(\mathbf{x}), \quad \boldsymbol{\psi}_H(\mathbf{x}) = \sum_{i=1}^N \Psi_i^0 \phi_i(\mathbf{x}),$$

where $\{\phi_i\}$ for $i = 1, \dots, N$ forms a basis of \mathcal{V}_h .

2. Take K element-block Jacobi iterations on the primal problem. That is, for $k = 0, \dots, K - 1$,

$$U^{k+1} = U^k - \mathbf{P}^{-1}(U^k)R(U^k),$$

where $R(U^k)$ is the primal residual vector evaluated at U^k and $\mathbf{P}(U^k)$ is the element-block diagonal of the Jacobian matrix, $\frac{dR}{dU}$, evaluated at U^k .

3. Set $\mathbf{u}_h(\mathbf{x}) = \sum_{i=1}^N U_i^K \phi_i(\mathbf{x})$.

4. Take K element-block Jacobi iterations on the dual problem. That is, for $k = 0, \dots, K - 1$,

$$\Psi^{k+1} = \Psi^k - \mathbf{P}^{-T}(U^K)R^\psi(U^K; \Psi^k),$$

where $R^\psi(U^K; \Psi^k)$ is the residual vector for the dual problem taken about U^K and evaluated at Ψ^k .

5. Set $\boldsymbol{\psi}_h(\mathbf{x}) = \sum_{i=1}^N \Psi_i^K \phi_i(\mathbf{x})$.

Finally, given the surrogate solutions \mathbf{u}_h and $\boldsymbol{\psi}_h$, the error can be approximated using either the primal residual or the dual residual:

$$|\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{u}_H)| \approx \epsilon_{prim} \equiv |R_h(\mathbf{u}_H, \boldsymbol{\psi}_h - \boldsymbol{\psi}_H)|, \quad (4.5)$$

$$|\mathcal{J}(\mathbf{u}) - \mathcal{J}(\mathbf{u}_H)| \approx \epsilon_{dual} \equiv |R_h^\psi(\mathbf{u}_H; \mathbf{u}_h - \mathbf{u}_H, \boldsymbol{\psi}_H)|, \quad (4.6)$$

where R_h and R_h^ψ denote the primal and dual residual forms, respectively, on the space \mathcal{V}_h .

An elemental error indicator is required for the mesh adaptation procedure. This indicator is constructed by averaging the primal residual and dual residual approximations to the error contribution from the element. In particular, for each element κ ,

$$\epsilon_\kappa = \frac{1}{2} \left(|R_h(\mathbf{u}_H, (\boldsymbol{\psi}_h - \boldsymbol{\psi}_H)|_\kappa)| + |R_h^\psi(\mathbf{u}_H; (\mathbf{u}_h - \mathbf{u}_H)|_\kappa, \boldsymbol{\psi}_H)| \right). \quad (4.7)$$

Another error estimate can be constructed by summing the elemental contributions above:

$$\epsilon = \sum_{\kappa} \epsilon_{\kappa}.$$

Due to the absolute values, this estimate is generally larger than both ϵ_{prim} and ϵ_{dual} .

4.3 Numerical Results

In this section, the error estimation algorithm is applied to three high Re test cases. In all cases, uniform refinement studies have been conducted to enable comparison of the error estimates to the error computed relative to fine mesh or extrapolated results. Only results for the standard weighting and dual consistent discretizations are presented because the mixed formulation implementation has not been parallelized.

4.3.1 Flat Plate

The first test case is $Re_c = 1 \times 10^7$, $M_{\infty} = 0.25$ flow over a flat plate with zero pressure gradient. The coarse mesh in this study is a 1516 element mesh generated by mesh adaptation using the $p = 3$, dual consistent discretization. At each refinement level, the mesh spacing is halved, leading to a set of three nested meshes. These meshes—referred to as the coarse, medium, and fine meshes—contain 1516, 6064, and 24256 elements. The exact drag is taken to be that computed from the solution of the $p = 4$, dual consistent discretization on a uniform refinement of the fine mesh. This value is 28.579496056 counts.

Dual Consistent Discretization

Figure 4-2 shows the drag error for the dual consistent discretization. The discretization attains optimal order of accuracy for the drag for $p = 1, 2$. However, for $p = 3$, the order is suboptimal. The cause of this behavior is uncertain, but it may be linked to the singularity in the solution at the leading edge of the plate or to a lack of resolution at the boundary layer edge. Despite the observed rate, the accuracy of the $p = 3$ solution is very good. In particular, the error on the coarse mesh is approximately $1 \times 10^{-4}\%$ of the total drag. Figure 4-3 shows the error estimates ϵ_{prim} and ϵ_{dual} . The trends observed in the error estimates are very similar to those of the exact error. Moreover, taking the maximum of the primal and dual estimates gives a slightly conservative but still accurate error estimate. Figure 4-4 shows this maximum estimate, as well as its effectivity, defined as the ratio of the maximum estimate to the actual error. The effectivity in this case is in the range (1.5, 4.6).

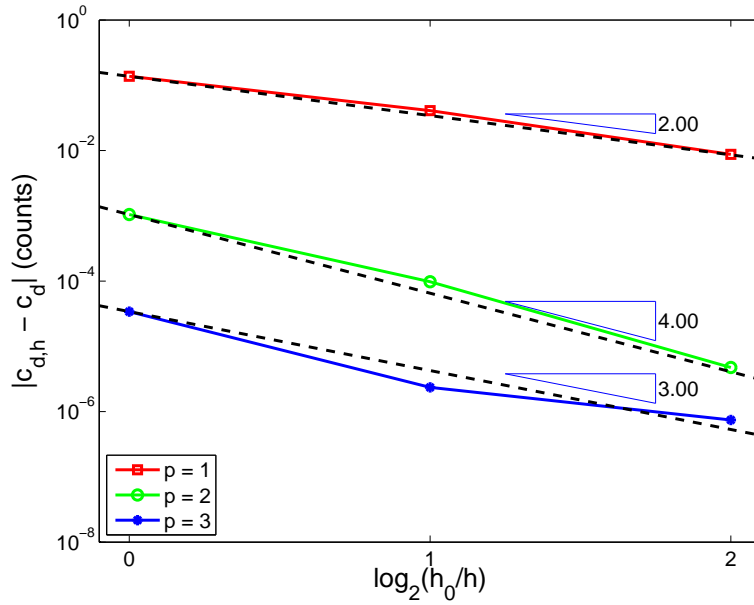


Figure 4-2: Drag error for the dual consistent discretization versus uniform grid refinement for the flat plate test case

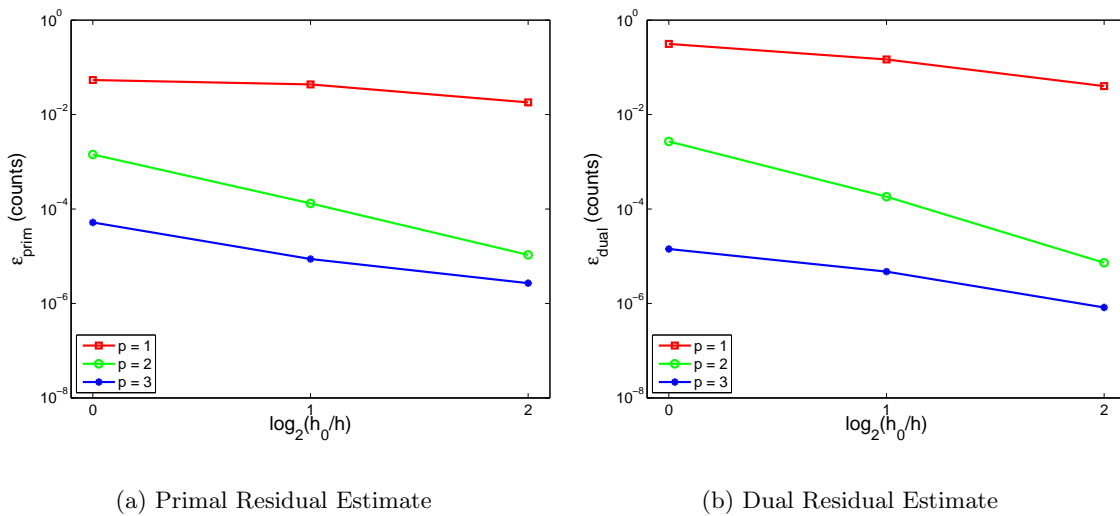


Figure 4-3: Primal and dual residual drag error estimates for the dual consistent discretization versus uniform grid refinement for the flat plate test case

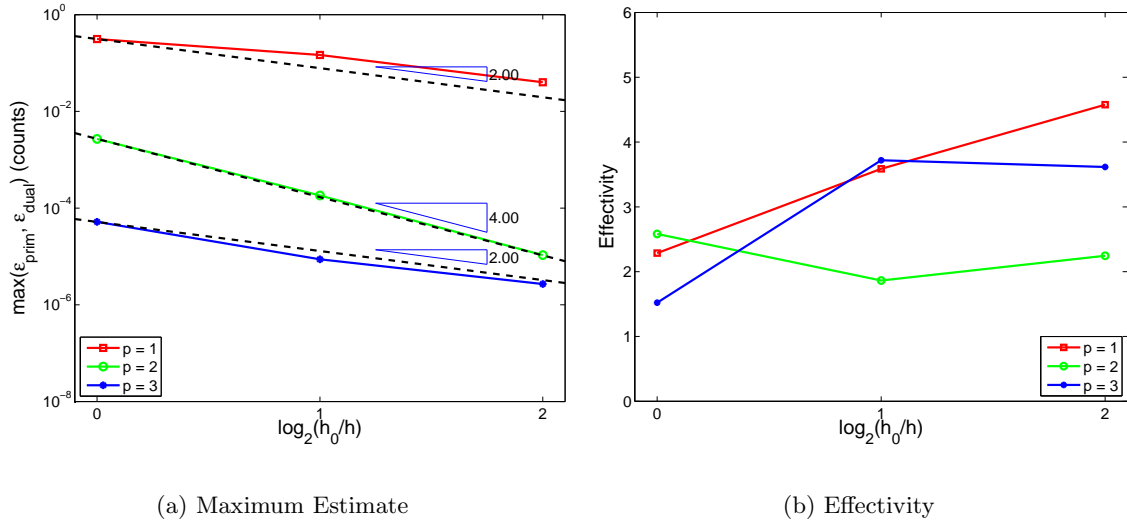


Figure 4-4: Maximum error estimate and effectivity for the dual consistent discretization versus uniform grid refinement for the flat plate test case

Standard Weighting Discretization

Figure 4-5 shows the drag error for the standard weighting discretization. The discretization attains optimal order of accuracy for the drag for $p = 1$. For $p = 2$, the discretization is suboptimal. This behavior is expected and agrees with the results observed for the model problem in Chapter 3. For $p = 3$, the order is also suboptimal. As with the $p = 3$, dual consistent discretization, the cause of this behavior is unclear. Figure 4-6 shows the error estimates ϵ_{prim} and ϵ_{dual} . Unlike the dual consistent discretization, the dual inconsistent scheme maximum error estimates do not provide a conservative estimate for all p . In particular, as shown in Figure 4-7, the effectivity for $p = 2$ is less than one for each grid, and it decreases with grid refinement.

4.3.2 NACA 0012

The second test case is $Re_c = 1 \times 10^7$, $M_\infty = 0.25$, $\alpha = 0$ flow over a NACA 0012 airfoil. The computational domain and meshes for this case are as described in Section 3.4.2. The “exact” drag error results for this case are also shown in Section 3.4.2. Thus, only the error estimates and effectivities are shown here.

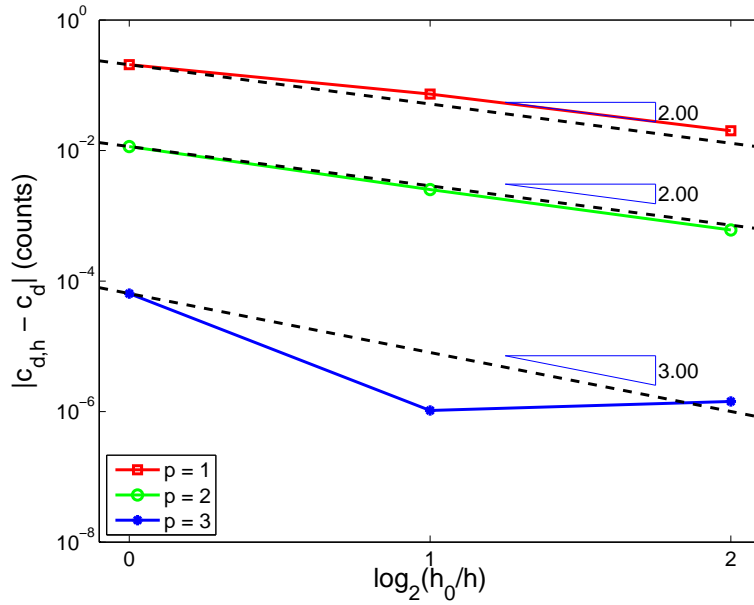
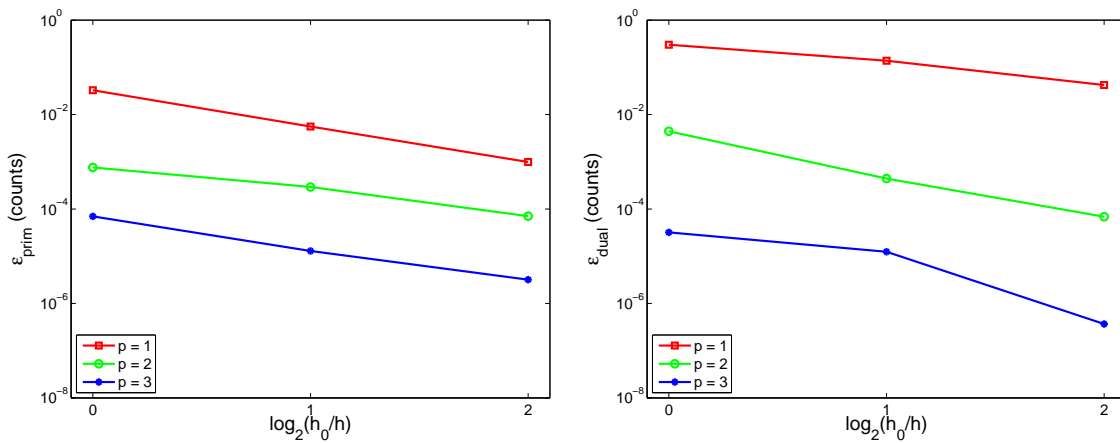


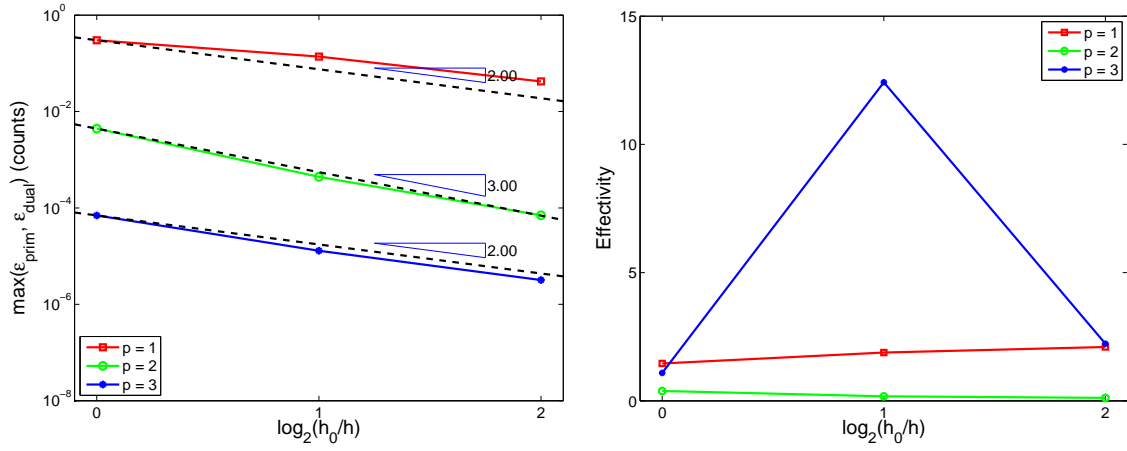
Figure 4-5: Drag error for the standard weighting discretization versus uniform grid refinement for the flat plate test case



(a) Primal Residual Estimate

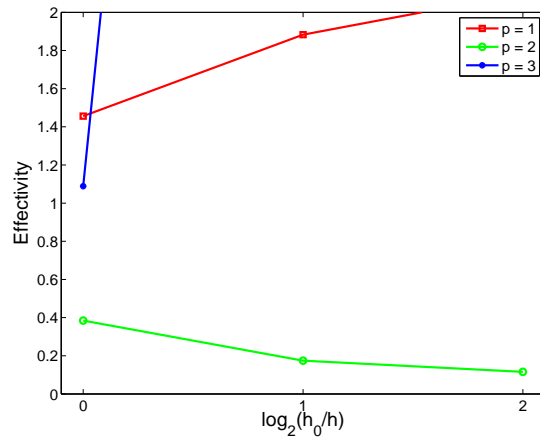
(b) Dual Residual Estimate

Figure 4-6: Primal and dual residual drag error estimates for the standard weighting discretization versus uniform grid refinement for the flat plate test case



(a) Maximum Estimate

(b) Effectivity



(c) Effectivity (zoom)

Figure 4-7: Maximum error estimate and effectivity for the standard weighting discretization versus uniform grid refinement for the flat plate test case

Dual Consistent Discretization

Figure 4-8 shows the primal and dual error estimates. Unlike the flat plate results, the two estimates show somewhat different behavior. For example, on the second refinement, the $p = 2$ primal estimate decreases dramatically while the dual estimate is relatively constant. However, as Figure 4-9 shows, the maximum of the two error estimates generally provides a

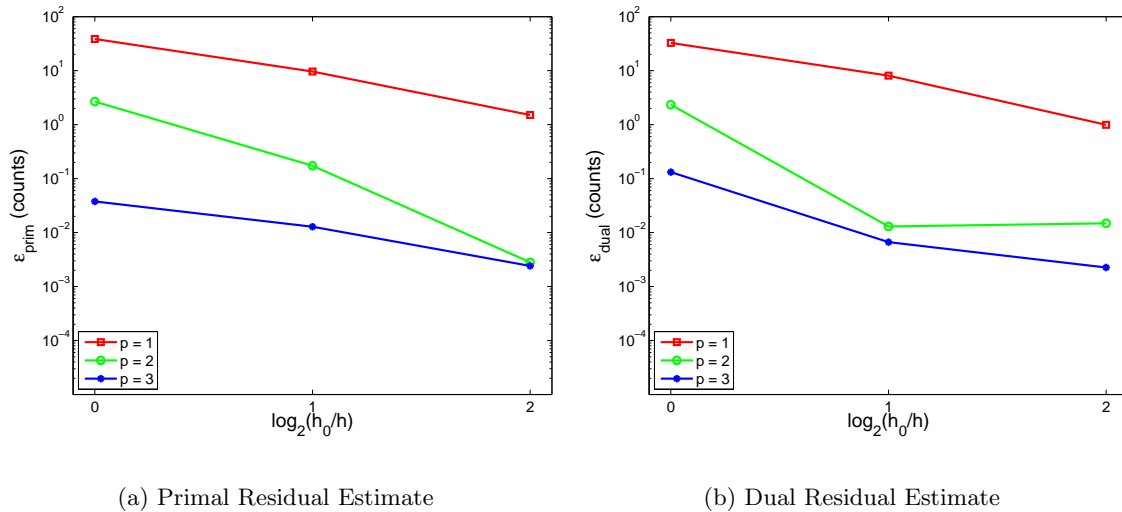


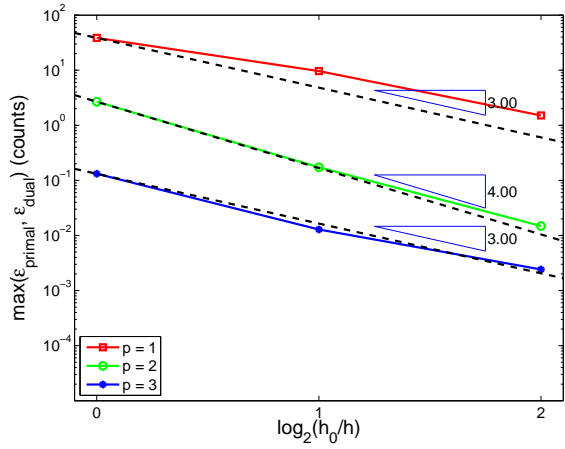
Figure 4-8: Primal and dual residual drag error estimates for the dual consistent discretization versus uniform grid refinement for the NACA 0012 test case

conservative but reasonably accurate estimate. In particular, the effectivity of the maximum estimate is between 0.8 and 2.1 for all cases except $p = 2$ on the medium mesh.

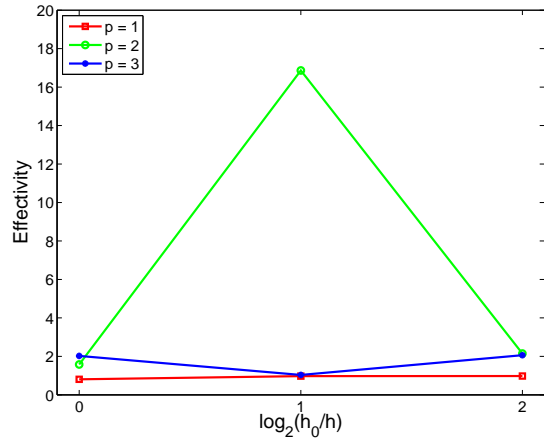
In Section 3.4.2, it was observed that the order of accuracy for $p \geq 3$ for this case is lower than that expected for smooth solutions. This behavior was attributed to under-resolution at the boundary layer edge. To examine this claim in more detail, Figures 4-10 and 4-11 show the elemental error estimate, ϵ_K , and Mach number for $p = 2$ and $p = 4$, respectively. The figure demonstrates that, in the $p = 2$ case, while the error estimate is non-zero at the boundary layer edge, the errors are largest in the flow upstream of the airfoil. Alternatively, in the $p = 4$ case, the error estimate is largest near the boundary layer edge.

Standard Weighting Discretization

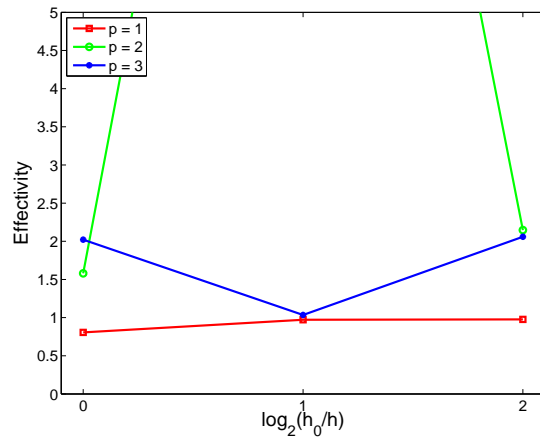
Figure 4-12 shows the primal and dual error estimates. The results are qualitatively similar to those for the dual consistent discretization. The maximum error estimate and effectivity are shown in Figure 4-13. As with the dual consistent discretization, the maximum error estimate generally provides an accurate error estimate for this case. However, for $p = 2$



(a) Maximum Estimate



(b) Effectivity



(c) Effectivity (zoom)

Figure 4-9: Maximum error estimate and effectivity for the dual consistent discretization versus uniform grid refinement for the NACA 0012 test case

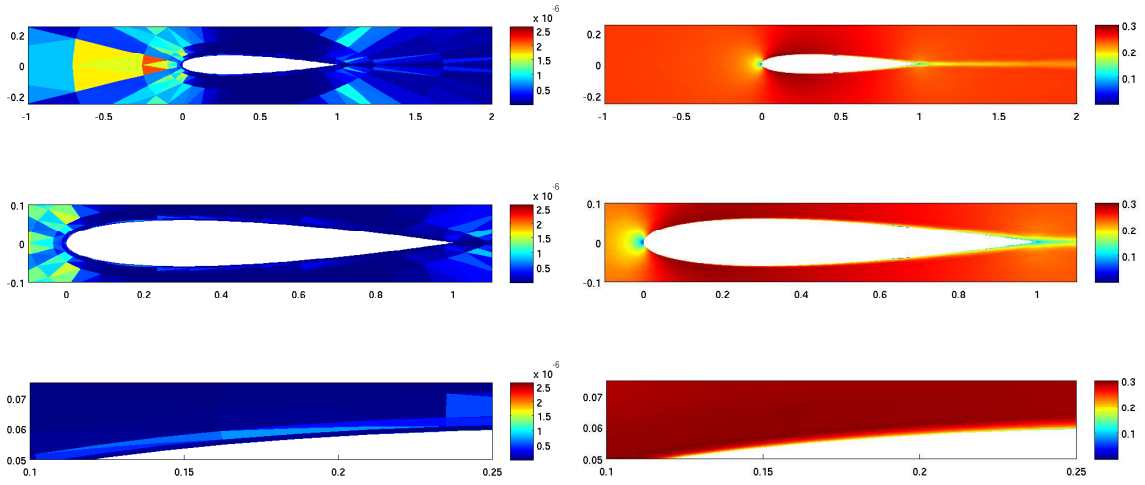


Figure 4-10: Elemental error estimate (left) and Mach number (right) for the $p = 2$, dual consistent discretization on the coarse mesh for the NACA 0012 test case

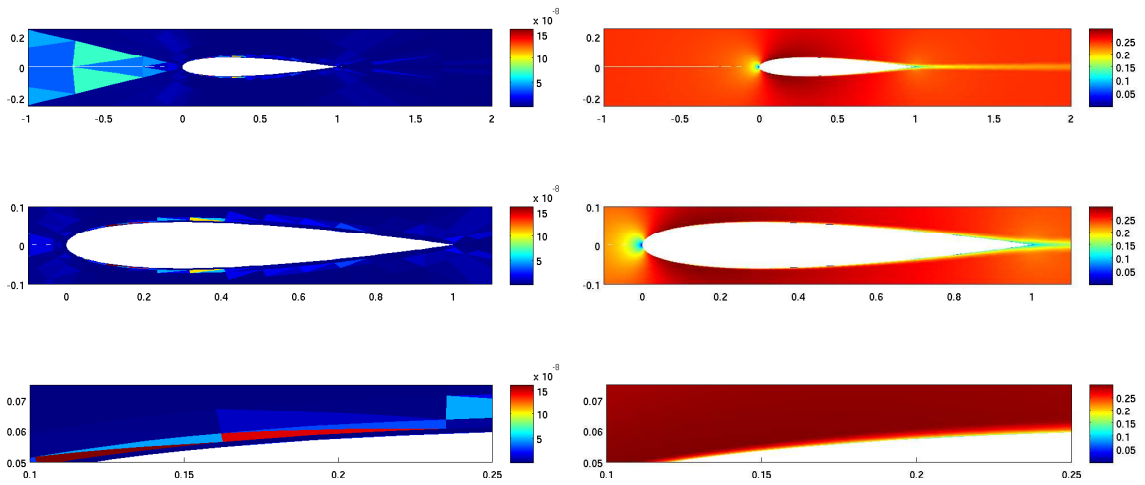


Figure 4-11: Elemental error estimate (left) and Mach number (right) for the $p = 4$, dual consistent discretization on the coarse mesh for the NACA 0012 test case

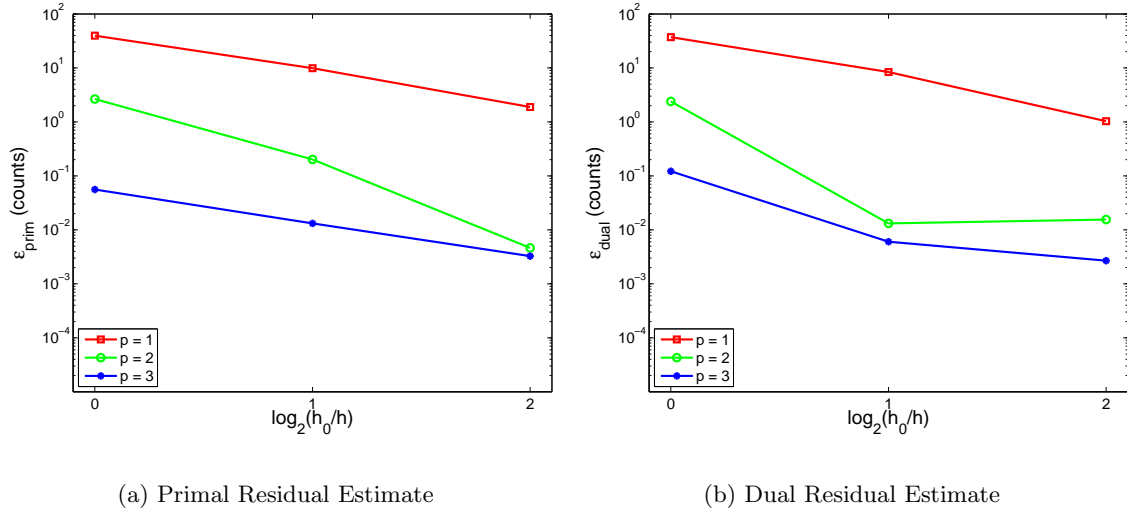


Figure 4-12: Primal and dual residual drag error estimates for the standard weighting discretization versus uniform grid refinement for the NACA 0012 test case

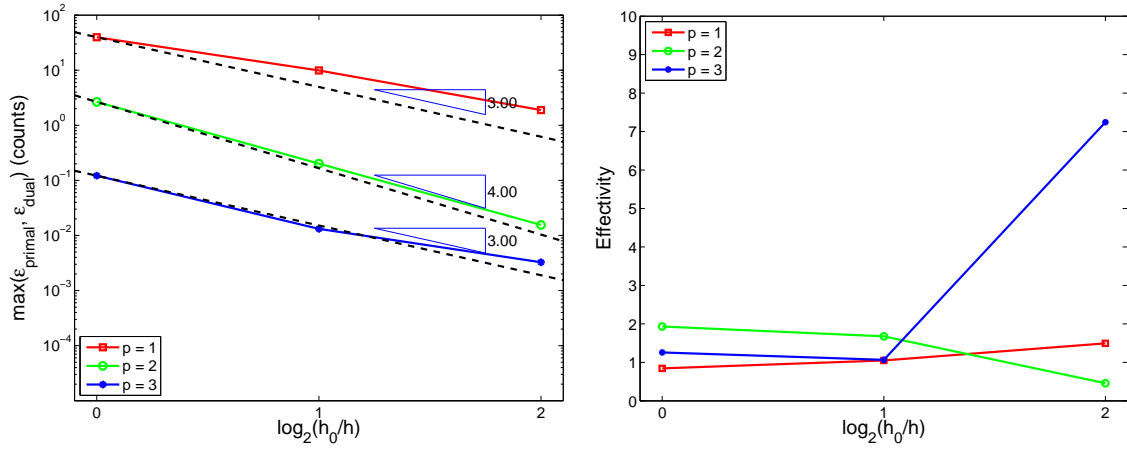
the error estimate is converging more rapidly than the actual error, leading to a decreasing effectivity. Thus, on the fine mesh, the $p = 2$ effectivity is approximately 0.5. The only other case with an effectivity outside the range of (0.8, 2) is the $p = 3$ fine grid result, where the effectivity is almost seven.

4.3.3 RAE 2822

The final test case is $Re_c = 6.3 \times 10^6$, $M_\infty = 0.6$, $\alpha = 2.57^\circ$ flow over a RAE 2822 airfoil. The computational domain, meshes, and “exact” drag for this case are given in Section 3.4.3, which also contains the “exact” drag error results. Thus, only the error estimates and effectivities are shown here. Furthermore, only the dual consistent discretization is used.

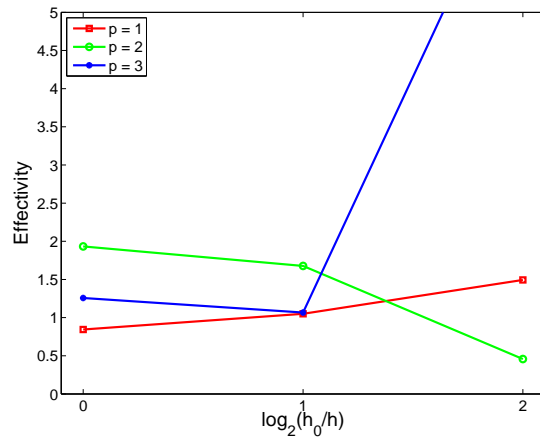
Figure 4-14 shows the primal and dual error estimates. The primal error estimate for $p = 1$ on the coarse mesh is so large (3.5×10^7 counts) that it is not shown in the figure. This large error estimate results from the fact that the element-block Jacobi iterative procedure used to compute the surrogate primal solution diverges for this case. However, for the other cases, where the element-block Jacobi solve is well-behaved, the error estimates are reasonable. The maximum error estimate and its effectivity are shown in Figure 4-15. The effectivity is in the range (0.9, 11.0) for all but the $p = 1$ result on the coarse mesh.

As noted in Section 3.4.3, the order of accuracy for $p \geq 3$ for this case is below that expected for smooth flow. Figure 4-16 shows the $p = 3$ elemental error estimates on the coarse and fine meshes. At first glance, it appears that the error is small everywhere near



(a) Maximum Estimate

(b) Effectivity



(c) Effectivity (zoom)

Figure 4-13: Maximum error estimate and effectivity for the standard weighting discretization versus uniform grid refinement for the NACA 0012 test case

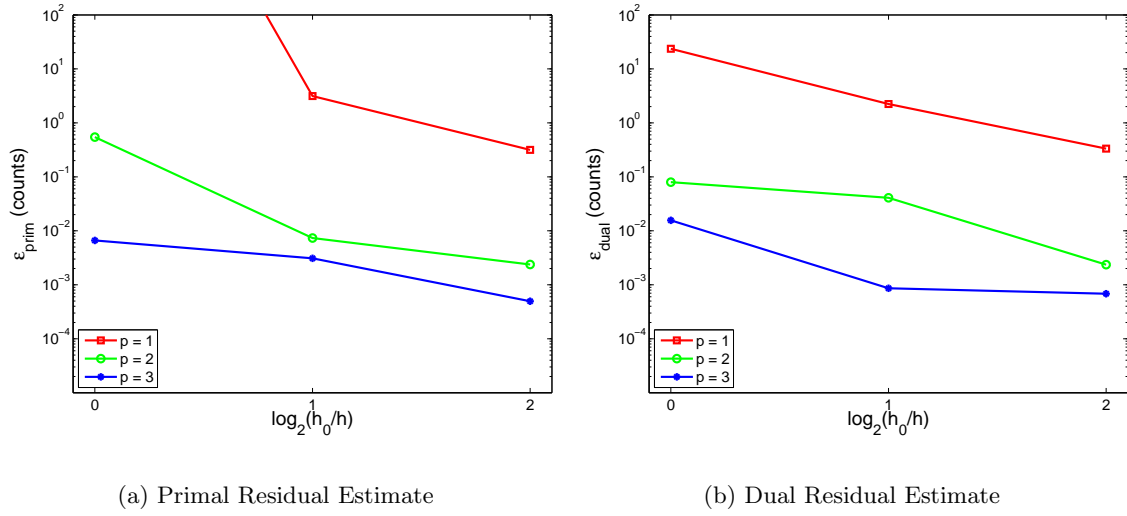


Figure 4-14: Primal and dual residual drag error estimates for the dual consistent discretization versus uniform grid refinement for the RAE 2822 test case

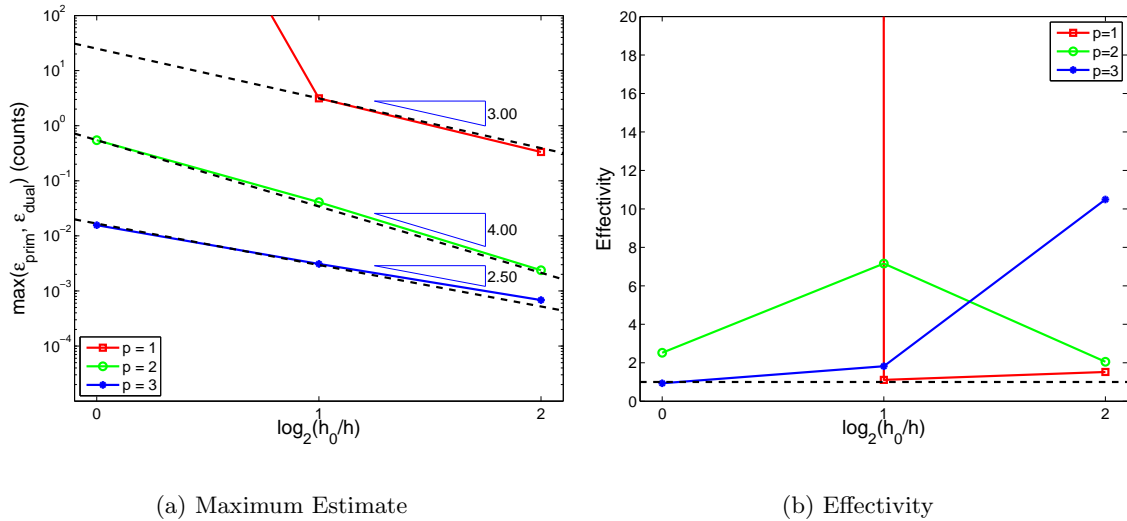


Figure 4-15: Maximum error estimate and effectivity for the dual consistent discretization versus uniform grid refinement for the RAE 2822 test case

the airfoil. However, close inspection of the trailing edge region, as shown for the fine mesh in Figure 4-17, shows that the elemental errors in this region are large. These errors indicate a lack of resolution in this region. Thus, one can conclude that under-resolution of the trailing edge singularity is leading to a loss of accuracy for $p \geq 3$ for this case.

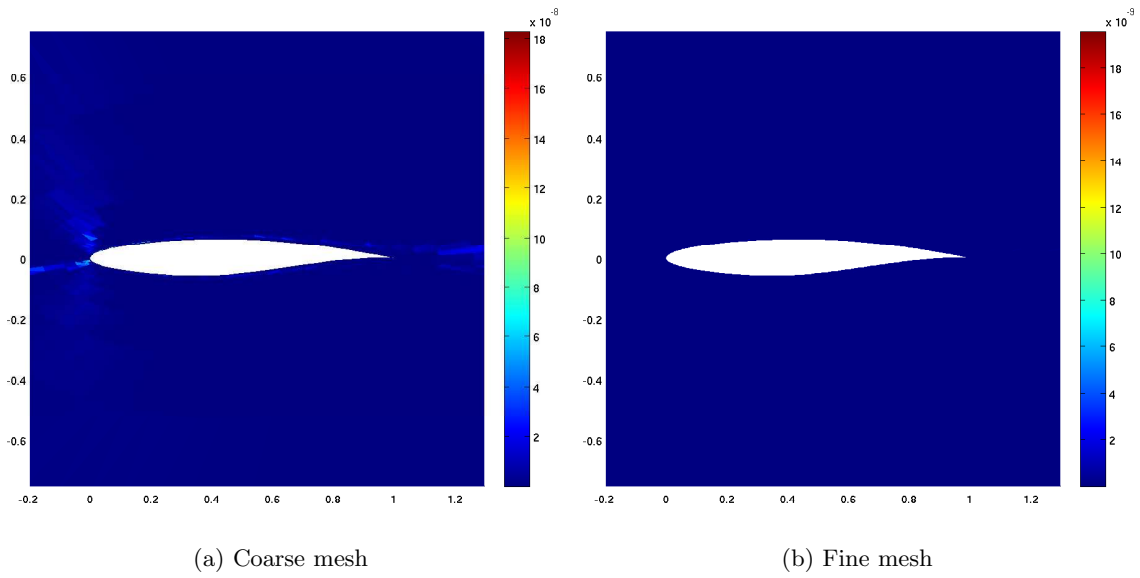


Figure 4-16: Elemental error estimate for the $p = 3$, dual consistent discretization on the coarse and fine meshes for the RAE 2822 test case

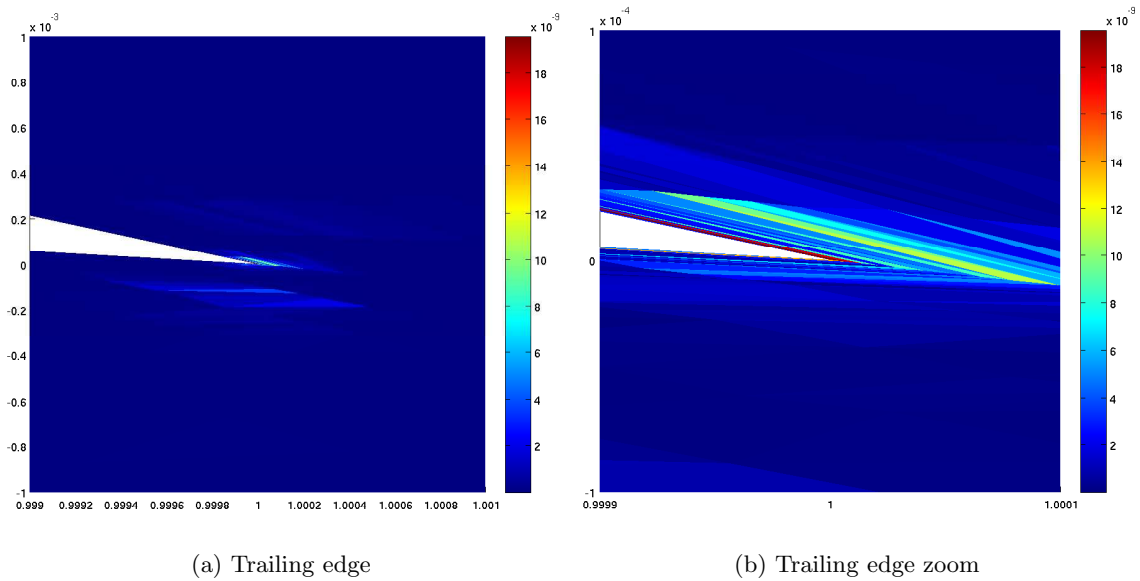


Figure 4-17: Elemental error estimate at the trailing edge for the $p = 3$, dual consistent discretization on the fine mesh for the RAE 2822 test case

Chapter 5

Output-Based Adaptation

In this chapter, the output error estimate is used to drive a mesh adaptation algorithm. The algorithm combines the output error estimate with interpolation error estimates to determine a desired metric for the new mesh. This technique, described in Section 5.1, closely follows those of Venditti and Darmofal [102, 103] and Fidkowski and Darmofal [42, 39]. Thus, the main contributions of this chapter are the application of the algorithm to high-order discretizations of the RANS equations and the use of curved, boundary conforming meshes. Two techniques for generating curved meshes are detailed in Section 5.1.3. Section 5.2 provides numerical results.

5.1 Algorithm

The adaptation algorithm used here takes the following form:

1. Generate an initial mesh.
2. Solve the steady state problem, $\mathbf{R}(\mathbf{U}) = 0$, where \mathbf{R} denotes the residual vector and \mathbf{U} denotes the discrete solution. Specifically,
 - (a) Set the solution to some initial guess \mathbf{U}^0 . Initialize the CFL number, and compute the corresponding time step, Δt . Set $i = 1$.
 - (b) While $\|\mathbf{R}(\mathbf{U}^{i-1})\| > \delta_{tol}$,
 - i. Update the primal state: $\mathbf{U}^i = \mathbf{U}^{i-1} - \left(\frac{1}{\Delta t} \mathcal{M} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}^{i-1}} \right)^{-1} \mathbf{R}(\mathbf{U}^{i-1})$.
 - ii. Update Δt according to Section 2.4.
 - iii. $i \rightarrow i + 1$.

where δ_{tol} is the solution tolerance, \mathcal{M} is the mass matrix.

3. Solve the adjoint problem: $\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}}^T \Psi = \frac{\partial J}{\partial \mathbf{U}} \Big|_{\mathbf{U}}^T$, where J is the discrete output of interest.
4. Using \mathbf{U} and Ψ , compute ϵ , the output error estimate, as described in Chapter 4.
5. If $\epsilon < \epsilon_{tol}$, where ϵ_{tol} is the requested error tolerance, quit. Otherwise, adapt the mesh, and return to step 2.

Thus, at each adaptation iteration, an error estimate is computed, and, assuming that estimate is greater than the desired tolerance, a new computational mesh is generated. The desired mesh is specified by means of an $n \times n$ Riemannian metric, \mathbf{M} , defined over the physical domain, where n is the dimension of the domain. The metric contains information on the desired mesh spacing in physical space. In particular, under the metric, the mesh spacing is required to be unity in all directions. To be precise, let $\hat{\mathbf{t}}$ be a unit vector, and let $h_t(\mathbf{x})$ be the desired mesh spacing in the $\hat{\mathbf{t}}$ direction at \mathbf{x} . Then, the metric at \mathbf{x} is defined by requiring

$$h_t^2 \hat{\mathbf{t}}^T \mathbf{M} \hat{\mathbf{t}} = 1, \quad (5.1)$$

for all directions $\hat{\mathbf{t}}$. This requirement defines an ellipsoid in physical space. The eigenvectors of \mathbf{M} , \mathbf{e}_i for $i = 1, \dots, n$, give the directions of the principal axes of an ellipsoid, and the eigenvalues of \mathbf{M} , λ_i for $i = 1, \dots, n$ are related to the lengths of the ellipsoid axes by $h_i^2 = 1/\lambda_i$.

The mesh metric calculation involves two parts: anisotropy detection and mesh optimization.

5.1.1 Anisotropy Detection

The desired anisotropy calculation is motivated by a Hessian-based analysis [21, 103]. The analysis begins with an interpolation error estimate. If u_H is a linear interpolant of a smooth scalar function u , assuming the interpolation error is zero at the segment endpoints, the interpolation error along a segment $\delta \mathbf{x}$ is bounded by

$$\max_{\mathbf{x} \in \delta \mathbf{x}} |u(\mathbf{x}) - u_H(\mathbf{x})| \leq C_1 h_{\delta \mathbf{x}}^2 \max_{\mathbf{x} \in \delta \mathbf{x}} \left| u_{\delta \mathbf{x}}^{(2)}(\mathbf{x}) \right|,$$

where $h_{\delta \mathbf{x}}$ is the length of the segment, C_1 is a constant that is independent of u , and $u_{\delta \mathbf{x}}^{(2)}$ is the second derivative of u in the $\delta \mathbf{x}$ direction [96]. Using the Hessian matrix, \mathbf{H} , where

$$H_{ij} = \frac{\partial^2 u}{\partial x_i \partial x_j},$$

the interpolation error bound can be rewritten as

$$\max_{\mathbf{x} \in \delta \mathbf{x}} |u(\mathbf{x}) - u_H(\mathbf{x})| \leq C_1 \max_{\mathbf{x} \in \delta \mathbf{x}} (\delta \mathbf{x}^T |\mathbf{H}(\mathbf{x})| \delta \mathbf{x}),$$

where $|\mathbf{H}| = \mathbf{V}|\mathbf{\Lambda}|\mathbf{V}^{-1}$ for $\mathbf{H} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$.

The desired mesh is designed to give equal interpolation errors in all directions. Thus, for some constant C , the requested mesh spacing should satisfy

$$h_t^2 \hat{\mathbf{t}}^T |\mathbf{H}| \hat{\mathbf{t}} = \frac{1}{C}, \quad (5.2)$$

for all unit vectors $\hat{\mathbf{t}}$. Using (5.1), (5.2) can be satisfied by requiring that

$$\mathbf{M} = C|\mathbf{H}|. \quad (5.3)$$

This relationship determines the metric up to a multiplicative constant which is determined by the mesh optimization step. Furthermore, given that the exact Hessian matrix is not known it must be approximated to compute the desired anisotropy.

Following Fidkowski [42, 39], this analysis can be extended to $p > 1$ interpolation. As with the $p = 1$ case, the analysis begins with an interpolation error bound:

$$\max_{\mathbf{x} \in \delta \mathbf{x}} |u(\mathbf{x}) - u_H(\mathbf{x})| \leq C_2 h_{\delta \mathbf{x}}^{p+1} \max_{\mathbf{x} \in \delta \mathbf{x}} \left| u_{\delta \mathbf{x}}^{(p+1)}(\mathbf{x}) \right|, \quad (5.4)$$

where u_H is a p th order interpolant of u that interpolates u exactly at the endpoints of the segment $\delta \mathbf{x}$. For $p > 1$, there is no analog of the Hessian matrix that can be used to bound the interpolation error and define the mesh metric. Instead, the principal stretching directions are explicitly computed. Specifically, define \mathbf{e}_1 to be the direction of the maximum $(p+1)$ derivative. Let \mathbf{e}_2 be the direction of the maximum $(p+1)$ derivative in the plane orthogonal to \mathbf{e}_1 , and so on. Then, the desired anisotropy is determined by requiring equal interpolation error in the directions given by \mathbf{e}_i . Thus, for $i, j = 1, \dots, n$,

$$h_i^{p+1} u_{\mathbf{e}_i}^{(p+1)} = C \Rightarrow \frac{h_i}{h_j} = \left(\frac{u_{\mathbf{e}_j}^{(p+1)}}{u_{\mathbf{e}_i}^{(p+1)}} \right)^{1/(p+1)}.$$

As with (5.3), this relationship determines the metric up to a multiplicative constant. This constant is based on the error estimate, as shown in Section 5.1.2. Furthermore, as with the exact Hessian, the exact $(p+1)$ derivatives are not known. Thus, they must be estimated to compute the desired anisotropy.

To apply this anisotropy detection method on curved meshes, a minor modification

is required. Specifically, given that the numerical solution is a p th order polynomial in reference space but not physical space, the interpolation error estimate (5.4) does not hold. However, an analogous bound holds in reference space. In particular, let

$$u(\mathbf{x}) = (u \circ f)(\boldsymbol{\xi}) = \tilde{u}(\boldsymbol{\xi}),$$

where f is the map from reference to physical space and $\boldsymbol{\xi}$ denotes the position in the reference space. Then, if \tilde{u}_H is a p th order interpolant of \tilde{u} ,

$$\max_{\boldsymbol{\xi} \in \delta\boldsymbol{\xi}} |\tilde{u}(\boldsymbol{\xi}) - \tilde{u}_H(\boldsymbol{\xi})| \leq C_2 h_{\delta\boldsymbol{\xi}}^{p+1} \max_{\boldsymbol{\xi} \in \delta\boldsymbol{\xi}} \left| \tilde{u}_{\delta\boldsymbol{\xi}}^{(p+1)}(\boldsymbol{\xi}) \right|,$$

where $\delta\boldsymbol{\xi}$ is a segment in reference space. Thus, following the technique above, one can define the principal stretching directions in reference space, $\tilde{\mathbf{e}}_i$. Then, requiring equal interpolation errors in the principal directions gives

$$\tilde{h}_i^{p+1} \tilde{u}_{\tilde{\mathbf{e}}_i}^{(p+1)} = \text{constant} \Rightarrow \frac{\tilde{h}_i}{\tilde{h}_j} = \left(\frac{\tilde{u}_{\tilde{\mathbf{e}}_j}^{(p+1)}}{\tilde{u}_{\tilde{\mathbf{e}}_i}^{(p+1)}} \right)^{1/(p+1)},$$

where \tilde{h}_i are the desired mesh spacings in reference space. This relationship determines the metric in reference space to within a multiplicative constant.

To define the metric in physical space, the mapping from the reference to the physical space is used. In particular, given the metric in reference space, $\tilde{\mathbf{M}}$, at $\boldsymbol{\xi}_0$, the points of the corresponding ellipsoid, $\boldsymbol{\xi}_e$, satisfy

$$(\boldsymbol{\xi}_e - \boldsymbol{\xi}_0)^T \tilde{\mathbf{M}}(\boldsymbol{\xi}_0) (\boldsymbol{\xi}_e - \boldsymbol{\xi}_0) = 1.$$

The image of this ellipsoid under the map f is given by $f(\boldsymbol{\xi}_e)$. Certainly, since f may be nonlinear, this image may not be an ellipsoid. However, linearizing f about $\boldsymbol{\xi}_0$ gives

$$f(\boldsymbol{\xi}_e) \approx \mathbf{x}_e = f(\boldsymbol{\xi}_0) + Df(\boldsymbol{\xi}_0)(\boldsymbol{\xi}_e - \boldsymbol{\xi}_0).$$

Using this linearized form, the points \mathbf{x}_e define an ellipsoid centered at $\mathbf{x}_0 = f(\boldsymbol{\xi}_0)$. This ellipsoid is used to define the metric in physical space at the point \mathbf{x}_0 . To illustrate this

calculation, let

$$\tilde{\mathbf{H}} = \tilde{\mathbf{V}}\tilde{\mathbf{\Lambda}}\tilde{\mathbf{V}}^{-1} = \left[\begin{array}{c|c|c} \tilde{\mathbf{e}}_1 & \cdots & \tilde{\mathbf{e}}_n \end{array} \right] \left[\begin{array}{ccc} \tilde{h}_1 & & \\ & \ddots & \\ & & \tilde{h}_n \end{array} \right] \left[\begin{array}{c} \tilde{\mathbf{e}}_1^T \\ \vdots \\ \tilde{\mathbf{e}}_n^T \end{array} \right].$$

If $\hat{\mathbf{t}}$ is a unit vector, then

$$\boldsymbol{\xi}_e - \boldsymbol{\xi}_0 = \tilde{\mathbf{H}}\hat{\mathbf{t}},$$

gives the point $\boldsymbol{\xi}_e$ on the ellipsoid defined by the metric in reference space. Then,

$$\mathbf{x}_e - f(\boldsymbol{\xi}_0) = Df(\boldsymbol{\xi}_0)\tilde{\mathbf{H}}\hat{\mathbf{t}},$$

gives the ellipsoid defining the metric in physical space. Thus, the directions and lengths of the principal axes of this ellipsoid are given by the left singular vectors and singular values of $Df(\boldsymbol{\xi}_0)\tilde{\mathbf{H}}$. This process is illustrated in two dimensions in Figure 5-1.

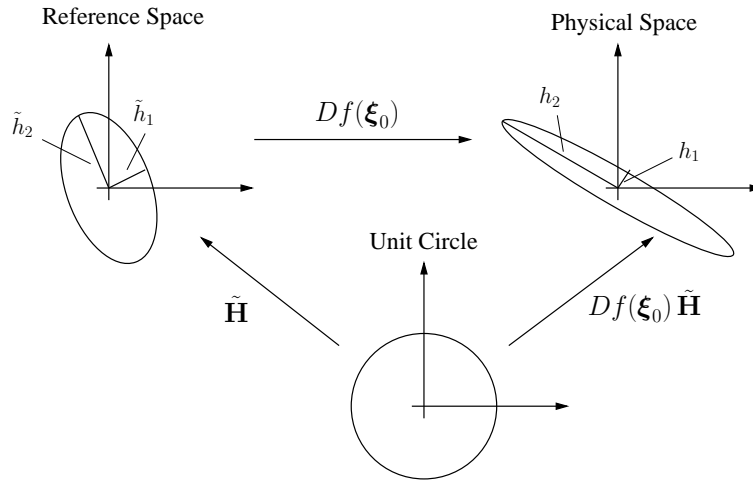


Figure 5-1: Illustration of transfer of desired mesh anisotropy from reference to physical space for two dimensional case

As noted earlier, the exact $(p + 1)$ derivatives are not known. Moreover, given that the discrete solution \mathbf{u}_H is order p , it contains no information on the $(p + 1)$ derivatives. Thus, in this work, the $(p + 1)$ derivatives are estimated using the surrogate solution, \mathbf{u}_h , which is computed for the error estimate. This function is a $(p + 1)$ th order polynomial in the reference element. Thus, the $(p + 1)$ derivatives are constant over the reference element, which implies that the desired anisotropy in the reference element is also constant. To transfer the anisotropy request to physical space, the Jacobian, Df , is required. In general,

this Jacobian varies over the reference element. In this work, the value at the centroid of the reference element is used to calculate the desired anisotropy in the physical space.

5.1.2 Mesh Optimization

As noted in Section 5.1.1, the anisotropy detection step only determines the desired mesh metric to within a multiplicative constant. That constant is determined by the mesh optimization step. Mesh optimization refers to determining which elements to refine/coarsen and the appropriate level of refinement/coarsening. In this work, the method of Fidkowski [42, 39] is used.

The main idea of the method is to attempt to equidistribute the error over the desired mesh. To make this idea concrete, let e_0 be the desired global error. This desired error is given by

$$e_0 = \max(\eta_a \epsilon, \eta_t \epsilon_{tol}),$$

where $\epsilon = \sum_{\kappa} \epsilon_{\kappa}$ is the conservative error estimate for the solution on the current mesh, ϵ_{tol} is the error tolerance, and $0 < \eta_a < 1$ and $0 < \eta_t \leq 1$ are the adaptation aggressiveness and target error fraction, respectively. Then, letting N_f be the estimated total number of elements in the desired mesh, e_0/N_f is the allowable error per element in the desired mesh.

Furthermore, let n_{κ} be the number of elements in the desired mesh contained in the element κ in the current mesh. Then, n_{κ} can be approximated by

$$n_{\kappa} = \prod_{i=1}^n \frac{h_i^c}{h_i}, \quad (5.5)$$

where h_i denotes the desired mesh spacings and h_i^c denotes the current mesh spacings. The current mesh spacings are computed from the current grid-implied metric. To define the grid-implied metric, let ℓ be the affine map from the reference element to the unit equilateral triangle/tetrahedron. This map, together with the map from the reference element to physical space, is illustrated in Figure 5-2. The grid-implied metric is defined by the left singular vectors and singular values of the product $D\ell^{-1}Df_{\kappa}$, where Df_{κ} is evaluated at the centroid of the reference element. This definition is an extension of that used by Fidkowski [42, 39] to the curved element case.

Given n_{κ} , one can write the allowable error for the elements in the desired mesh contained in the element κ of the current mesh as $n_{\kappa}e_0/N_f$. To set the mesh spacing on the desired mesh to achieve this error, it is necessary to relate the mesh spacing to the expected error. This relationship is given by an *a priori* error estimate. Specifically, it is assumed

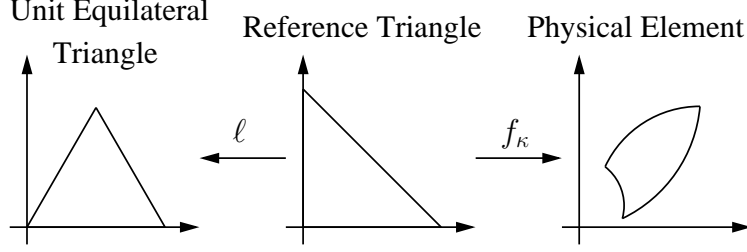


Figure 5-2: Mappings from the reference element to the unit equilateral triangle and physical space

that

$$\frac{\epsilon_\kappa}{\epsilon_\kappa^c} = \left(\frac{h_1}{h_1^c} \right)^{r_\kappa},$$

where ϵ_κ^c is the current error estimate on κ , given by 4.7, and ϵ_κ is the expected error on the elements of the desired mesh contained in κ . In this estimate, it is assumed that the smallest mesh spacing principal directions in the current and desired meshes, \mathbf{e}_1^c and \mathbf{e}_1 , align. Clearly, this alignment need not occur. However, one expects that, as the adaptation progresses, the principal directions will converge.

The order of accuracy r_κ is taken to be $r_\kappa = s + t$, where s is the convergence rate of the primal solution in an appropriate energy norm and t is the convergence rate of the adjoint solution in an appropriate energy norm. See Appendix C for an *a priori* output error estimate motivating this choice. For elliptic problems, assuming exact geometry representation and a solution space containing p th order polynomials in physical space, one can show

$$s = \min(p, \gamma - 1), \quad t = \min(p, \gamma^\psi - 1),$$

where γ and γ^ψ denote the regularities of the exact primal and dual solutions, respectively—i.e. $\mathbf{u} \in H^\gamma(\Omega)$ and $\boldsymbol{\psi} \in H^{\gamma^\psi}(\Omega)$. In this work, unless otherwise noted, optimal rates— $s = t = p$ —are assumed for all elements not containing geometric singularities. For elements touching a geometric singularity—e.g. the leading edge of an infinitely thin flat plate or a sharp trailing edge— $s + t = 1$ is used.

Equating the allowable and expected errors on the element κ gives

$$n_\kappa \frac{e_0}{N_f} = \epsilon_\kappa^c \left(\frac{h_1}{h_1^c} \right)^{r_\kappa}. \quad (5.6)$$

Using (5.5), h_1/h_1^c can be written in terms of the current mesh spacing ratio, the desired

mesh spacing ratio from the anisotropy detection step, and n_κ . In two dimensions,

$$\frac{h_1}{h_1^c} = \left(\frac{1}{n_\kappa} \frac{h_2^c h_1}{h_1^c h_2} \right)^{1/2}. \quad (5.7)$$

Then, substituting (5.7) into (5.6) and solving for n_κ gives

$$n_\kappa = \left(\frac{N_f}{e_0} \epsilon_\kappa^c \right)^{\frac{2}{r_\kappa+2}} \left(\frac{h_2^c h_1}{h_1^c h_2} \right)^{\frac{r_\kappa}{r_\kappa+2}}. \quad (5.8)$$

Finally, summing over the elements gives an equation for N_f :

$$N_f = \sum_{\kappa \in T_h} n_\kappa = \sum_{\kappa \in T_h} \left[\left(\frac{N_f}{e_0} \epsilon_\kappa^c \right)^{\frac{2}{r_\kappa+2}} \left(\frac{h_2^c h_1}{h_1^c h_2} \right)^{\frac{r_\kappa}{r_\kappa+2}} \right]. \quad (5.9)$$

Equation (5.9) can be solved for N_f , allowing n_κ to be computed from (5.8). Then, h_1 can be computed using (5.7). This calculation combined with the desired anisotropy from Section 5.1.1 completely determines the desired mesh metric.

5.1.3 Curved Mesh Generation

The metric computed as shown in Sections 5.1.1 and 5.1.2 can be input to a mesh generation package to generate a linear mesh. In this work, the Bi-dimensional Anisotropic Mesh Generator (BAMG) package is used [19, 58]. However, it is well known that to fully realize the potential of high-order methods, the geometry representation should be high-order accurate as well [13]. Thus, for curved geometries, linear meshes are not appropriate. Unfortunately, curved mesh generation is a non-trivial task. In this work, curved meshes are generated by first generating a linear mesh and then operating on that linear mesh to arrive at a suitable curved mesh. Specifically, two methods will be examined: user-generated global mappings and linear elasticity mesh movement.

User-Generated Global Mapping

The user-generated global mapping approach is motivated by the observation that it is often possible to specify a global map from a simple, linear meshing domain—such as a rectangle—to the physical domain of interest. Figure 5-3 shows a cartoon example of such a map, $g : \Omega_M \rightarrow \Omega$, where Ω_M denotes the meshing domain.

Given this map, the idea applied here is to generate a linear mesh of the meshing domain and then map that mesh to the physical domain. Of course, the resulting mesh is required to satisfy the desired mesh metric in physical space. To enforce this condition, a metric is

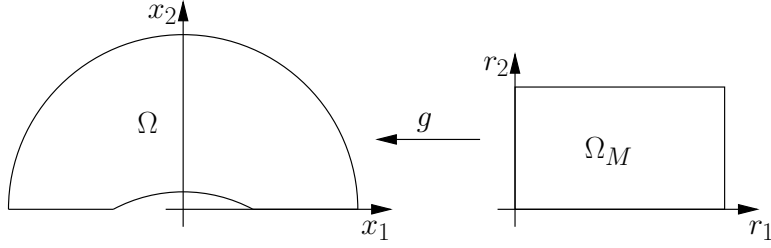


Figure 5-3: Illustration of the global map, g , from meshing to physical domain

generated for the meshing domain. The procedure for transferring the metric in physical space to a metric in meshing space is analogous to that of transferring the anisotropy request from the reference to the physical space as shown in Section 5.1.1. Specifically, ignoring any high-order terms, the map g^{-1} takes the ellipsoid defined by the metric in physical space to an ellipsoid in meshing space:

$$\mathbf{r}_e = g^{-1}(\mathbf{x}_0) + Dg^{-1}(\mathbf{x}_0)(\mathbf{x}_e - \mathbf{x}_0),$$

where \mathbf{x}_0 is the center of the ellipsoid in physical space, \mathbf{x}_e are points on the ellipsoid in physical space, and \mathbf{r}_e are points on the ellipsoid in meshing space. This ellipsoid is used to define the mesh metric in meshing space.

Using this metric, one can generate a linear mesh of the meshing space. Then, a high-order mesh of physical space is generated by adding uniformly spaced higher-order nodes to the linear mesh in meshing space and mapping the nodes to the physical domain. After computing the nodes in the physical domain, the mesh of the physical domain is defined by high-order Lagrange interpolation of these nodes. This algorithm restricts the use of the map g and the meshing domain to the mesh generation step. The alternative is to define the mesh of the physical space as the image of the linear mesh of the meshing domain under the map g . While this latter approach has some nice features—e.g. exact geometry representation—it is not used here because it requires the map g to compute the residual.

Linear Elasticity Node Movement

While the user-generated mapping technique yields good results, it places a fairly high burden on the user to generate a suitable meshing domain and associated global map. Especially for complicated three-dimensional problems, this burden is unacceptable. Thus, a second, more general, approach is also explored. In this approach, the metric is used directly to generate a linear mesh of the physical domain. After adding uniformly spaced higher-order nodes to the linear mesh, all mesh nodes are moved to generate a valid, high-

order mesh.

The node movement is governed by the linear elasticity equations, as used by [77]. Let Ω_L denote the domain covered by the linear triangulation, $T_{h,L}$. Then, in two dimensions, the node movement is governed by

$$\begin{aligned}\nabla \cdot (\nabla \mathbf{V} + \boldsymbol{\sigma}) &= 0, \quad \mathbf{x} \in \Omega_L \\ \mathbf{V} - \mathbf{V}_{BC} &= 0, \quad \mathbf{x} \in \partial\Omega_L.\end{aligned}$$

where $\mathbf{V} = [u, v]^T$ is the displacement vector,

$$\boldsymbol{\sigma} = \begin{bmatrix} AR(\nabla \cdot \mathbf{V}) & 0 \\ 0 & AR(\nabla \cdot \mathbf{V}) \end{bmatrix},$$

and AR is the aspect ratio of the underlying linear element. Using the aspect ratio to set the material properties in this manner results in high aspect ratio cells that act like nearly incompressible materials, which adds robustness to the node movement in highly stretched regions [77]. The Dirichlet boundary conditions, \mathbf{V}_{BC} , are set by requiring that

$$\mathbf{x}_p = \mathbf{x} + \mathbf{V}_{BC}(\mathbf{x}), \quad \mathbf{x} \in \partial\Omega_L,$$

where \mathbf{x}_p is the projection of the point \mathbf{x} onto the exact geometry of the physical domain.

The node movement equations are discretized using a high-order continuous Galerkin discretization. The order of this discretization, denoted q , is set by the desired order of the geometry representation. Thus, the mesh movement is given by the solution of the following problem: find $\mathbf{V}_h \in X_{h,BC}^q$ such that

$$\sum_{\kappa \in T_{h,L}} \int_{\kappa} \nabla \phi_h^T \cdot (\nabla \mathbf{V}_h + \boldsymbol{\sigma}_h) = 0, \quad \forall \phi_h \in X_{h,0}^q,$$

where the function spaces are given by

$$\begin{aligned}X_h^q &\equiv \{\phi_h \in C^0(\Omega_L) \mid \phi_h|_{\kappa} \in P^q(\kappa), \forall \kappa \in T_{h,L}\}, \\ X_{h,0}^q &\equiv \{\phi_h \in X_h^q \mid \phi_h|_{\partial\Omega_L} = 0\}, \\ X_{h,BC}^q &\equiv \{\phi_h \in X_h^q \mid \phi_h(\mathbf{x}_j) = \mathbf{V}_{BC}(\mathbf{x}_j), \forall j = 1, \dots, N_{bn}\},\end{aligned}$$

and $\{\mathbf{x}_j \mid j = 1, \dots, N_{bn}\}$ denotes the nodes on $\partial\Omega_L$. Clearly, this discrete problem is linear. Thus, computing the node movement solution requires a single linear solve. In this implementation, the sparse linear system is solved using Gaussian elimination.

5.2 Numerical Results

In this section, the adaptation algorithm is applied to four two-dimensional test flows: a flat plate, an ellipse, and two airfoil flows. In all cases, the output used to drive the adaptation is drag, and the Mach number is used to determine the anisotropy. Furthermore, the adaptation aggressiveness factor, η_a , is set to 0.1, and the error tolerance is set arbitrarily small such that it will never be reached. This tolerance was chosen to simulate very stringent accuracy requirements. For brevity, the results for all three source term discretizations are shown only for the flat plate case. For the remaining cases, only the dual consistent discretization is used.

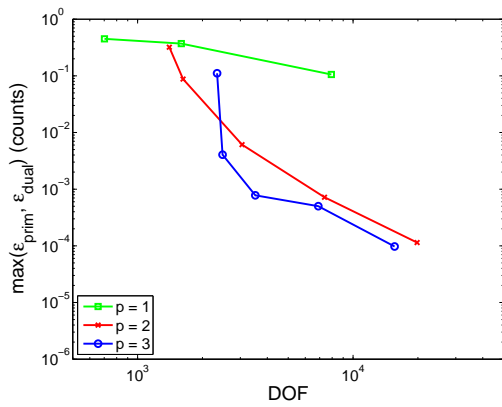
5.2.1 Flat Plate

The first test case is $M_\infty = 0.25$, $Re_c = 1 \times 10^7$ flow over a flat plate with zero pressure gradient. Figure 5-4 shows the adaptation histories for each source term discretization investigated. In particular, the error estimate and actual error are plotted versus DOF for each adaptation iteration. The error estimate shown is the maximum of the primal and dual residual estimates given by (4.5) and (4.6), respectively. The actual error is computed relative to the same value used for this case in Section 4.3.1.

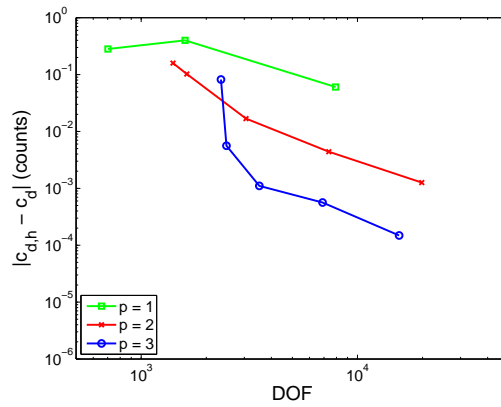
From the figure, it is clear that, for all the source term discretizations examined, the $p = 2$ and $p = 3$ discretizations are substantially more efficient than $p = 1$ in terms of DOF required to achieve a desired accuracy. For example, using the dual consistent discretization, the $p = 2$ and $p = 3$ discretizations both achieve an error of approximately 5×10^{-3} drag counts using approximately 2.5×10^3 DOF. Alternatively, the $p = 1$ discretization requires more than 4×10^4 DOF, an order of magnitude increase.

To facilitate comparison of the three source discretizations, the adaptation histories are shown again in Figure 5-5. For all p , the three source term discretizations produce very similar error estimates. However, the actual errors are somewhat different. The most drastic differences occur for $p = 2$, where the dual consistent and mixed formulation discretizations significantly outperform the standard weighting scheme. For example, using 7374 DOF, the standard weighting scheme gives an error of approximately 4×10^{-3} counts while the dual consistent and mixed formulations give errors of less than 3×10^{-4} counts using similar DOF.

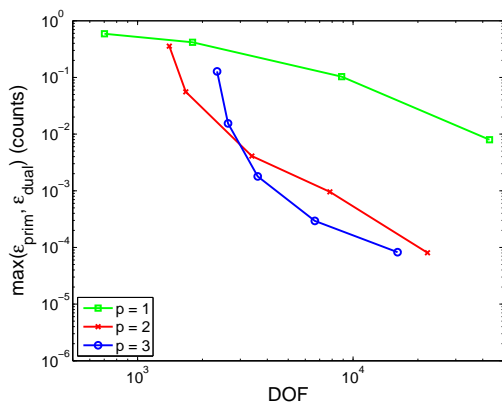
Figures 5-6 through 5-8 show skin friction distributions on the initial mesh and after two adaptation iterations for $p = 1, 2, 3$. The figures show that the computed skin friction matches the experimental data of Wieghart [32] very closely after two adaptation iterations for all p and all discretizations used. Furthermore, the skin friction distributions reinforce



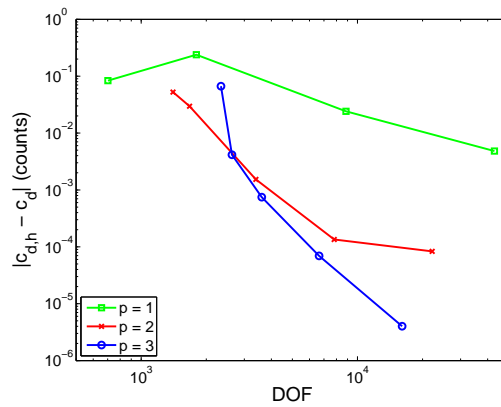
(a) Standard weighting, Error estimate



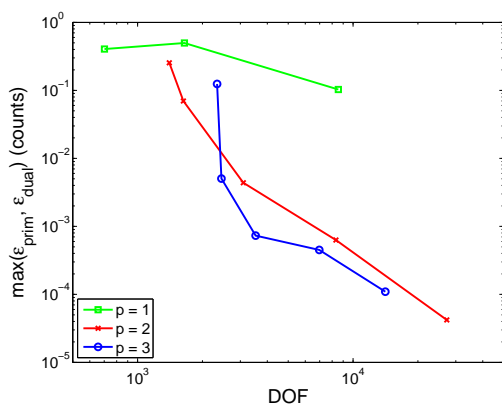
(b) Standard weighting, Actual error



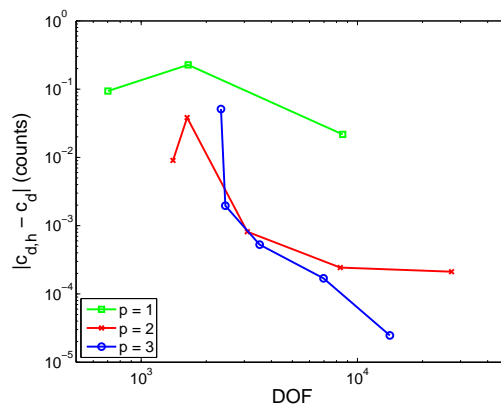
(c) Dual consistent, Error estimate



(d) Dual consistent, Actual error

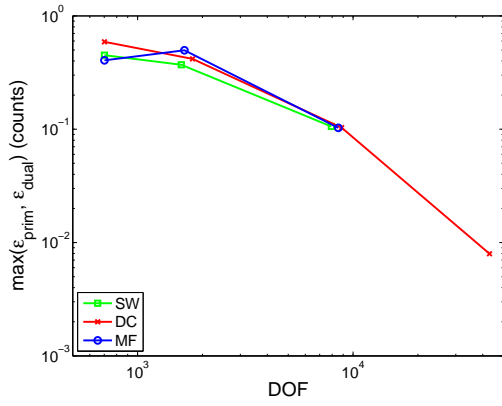


(e) Mixed formulation, Error estimate

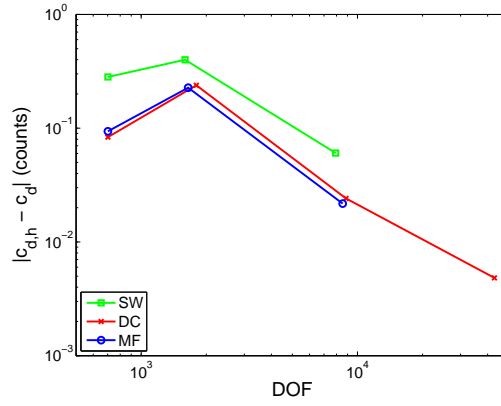


(f) Mixed formulation, Actual error

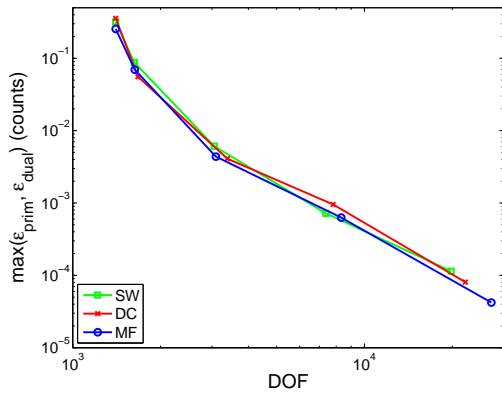
Figure 5-4: Estimated and actual drag error versus DOF for adaptation on the drag on a flat plate, comparison of $p = 1, 2, 3$ results



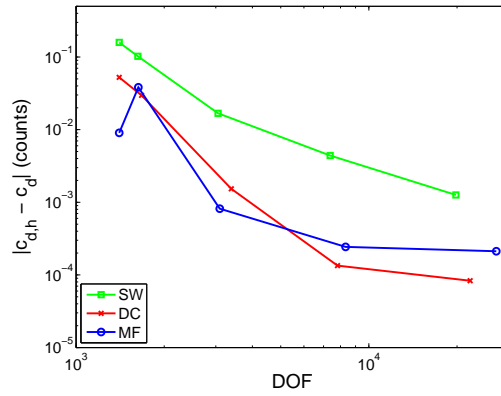
(a) $p = 1$, Error estimate



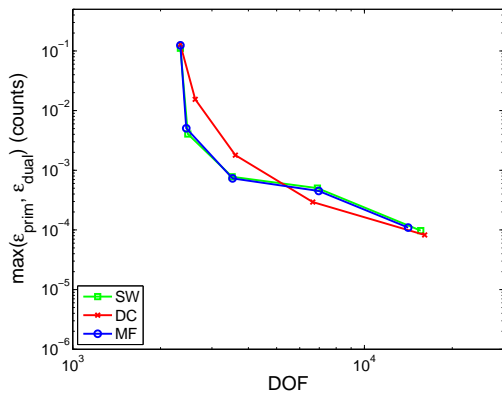
(b) $p = 1$, Actual error



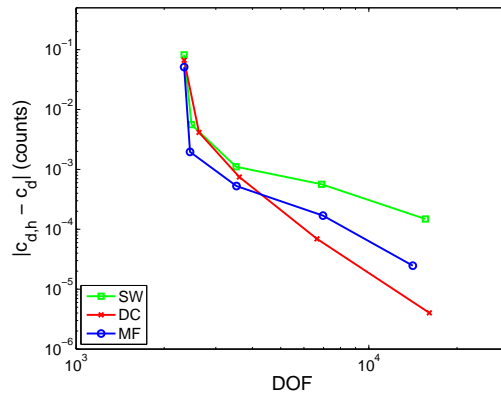
(c) $p = 2$, Error estimate



(d) $p = 2$, Actual error



(e) $p = 3$, Error estimate



(f) $p = 3$, Actual error

Figure 5-5: Estimated and actual drag error versus DOF for adaptation on the drag on a flat plate, comparison of standard weighting (SW), dual consistent (DC) and mixed formulation (MF) discretizations

the conclusion from the adaptation histories that the $p = 2$ and $p = 3$ discretizations are far superior to $p = 1$. In particular, for all three discretizations, after two adaptation iterations, the $p = 2$ and $p = 3$ skin friction distributions appear smoother than the $p = 1$ result, despite the fact that the $p = 1$ uses more than twice the DOF.

5.2.2 Ellipse

The second test case is $M_\infty = 0.25$, $Re_c = 1 \times 10^7$, $\alpha = 0$ flow over an ellipse. The ellipse has a semi-major axis length $a = 1.0$ and a semi-minor axis length $b = 0.1$. Furthermore, assuming the x -axis is oriented along the major axis of the ellipse, the flow is symmetric about $y = 0$. Thus, only the upper half of the domain is used for the simulation. The boundary of the physical domain is given by $\partial\Omega = \cup_{i=0}^3 \Gamma_i$, where

$$\begin{aligned}\Gamma_0 &= \{(x, y) \mid -100 \leq x \leq -1, y = 0\}, \\ \Gamma_1 &= \{(x, y) \mid -1 \leq x \leq 1, y = b\sqrt{1 - x^2/a^2}\}, \\ \Gamma_2 &= \{(x, y) \mid 1 \leq x \leq 100, y = 0\}, \\ \Gamma_3 &= \{(x, y) \mid -100 \leq x \leq 100, y = \sqrt{100 - x^2}\}.\end{aligned}$$

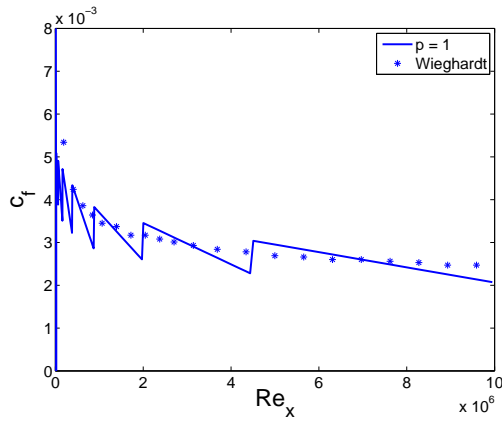
On Γ_3 , the full state boundary condition is used. Boundaries Γ_0 and Γ_2 are symmetry lines, and Γ_1 is a no slip, adiabatic wall.

Given the simple geometry of the domain, it is straightforward to construct a mapping from a rectangular meshing domain to the physical domain. Specifically, the mapping from the meshing space to the physical space, $g : \Omega_M \rightarrow \Omega$, is given by

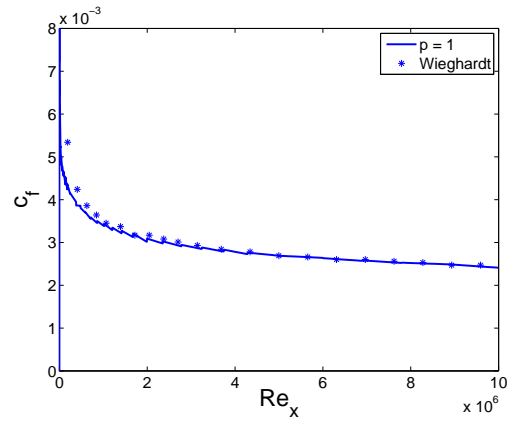
$$g(\xi, \eta) = \left[\begin{array}{c} \exp \xi \cos \eta \\ \frac{\exp \xi}{\log R} \left[b\sqrt{1 - \frac{\cos^2 \eta}{a^2}} (\log R - \xi) + \xi \sin \eta \right] \end{array} \right],$$

where $R = 100.0$ and $\Omega_M = \{(\xi, \eta) \mid 0 \leq \xi \leq \log R, 0 \leq \eta \leq \pi\}$. Furthermore, superparametric elements are used. In particular, the geometry of each element is described using $q = 5$ Lagrange polynomials for all p . To illustrate the mesh map, Figure 5-9 shows the initial mesh used for the adaptation run in both the meshing and physical spaces.

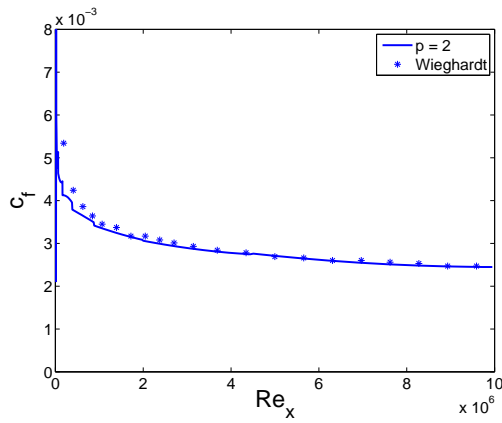
Figure 5-10 shows the adaptation history for this case. The actual error is computed relative to the drag from a $p = 3$ solution on a 8752 element mesh generated by uniformly refining (in the meshing space) the finest mesh generated by adaptation on the $p = 3$ discretization. This value is 81.344140792 counts. The figure shows that, in general, the $p = 3$ discretization is slightly more efficient, in terms of DOF required to achieve a given



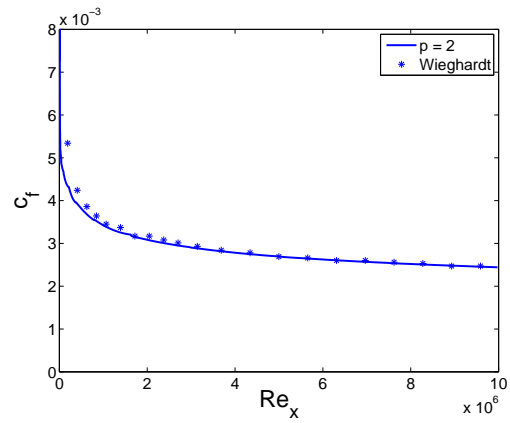
(a) Adapt Iter 0, $p = 1$, $DOF = 702$



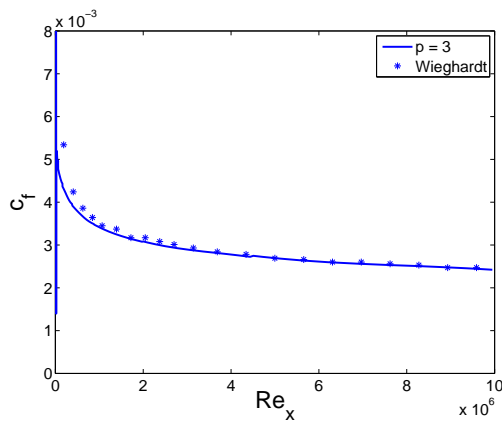
(b) Adapt Iter 2, $p = 1$, $DOF = 7929$



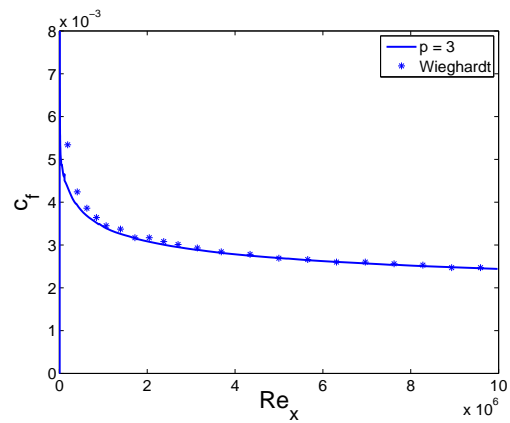
(c) Adapt Iter 0, $p = 2$, $DOF = 1404$



(d) Adapt Iter 2, $p = 2$, $DOF = 3054$

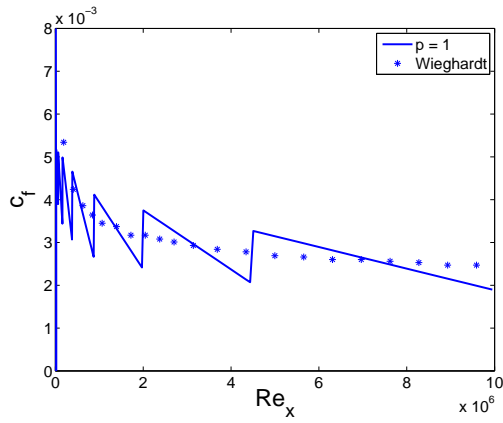


(e) Adapt Iter 0, $p = 3$, $DOF = 2340$

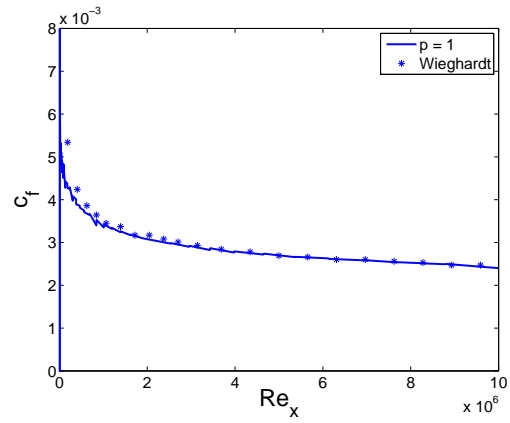


(f) Adapt Iter 2, $p = 3$, $DOF = 3520$

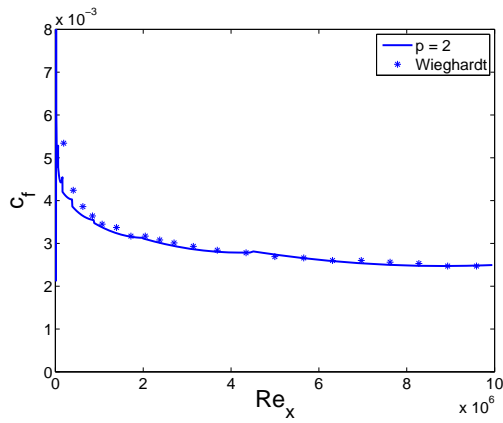
Figure 5-6: Skin friction distributions for flow over a flat plate computed using the standard weighting discretization



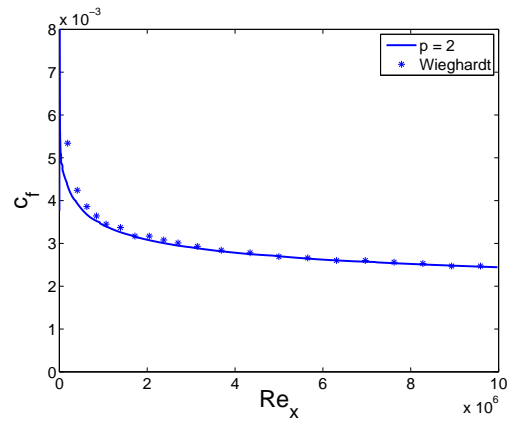
(a) Adapt Iter 0, $p = 1$, $DOF = 702$



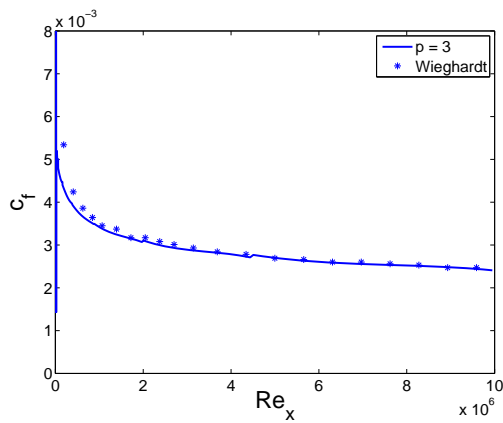
(b) Adapt Iter 2, $p = 1$, $DOF = 8850$



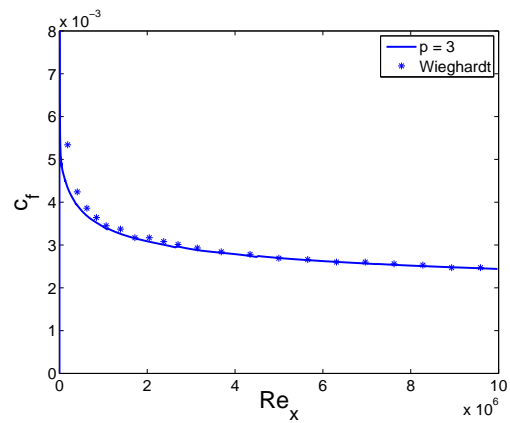
(c) Adapt Iter 0, $p = 2$, $DOF = 1404$



(d) Adapt Iter 2, $p = 2$, $DOF = 3390$

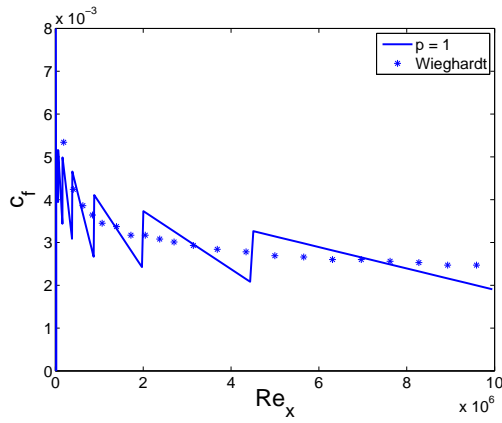


(e) Adapt Iter 0, $p = 3$, $DOF = 2340$

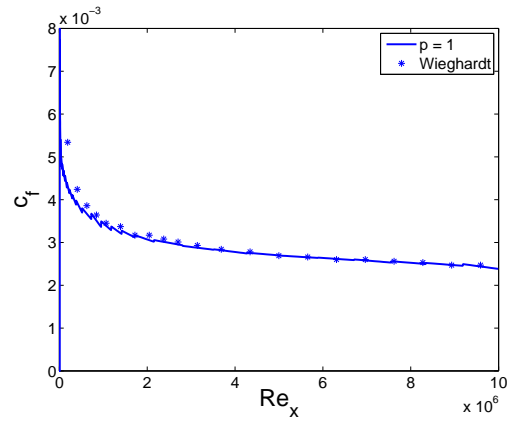


(f) Adapt Iter 2, $p = 3$, $DOF = 3610$

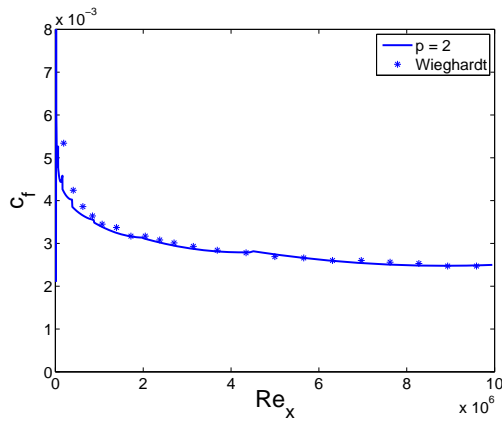
Figure 5-7: Skin friction distributions for flow over a flat plate computed using the dual consistent discretization



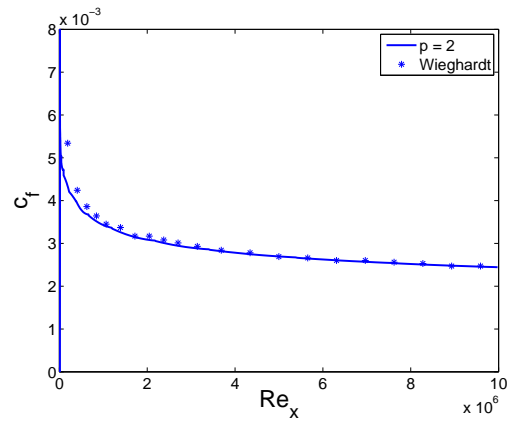
(a) Adapt Iter 0, $p = 1$, $DOF = 702$



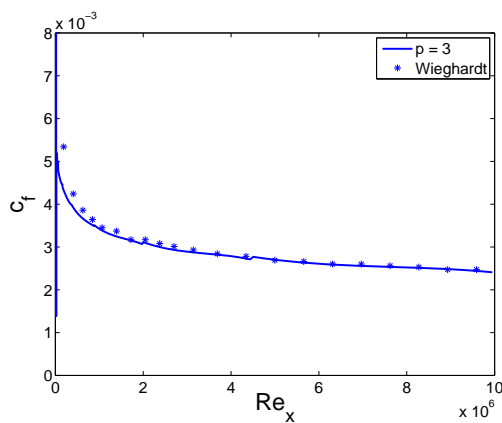
(b) Adapt Iter 2, $p = 1$, $DOF = 8544$



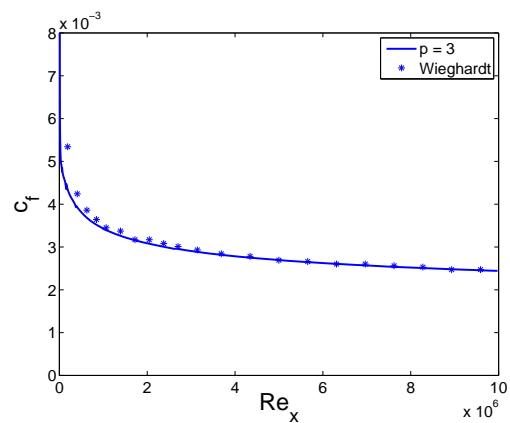
(c) Adapt Iter 0, $p = 2$, $DOF = 1404$



(d) Adapt Iter 2, $p = 2$, $DOF = 3096$

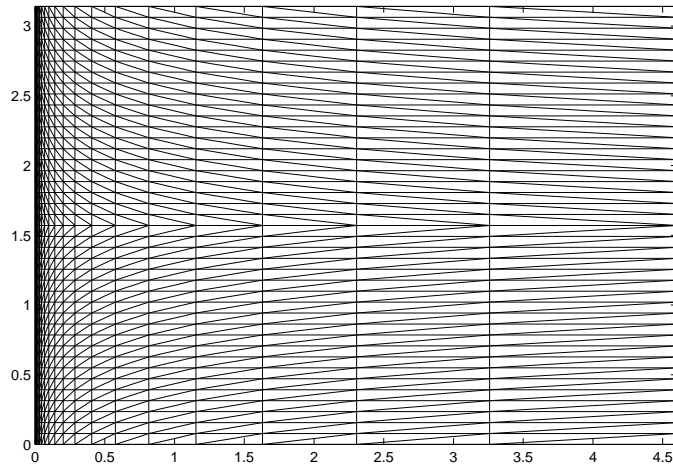


(e) Adapt Iter 0, $p = 3$, $DOF = 2340$

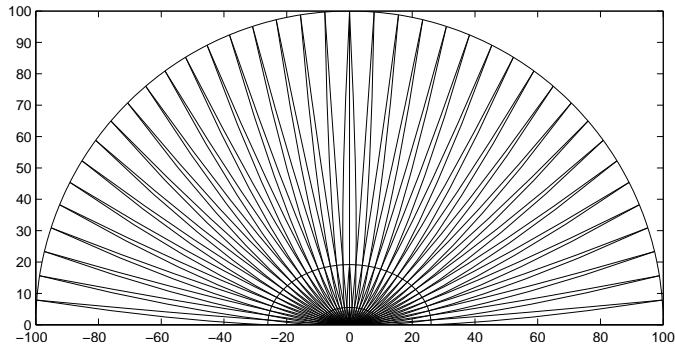


(f) Adapt Iter 2, $p = 3$, $DOF = 3530$

Figure 5-8: Skin friction distributions for flow over a flat plate computed using the mixed formulation discretization



(a) Meshing space



(b) Physical space

Figure 5-9: Initial mesh (1600 elements) for the ellipse adaptation test case, shown in both the meshing and physical spaces

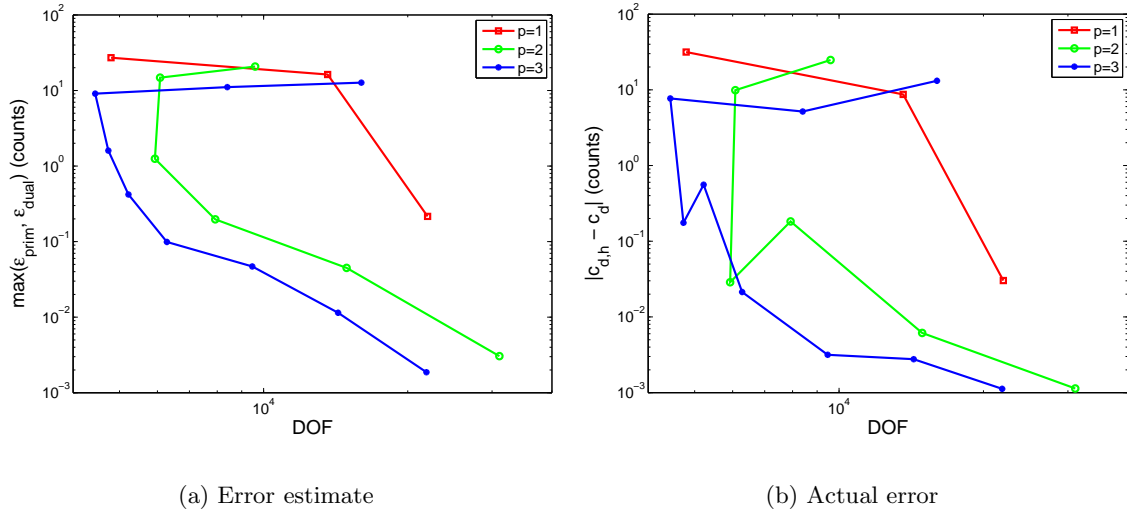
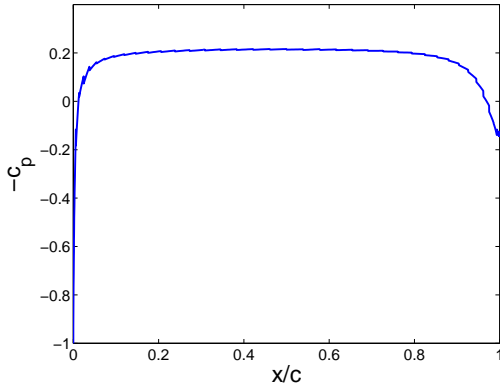


Figure 5-10: Estimated and actual drag error versus DOF for adaptation on the drag on an ellipse

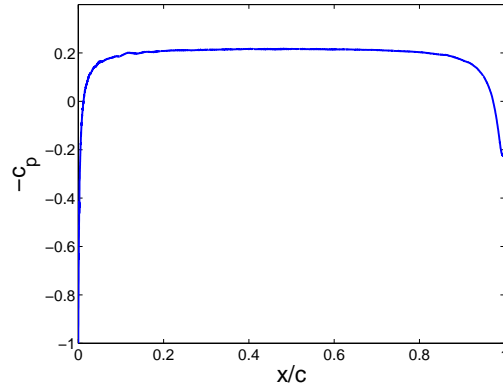
accuracy, than the $p = 2$ discretization. Moreover, both $p = 2$ and $p = 3$ are significantly more efficient than $p = 1$. For example, the $p = 1$ discretization requires 21999 DOF to achieve an error of approximately 3.0×10^{-2} counts. Alternatively, the $p = 3$ discretization gives an error of approximately 2.1×10^{-2} counts using only 6280 DOF.

Figure 5-11 shows the pressure coefficient distribution on the initial and final meshes for $p = 1, 2, 3$. While the differences in the pressure coefficient between the initial and final meshes are fairly minor, the skin friction changes dramatically. Figure 5-12 shows the skin friction distribution on the initial and final meshes for $p = 1, 2, 3$. On the initial mesh, the skin friction is highly oscillatory for all p . Furthermore, the values over much of the airfoil are significantly larger on the initial mesh than the final, leading to drag values that are too large. Despite the fact that the initial solutions are severely under-resolved, the adaptation algorithm is able to improve the quality of the meshes and the resulting solutions. Thus, on the final meshes, while some oscillations are still visible, their amplitude has been reduced, especially for $p = 2, 3$.

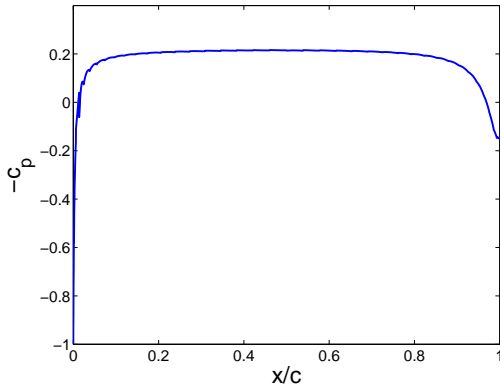
Finally, to demonstrate the changes the adaptation scheme makes to the mesh, Figure 5-13 compares the final mesh resulting from the $p = 3$ run with the initial mesh. While the initial mesh is somewhat anisotropic, it is clear that, as expected, the final adapted mesh has much more resolution near the wall.



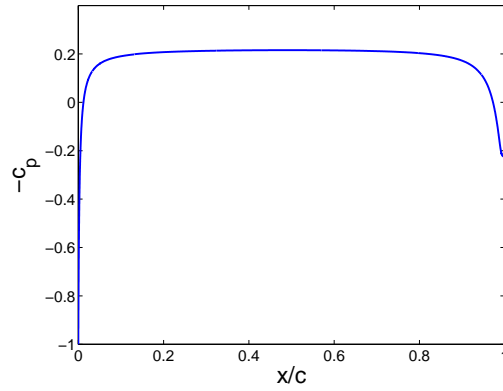
(a) $p = 1, DOF = 4800$



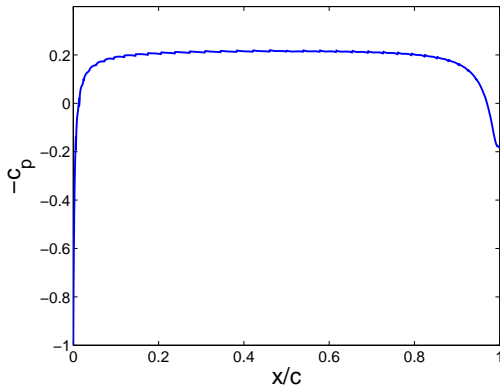
(b) $p = 1, DOF = 21999$



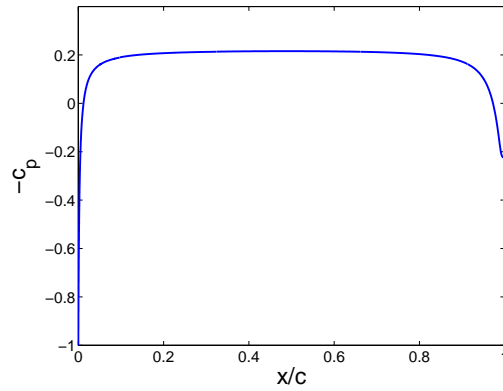
(c) $p = 2, DOF = 9600$



(d) $p = 2, DOF = 31074$

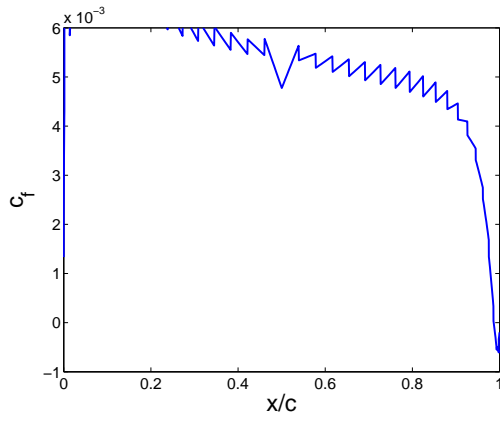


(e) $p = 3, DOF = 16000$

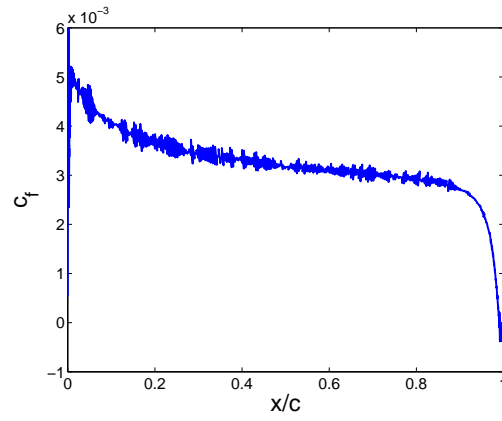


(f) $p = 3, DOF = 21880$

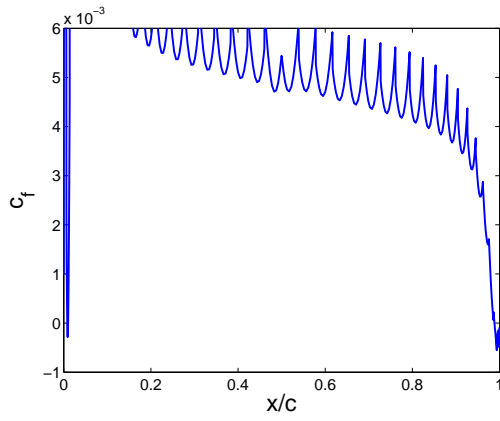
Figure 5-11: Pressure coefficient distributions on initial and final meshes for flow over an ellipse



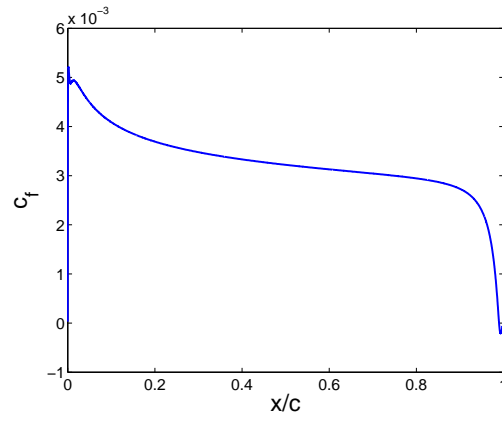
(a) $p = 1, DOF = 4800$



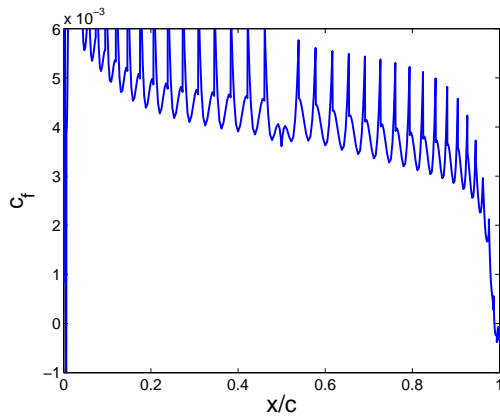
(b) $p = 1, DOF = 21999$



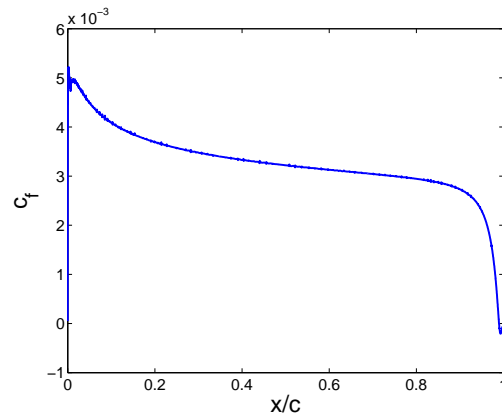
(c) $p = 2, DOF = 9600$



(d) $p = 2, DOF = 31074$



(e) $p = 3, DOF = 16000$



(f) $p = 3, DOF = 21880$

Figure 5-12: Skin friction distributions on initial and final meshes for flow over an ellipse

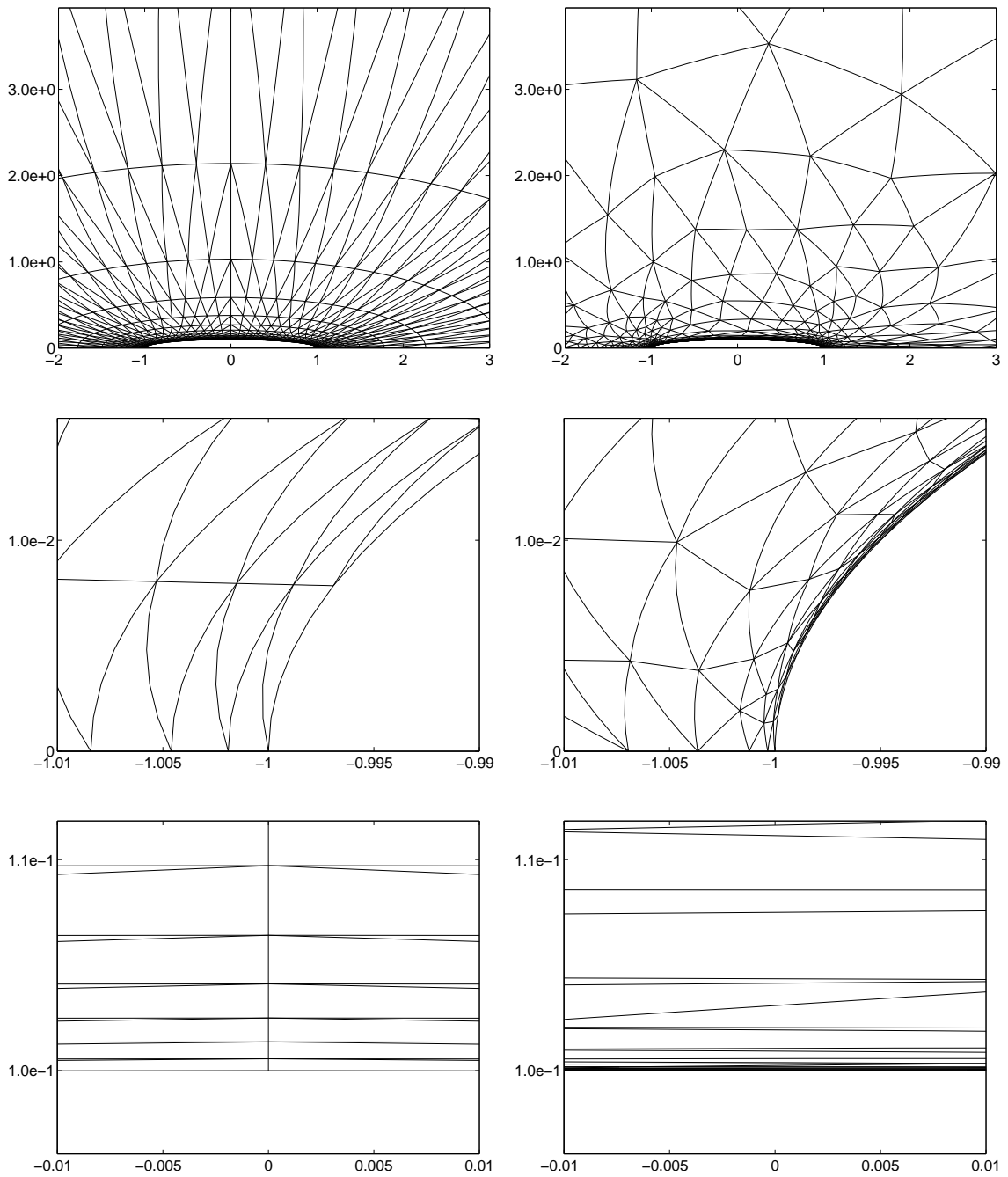


Figure 5-13: Initial (left) and final (right) meshes for the $p = 3$ adaptation on the ellipse test case

5.2.3 NACA 0012

The third test geometry is a NACA 0012 airfoil. The airfoil definition and computational domain for this case are given in Section 3.4.2. Two flow conditions are considered: $M_\infty = 0.25$, $Re_c = 1 \times 10^6$, $\alpha = 0$ and $M_\infty = 0.25$, $Re_c = 1 \times 10^7$, $\alpha = 0$. In both cases, the airfoil surface is a no slip, adiabatic wall while the outer boundary is treated with the farfield, full state approach. Only the linear elasticity mesh movement approach with isoparametric elements is used in this case. Finally, the assumed error convergence rate was lowered from $s + t = 2p$ to $s + t = p + 1$. This step was taken to improve the performance of the scheme for the initial adaptation iterations, where the solution is severely under-resolved. In this regime, it is unrealistic to expect to achieve the optimal asymptotic order of accuracy. Thus, the assumed rate was lowered, causing the adaptation to be more aggressive—i.e. decrease h more rapidly.

Figure 5-14 shows the adaptation history for the $Re_c = 1 \times 10^6$ case. The initial mesh for this case was generated by taking two $p = 3$ adaptation iterations on $M_\infty = 0.25$, $Re = 1 \times 10^5$, $\alpha = 0$ flow, starting from a 1280 element structured mesh. The actual error is computed relative to the drag from a $p = 3$ solution on a 8170 element mesh generated by taking an additional $p = 3$ adaptation iteration from the finest $p = 3$ result shown. For this solution, the drag value is 107.17549806 counts. As expected, the $p = 2$ and $p = 3$ schemes

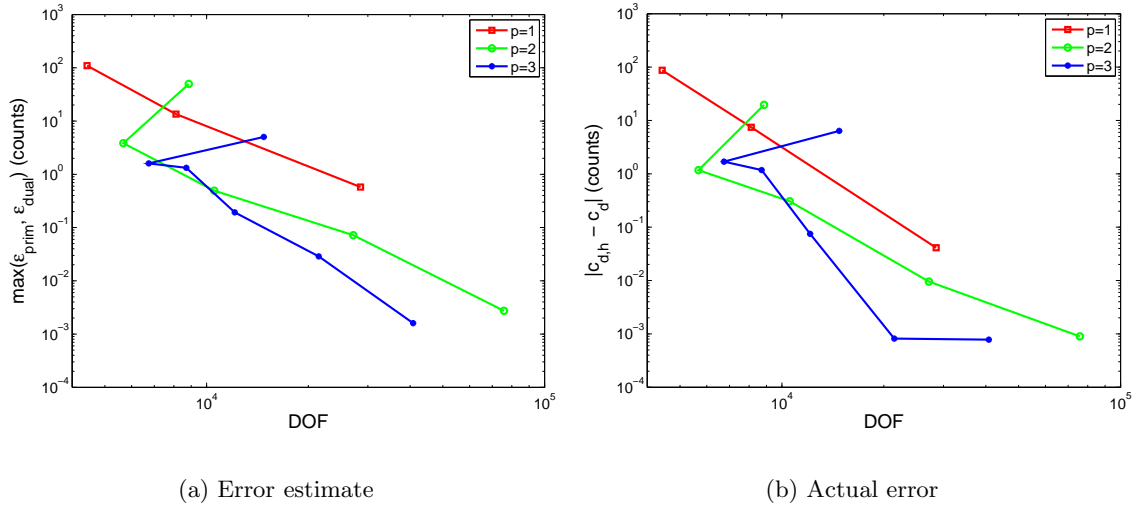


Figure 5-14: Estimated and actual drag error versus degrees DOF for adaptation on the drag on a NACA 0012 in $Re_c = 1 \times 10^6$ flow

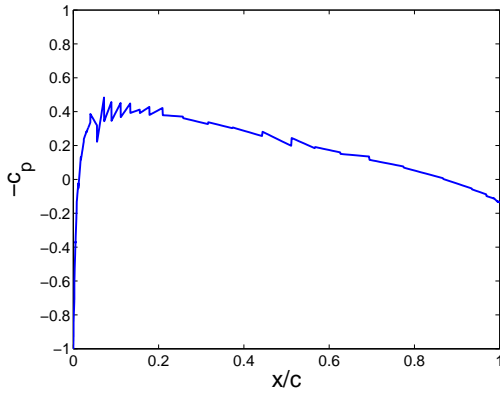
are both more efficient than the $p = 1$ discretization. For example, to achieve an error of 7.5×10^{-2} counts, the $p = 3$ discretization uses 12110 DOF, while the $p = 1$ discretization

would require more than 20000.

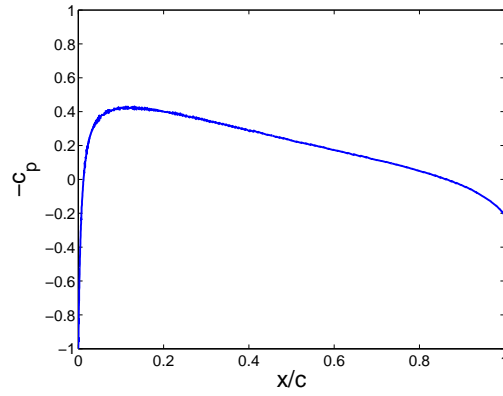
The pressure coefficient distribution on both the initial and final meshes for all p is shown in Figure 5-15. Oscillations on the initial mesh are more apparent here than in the ellipse case. However, after adaptation, smooth c_p profiles are obtained for all p . The skin friction is also oscillatory on the initial mesh, as shown in Figure 5-16. Again, the adaptation algorithm is successful in improving the mesh and solution qualities despite the poor quality of the initial solutions. Thus, the profiles on the final meshes are significantly smoother than on the initial meshes.

Figure 5-17 shows the adaptation history for the $Re_c = 1 \times 10^7$ case. The initial mesh for this case is the mesh generated after four $p = 3$ adaptation iterations on the $Re_c = 1 \times 10^6$ case. The actual error is computed relative to the extrapolated $p = 5$ result given in Section 3.4.2. For this case, the $p = 2$ method produces the best results for errors larger than approximately 1×10^{-3} counts. This fact results at least partially from the mesh generation parameters used here. The BAMG mesh generator bounds the rate of change of the mesh size by a user specified parameter (see “-ratio” in the BAMG User’s Guide [58]). When “-ratio” is set to r , the rate of change of the mesh spacing is bounded by $\log(r)$. For the $p = 1, 2$ cases, $r = 4$ was used. For $p = 3$, the solution diverged after a single adaptation iteration when using $r = 4$. Setting $r = 2$ alleviates this problem, but results in larger, less efficient meshes. Despite this fact, for small error tolerances, $p = 3$ is competitive with $p = 2$.

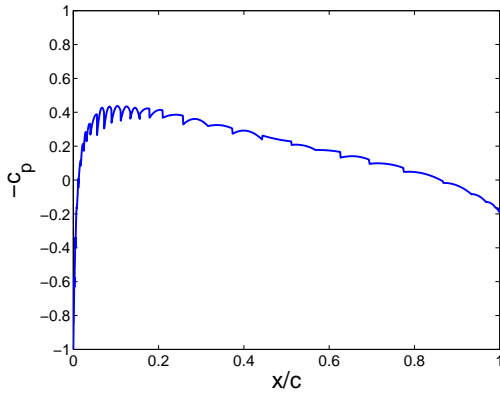
Figure 5-18 shows the pressure coefficient distribution on the initial and final meshes for $p = 1, 2, 3$. The oscillations observed on the initial mesh are virtually eliminated by the adaptive algorithm. Figure 5-19 shows the skin friction distributions. As observed in the previous cases, the oscillations in the skin friction are much more severe than those in the pressure coefficient. In the $p = 2, 3$ cases, these oscillations are largely eliminated by adaptation. However, the DOF required to eliminate the oscillations in the skin friction is larger than that necessary to achieve engineering accuracy for the drag. For $p = 1$, the skin friction on the final adapted mesh is still quite oscillatory. It is expected that additional adaptation iterations would produce a smooth skin friction distribution if enough DOF were allowed.



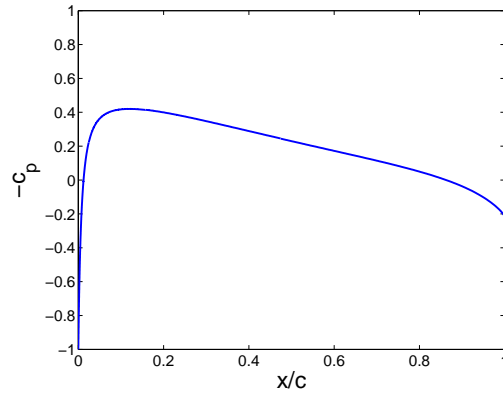
(a) $p = 1, DOF = 4428$



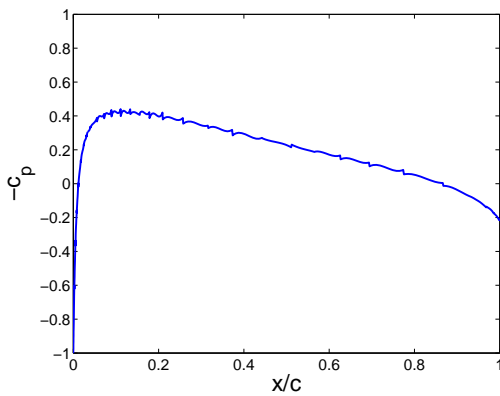
(b) $p = 1, DOF = 28533$



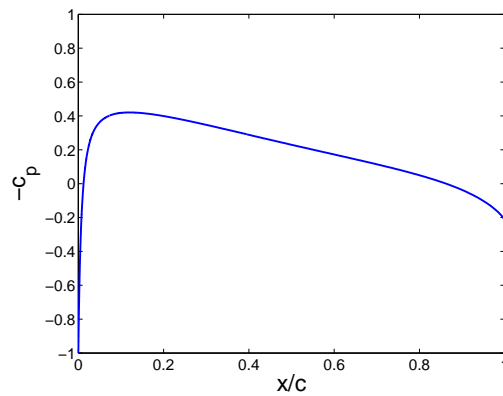
(c) $p = 2, DOF = 8856$



(d) $p = 2, DOF = 75846$

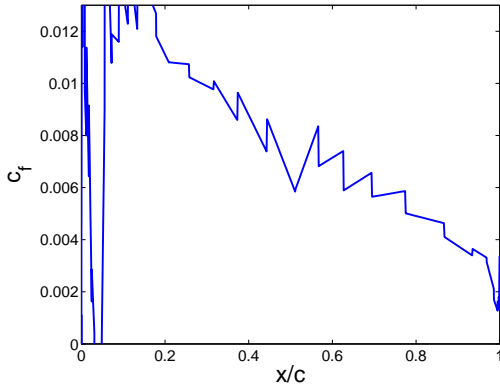


(e) $p = 3, DOF = 14760$

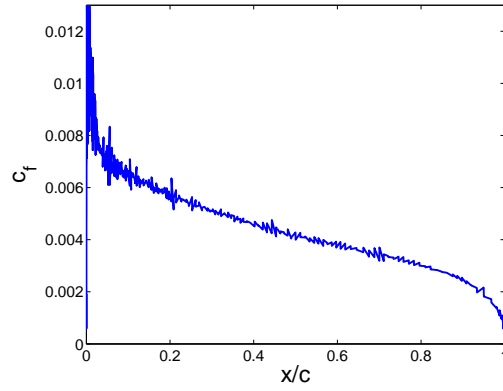


(f) $p = 3, DOF = 40820$

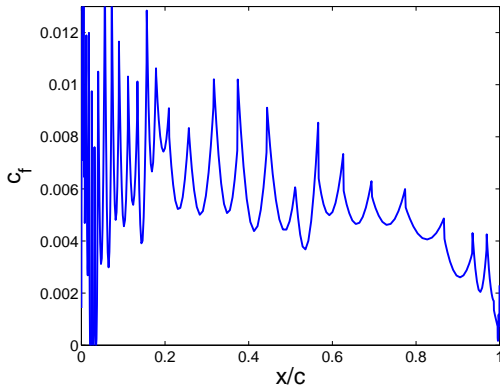
Figure 5-15: Pressure coefficient distributions on initial and final meshes for $Re_c = 1 \times 10^6$ flow over a NACA 0012



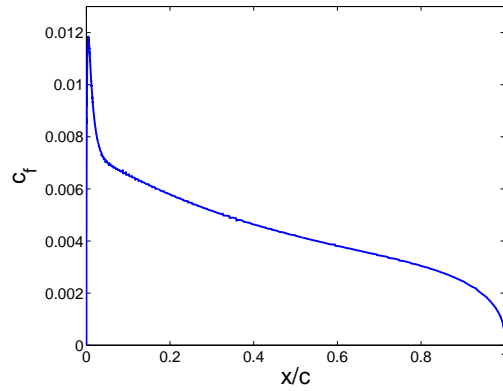
(a) $p = 1, DOF = 4428$



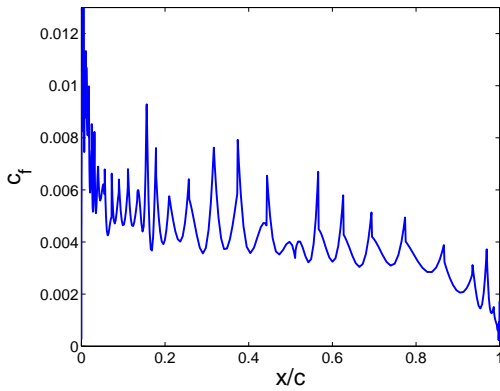
(b) $p = 1, DOF = 28533$



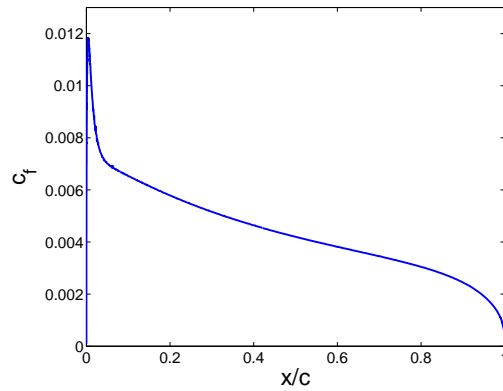
(c) $p = 2, DOF = 8856$



(d) $p = 2, DOF = 75846$



(e) $p = 3, DOF = 14760$



(f) $p = 3, DOF = 40820$

Figure 5-16: Skin friction distributions on initial and final meshes for $Re_c = 1 \times 10^6$ flow over a NACA 0012

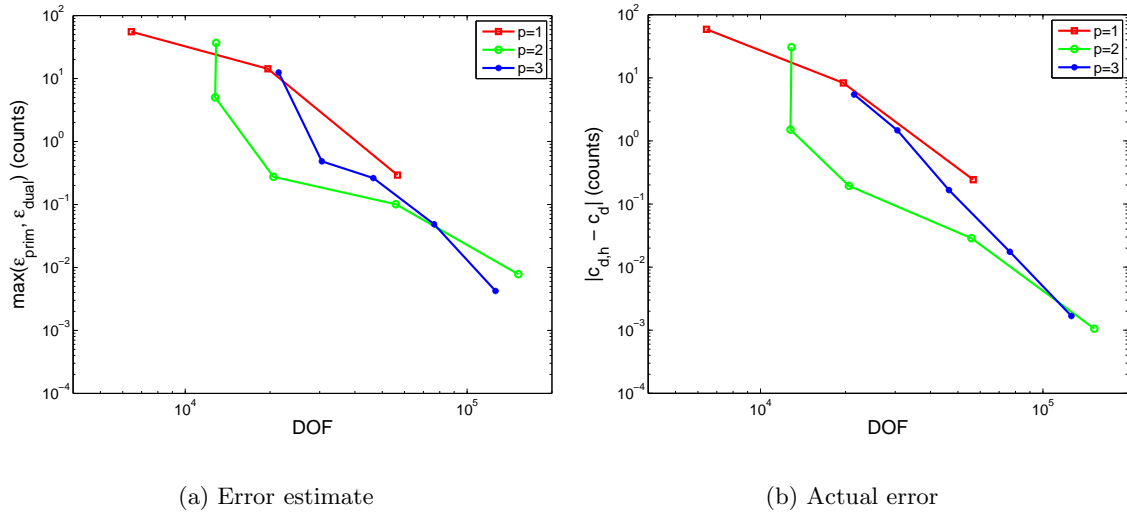
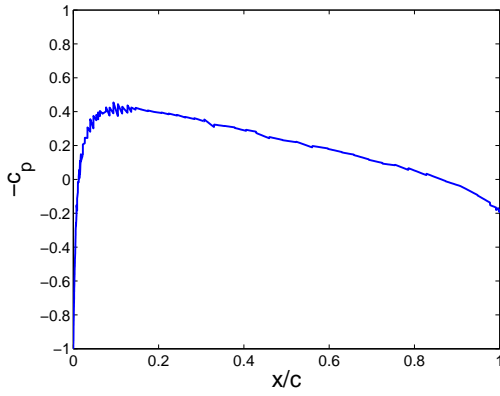
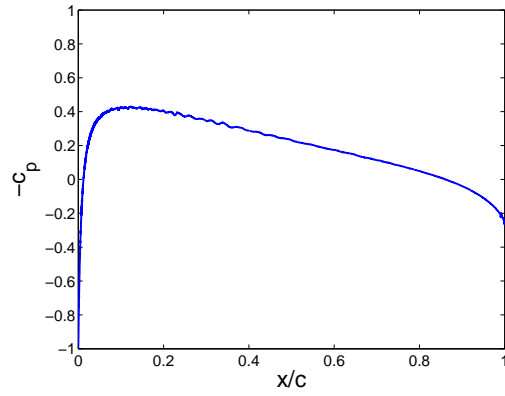


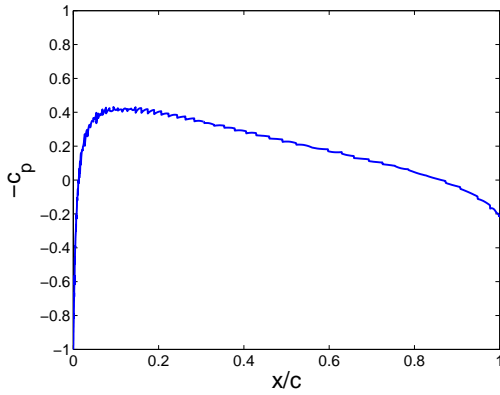
Figure 5-17: Estimated and actual drag error versus DOF for adaptation on the drag on a NACA 0012 in $Re_c = 1 \times 10^7$ flow



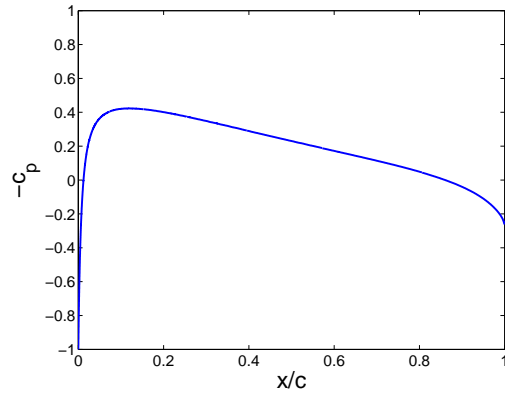
(a) $p = 1, DOF = 6438$



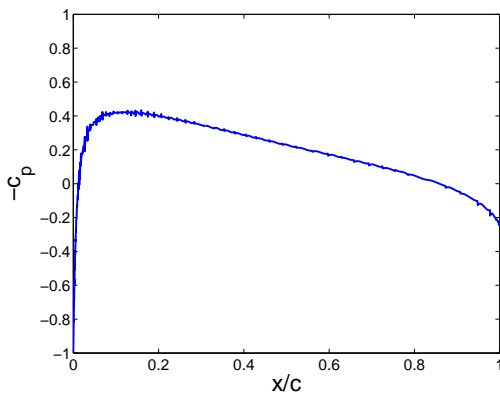
(b) $p = 1, DOF = 56664$



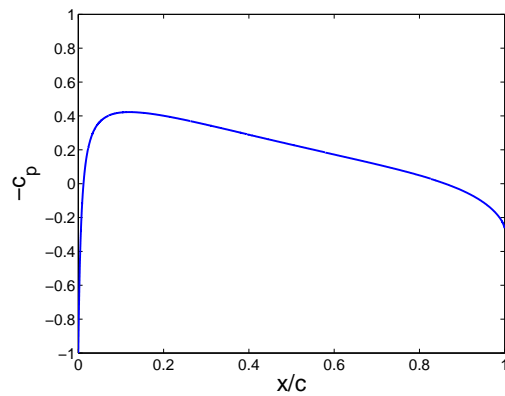
(c) $p = 2, DOF = 12876$



(d) $p = 2, DOF = 152082$

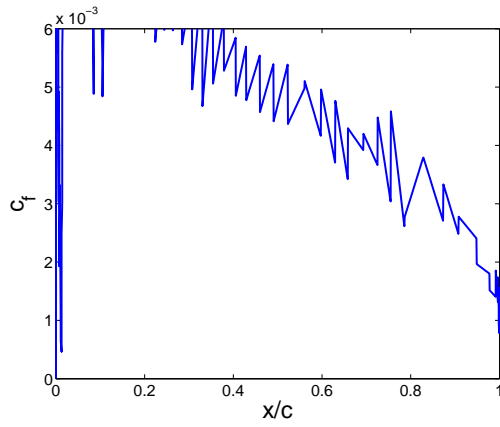


(e) $p = 3, DOF = 21460$

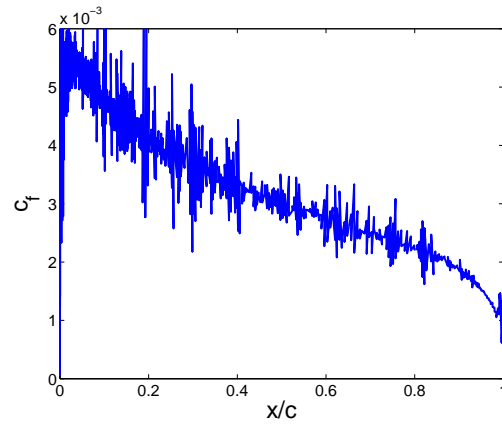


(f) $p = 3, DOF = 126170$

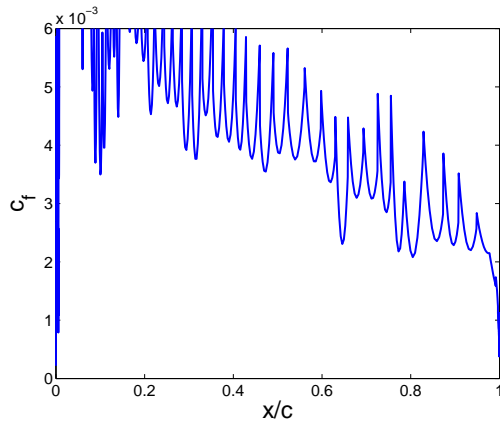
Figure 5-18: Pressure coefficient distributions on initial and final meshes for $Re_c = 1 \times 10^7$ flow over a NACA 0012



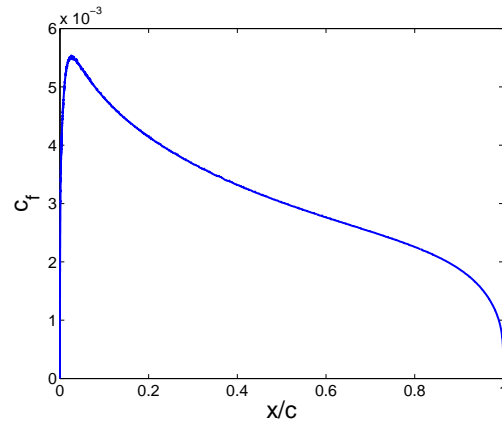
(a) $p = 1, DOF = 6438$



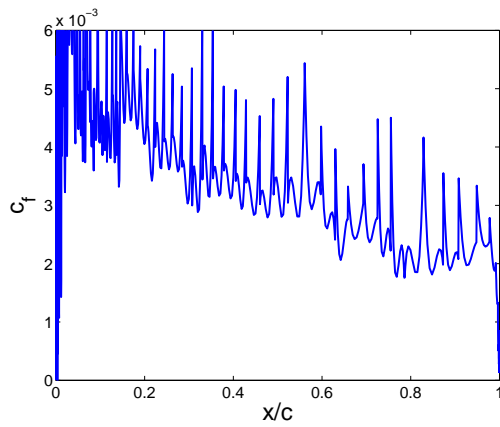
(b) $p = 1, DOF = 56664$



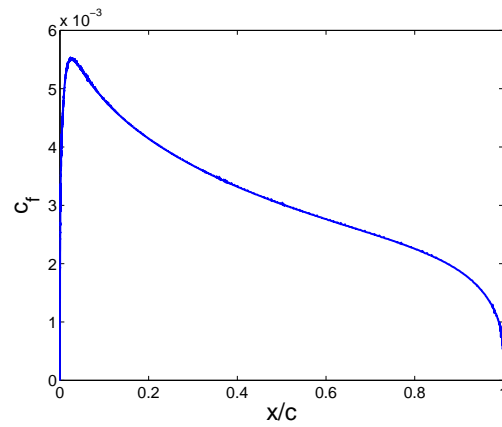
(c) $p = 2, DOF = 12876$



(d) $p = 2, DOF = 152082$



(e) $p = 3, DOF = 21460$



(f) $p = 3, DOF = 126170$

Figure 5-19: Skin friction distributions on initial and final meshes for $Re_c = 1 \times 10^7$ flow over a NACA 0012

Chapter 6

Unsteady Adaptation Algorithm

This chapter describes a modification to the adaptation algorithm presented in Chapter 5. The new algorithm, referred to as the unsteady adaptation algorithm, is intended to improve the robustness of the standard procedure by removing the need to converge a steady state solution before adapting. The motivation for this algorithm is described in more detail in Section 6.1. The algorithm is shown in Section 6.2, and Section 6.3 contains sample numerical results.

6.1 Motivation

One drawback of the adaptation algorithm presented in Chapter 5 is the need to obtain a steady state solution prior to adapting the mesh. This need places a very high robustness requirement on the flow solver. In particular, the solver must be able to obtain the steady flow and adjoint solutions on arbitrarily bad meshes. This requirement is particularly restrictive for high-order methods, which tend to produce oscillations in state variables in under-resolved regions of the flow. These oscillations often cause the discrete solution to diverge. See [79] for an example of this phenomenon in laminar flow. Thus, while it is important to maximize the robustness of the solver, another method for improving robustness is to remove the requirement that a steady state solution be obtained before adapting the mesh. One idea for removing this requirement is to march the solution forward in time and adapt the mesh at each time step.

6.2 Unsteady Algorithm

Underlying the unsteady algorithm is a procedure for estimating the error due to spatial discretization at the current time step. This error estimate is presented in Section 6.2.1.

Section 6.2.2 describes the unsteady adaptation algorithm for uniform time step. Finally, in Section 6.2.3, the algorithm is extended to allow the time step to vary throughout the mesh.

6.2.1 Error Estimation

In the unsteady case, the error estimation algorithm is slightly modified from the standard case. The output of interest is taken to be the average over the current time step of the steady state output of interest. Thus, if the output of interest of the steady simulation is denoted \mathcal{J} , the output of interest for the i th time step is given by

$$\mathcal{J}_u(\mathbf{w}) = \frac{1}{\Delta t} \int_{t^{i-1}}^{t^i} \mathcal{J}(\mathbf{w}).$$

where $\Delta t = t^i - t^{i-1}$. Only backward Euler time discretization is considered. Thus, the discrete solution from t^{i-1} to t^i is constant in time—i.e. $\mathbf{u}_H^i(\mathbf{x}, t) = \mathbf{u}_H^i(\mathbf{x})$ —which implies that

$$\mathcal{J}_u(\mathbf{u}_H^i) = \mathcal{J}(\mathbf{u}_H^i).$$

At each time step, the semi-linear form for the unsteady problem is given by

$$R_{u,H}(\mathbf{u}_H^i, \mathbf{v}_H) \equiv \frac{1}{\Delta t} \sum_{\kappa \in T_H} \int_{\kappa} \mathbf{v}_H^T (\mathbf{u}_H^i - \mathbf{u}_H^{i-1}) + R_H(\mathbf{u}_H^i, \mathbf{v}_H),$$

where R_H is the semi-linear form corresponding to the steady problem. Thus, the discrete problem for a single time step takes the following form: given $\mathbf{u}_H^{i-1} \in \mathcal{V}_H$, find $\mathbf{u}_H^i \in \mathcal{V}_H$ such that

$$R_{u,H}(\mathbf{u}_H^i, \mathbf{v}_H) = 0, \quad \forall \mathbf{v}_H \in \mathcal{V}_H.$$

Moreover, one can view this discrete problem as a spatial discretization of the following temporally discrete but spatially continuous problem: given $\mathbf{u}^{i-1} \in \mathcal{V}$, find $\mathbf{u}^i \in \mathcal{V}$ such that

$$\frac{1}{\Delta t} \int_{\Omega} \mathbf{v}^T (\mathbf{u}^i - \mathbf{u}^{i-1}) + R(\mathbf{u}^i, \mathbf{v}) = 0, \quad \forall \mathbf{v} \in \mathcal{V}, \quad (6.1)$$

where $R(\cdot, \cdot)$ is the weak form of the steady governing equations. Then, by the same analysis as in Section 4.1, the error can be written in terms of the primal or the adjoint residual:

$$\begin{aligned} \mathcal{J}_u(\mathbf{u}^i) - \mathcal{J}_u(\mathbf{u}_H^i) &= -R_{u,H}(\mathbf{u}_H^i, \boldsymbol{\psi} - \boldsymbol{\psi}_H), \\ \mathcal{J}_u(\mathbf{u}^i) - \mathcal{J}_u(\mathbf{u}_H^i) &= -\bar{R}_{u,H}^{\boldsymbol{\psi}}(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{u}^i - \mathbf{u}_H^i, \boldsymbol{\psi}_H), \end{aligned}$$

where $\boldsymbol{\psi}_H \in \mathcal{V}_H$ is arbitrary, $\boldsymbol{\psi}$ solves the adjoint problem

$$\bar{R}_{u,H}^{\boldsymbol{\psi}}(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}, \boldsymbol{\psi}) \equiv \bar{R}_{u,H}(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}, \boldsymbol{\psi}) - \bar{\mathcal{J}}_u(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}) = 0, \quad \forall \mathbf{v} \in \mathcal{V}_H + \mathcal{V},$$

and the mean-value linearizations are

$$\begin{aligned} \bar{R}_{u,H}(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}, \mathbf{w}) &\equiv \frac{1}{\Delta t} \sum_{\kappa \in T_h} \int_{\kappa} \mathbf{w}^T \mathbf{v} + \bar{R}_H(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}, \mathbf{w}), \\ \bar{\mathcal{J}}_u(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}) &\equiv \bar{\mathcal{J}}(\mathbf{u}^i, \mathbf{u}_H^i; \mathbf{v}). \end{aligned}$$

Of course, the exact solution, \mathbf{u}^i and adjoint, $\boldsymbol{\psi}$, are unknown. However, making the same approximations as in the steady case, the contribution to the error due to spatial discretization of a single element can be estimated by

$$\epsilon_{\kappa} = \frac{1}{2} \left(|R_{u,h}(\mathbf{u}_H^i, (\boldsymbol{\psi}_h - \boldsymbol{\psi}_H)|_{\kappa})| + |R_{u,h}^{\boldsymbol{\psi}}(\mathbf{u}_H^i; (\mathbf{u}_h^i - \mathbf{u}_H^i)|_{\kappa}, \boldsymbol{\psi}_H)| \right). \quad (6.2)$$

Then, a global error estimate is given by $\epsilon = \sum_{\kappa \in T_h} \epsilon_{\kappa}$. In (6.2), $\boldsymbol{\psi}_H$ is set to the discrete adjoint. That is, $\boldsymbol{\psi}_H$ solves the following problem: find $\boldsymbol{\psi}_H \in \mathcal{V}_H$ such that

$$R'_{u,H}[\mathbf{u}_H^i](\mathbf{v}_H, \boldsymbol{\psi}_H) \equiv \frac{1}{\Delta t} \sum_{\kappa} \int_{\kappa} \mathbf{v}_H^T \boldsymbol{\psi}_H + R'_H[\mathbf{u}_H^i](\mathbf{v}_H, \boldsymbol{\psi}_H) - \mathcal{J}'[\mathbf{u}_H^i](\mathbf{v}_H) = 0, \quad \forall \mathbf{v}_H \in \mathcal{V}_H.$$

The surrogate solutions $\boldsymbol{\psi}_h$ and \mathbf{u}_h are constructed by taking element-block Jacobi iterations on the $(p+1)$ th order unsteady problem. This procedure is exactly analogous to that presented in Section 4.2 for the standard case.

6.2.2 Comparison with Standard Algorithm

To best understand the unsteady procedure, it is compared with the standard algorithm. Recall the form of the standard adaptation algorithm, presented in Section 5.1. The main idea of the unsteady algorithm is to estimate the error and adapt the mesh at every time step. The solution procedure utilized in step 2 of the standard algorithm reduces to Newton's method as $\Delta t \rightarrow \infty$. For finite Δt , the state update is that resulting from a single Newton iteration on a backward Euler discretization of the analogous unsteady problem. By taking additional Newton iterations on this unsteady problem, one can find the update that solves the unsteady problem, as shown in Chapter 2. Given this solution and the appropriate adjoint, the error at the given time step can be estimated, as demonstrated in Section 6.2.1. This error estimate motivates the following algorithm:

1. Generate an initial mesh and initial condition, \mathbf{U}^0 . Initialize the CFL number, and

compute the corresponding time step, Δt . Set $i = 1$.

2. While $\|\mathbf{R}(\mathbf{U}^{i-1})\| > \delta_{tol}$,

(a) Solve the unsteady problem at the current time step, $\mathbf{R}_u(\mathbf{U}^i) = 0$, where

$$\mathbf{R}_u(\mathbf{U}^i) \equiv \frac{1}{\Delta t} \mathcal{M}(\mathbf{U}^i - \mathbf{U}^{i-1}) + \mathbf{R}(\mathbf{U}^i).$$

In particular,

i. Set $\mathbf{W}^0 = \mathbf{U}^{i-1}$ and $j = 1$.

ii. While $\|\mathbf{R}_u(\mathbf{W}^{j-1})\| > \delta_{u,tol}$, where $\delta_{u,tol}$ is the unsteady solution tolerance, and $j < J_{max}$, where J_{max} is the maximum number of iterations allowed to solve the unsteady problem,

A. Update the primal state: $\mathbf{W}^j = \mathbf{W}^{j-1} - \left(\frac{1}{\Delta t} \mathcal{M} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{W}^{j-1}}\right)^{-1} \mathbf{R}_u(\mathbf{W}^{j-1})$.

B. $j \rightarrow j + 1$.

iii. If $\|\mathbf{R}_u(\mathbf{W}^{j-1})\| > \delta_{u,tol}$, decrease the CFL number by the user specified factor, recompute the time step, and return to step (i). Otherwise, $\mathbf{U}^i = \mathbf{W}^{j-1}$.

(b) Find Ψ that solves the appropriate dual problem for the current time step:

$$\left(\frac{1}{\Delta t} \mathcal{M} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} \Big|_{\mathbf{U}^i}\right)^T \Psi = \frac{\partial J}{\partial \mathbf{U}} \Big|_{\mathbf{U}^i}.$$

(c) Use \mathbf{U}^i and Ψ to compute the error estimate, ϵ , for the current time step.

(d) Increase the CFL number by the user specified factor, and recompute the time step.

(e) If $\epsilon_{min} < \epsilon < \epsilon_{tol}$, do not adapt the mesh, $i \rightarrow i + 1$, and return to step (2). Otherwise, adapt the mesh, $i \rightarrow i + 1$, and return to step (2).

3. If $\epsilon < \epsilon_{tol}$, quit. Otherwise, the steady solution has been obtained, but it is not as accurate as requested. Thus, run standard adaptation algorithm.

The time step in this algorithm is computed from the CFL number, as shown in Section 2.4. Furthermore, the CFL number is controlled by user specified increase and decrease factors. In all cases shown, the CFL increase factor was set to 1.5, and the decrease factor was set to 2.0. Finally, given the error estimate and the surrogate solution used to set the anisotropy, the adaptation mechanics used in the unsteady algorithm are the same as those described in Section 5.1.

6.2.3 Extension to Variable Time Step

The algorithm presented in Sections 6.2.1 and 6.2.2 uses a time step that is constant throughout the mesh. Thus, although only first-order accurate, the backward-Euler temporal discretization is at least a consistent discretization of the unsteady problem. This algorithm should be quite robust—i.e. one expects that if small enough time steps are used, the algorithm could successfully march through any transients encountered between the initial condition and steady state solution. However, given that the time step is set by taking a minimum over the entire mesh, the time step will be restricted to that required by the smallest elements. To march to the steady state solution more rapidly, one can consider allowing the time step to vary from element to element. In this case, each element, κ , uses the time step Δt_κ , as defined in Section 2.4. Thus, the semi-linear form for the fully discrete problem becomes

$$R_{u,H}(\mathbf{u}_H^i, \mathbf{v}_H) \equiv \sum_{\kappa \in T_H} \frac{1}{\Delta t_\kappa} \int_\kappa \mathbf{v}_H^T (\mathbf{u}_H^i - \mathbf{u}_H^{i-1}) + R_H(\mathbf{u}_H^i, \mathbf{v}_H).$$

For this case, it is not clear that the corresponding temporally discrete but spatially continuous problem—as shown for uniform time step in (6.1)—is well-posed. However, in practice, this is of little concern, as one can instead estimate the error relative to a finer spatial discretization. Specifically, the DWR method is applied to estimate the error $\mathcal{J}(\mathbf{u}_h^i) - \mathcal{J}(\mathbf{u}_H^i)$, where $\mathbf{u}_h^i \in \mathcal{V}_h$ satisfies

$$\sum_{\kappa \in T_H} \frac{1}{\Delta t_\kappa} \int_\kappa \mathbf{v}_h^T (\mathbf{u}_h^i - \mathbf{u}_h^{i-1}) + R_h(\mathbf{u}_h^i, \mathbf{v}_h) = 0, \quad \forall \mathbf{v}_h \in \mathcal{V}_h,$$

and, for example, \mathcal{V}_h is a higher-order space than \mathcal{V}_H . The resulting error estimate has the same form as that given in (6.2). Moreover, the resulting unsteady algorithm is the same as that given in Section 6.2.2 except for the use of a different time step on each element.

6.3 Numerical Results

To demonstrate the unsteady algorithm, it is applied to three test cases. For the first two cases, simple geometries that have already been considered in Chapter 5 are used. Unlike the results of Chapter 5 however, the initial meshes used here are obviously inappropriate for RANS calculations. The final test geometry is a multi-element airfoil. This case is started from an inviscid-style, isotropic mesh, and the unsteady algorithm is used to drive the adaptation until the mesh is reasonable for obtaining the steady flow solution.

6.3.1 Flat Plate

The first test case is $Re_c = 1 \times 10^7$, $M_\infty = 0.25$ flow over a flat plate with zero pressure gradient. The steady adaptation algorithm is applied to this case in Section 5.2.1. The initial mesh used in that case contained 234 elements, and the height of the first layer of elements was approximately $\Delta y^+ = 9.5$ at $x/c = 1.0$. The initial mesh used here, shown in Figure 6-1, has only 78 elements. Furthermore, while the spacing in the x direction is

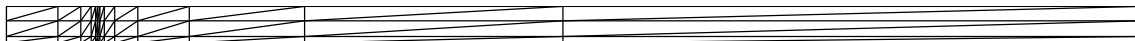


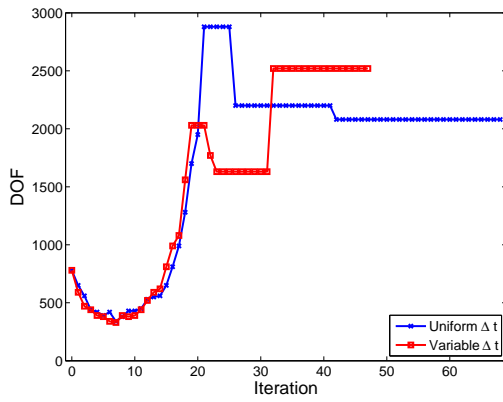
Figure 6-1: Initial mesh for unsteady adaptation applied to flow over a flat plate

the same as the 234 element mesh, the y spacing is much coarser. In particular, the height of the first layer of elements is approximately $\Delta y^+ = 2.1 \times 10^3$ at $x/c = 1.0$. This spacing is clearly too large for accurate RANS boundary layer computations. More importantly, it has not been possible to obtain converged $p > 1$, steady state solutions on this mesh. However, the unsteady adaptation algorithm succeeds using this very coarse initial mesh. Specifically, initializing the state to uniform flow, i.e.

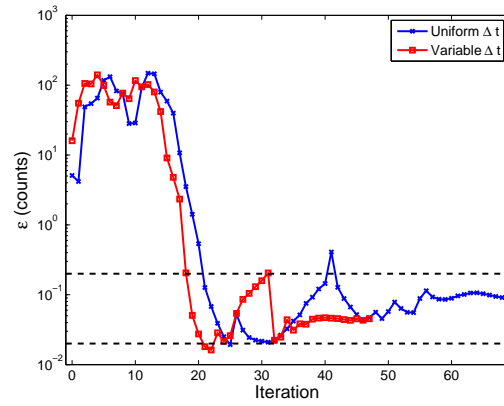
$$\frac{\rho}{\rho_\infty} = 1.0, \quad \frac{\rho u}{\rho_\infty u_\infty} = 1.0, \quad \frac{\rho v}{\rho_\infty u_\infty} = 0.0, \quad \frac{\rho E}{\rho_\infty u_\infty^2} = 29.07, \quad \frac{\rho \tilde{\nu}}{\mu_\infty} = 1.0,$$

with the initial CFL set to 1.0 and using the $p = 3$, dual consistent discretization, the unsteady adaptation algorithm eventually converges to a steady state solution that satisfies the desired error tolerance of 0.2 drag counts.

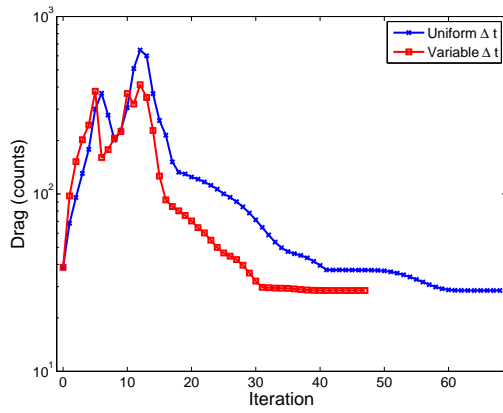
The adaptation histories for both the uniform and variable time step versions of the algorithm are shown in Figure 6-2. Examining the uniform time step results, the figure shows that the estimated error drops below the desired tolerance at iteration 21. However, since the spatial residual is not yet converged, the algorithm continues to march forward in time without adapting the mesh. At iteration 25, the error estimate drops below the minimum error request (one-tenth of the specified error tolerance). Thus, mesh adaptation is performed. In fact, the mesh is coarsened because the error is smaller than required. At iteration 42, the error estimate re-enters the desired range for the final time. Thus, no further adaptation is performed. Also at iteration 42, the time step is decreased by nearly two orders of magnitude. This decrease was required to obtain a converged solution for the time step. The steady state solution is obtained after an additional 26 time steps. The results for the variable time step case are qualitatively similar except that fewer iterations are required. In particular, the variable time step algorithm obtains the steady state solution



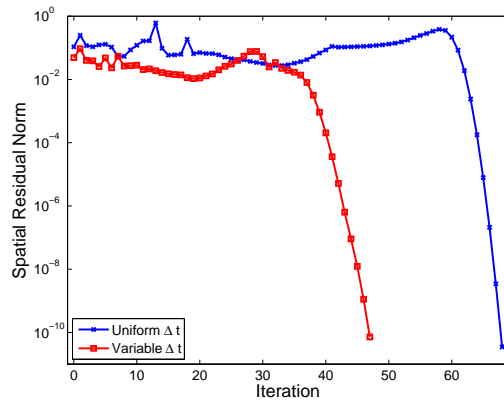
(a) DOF



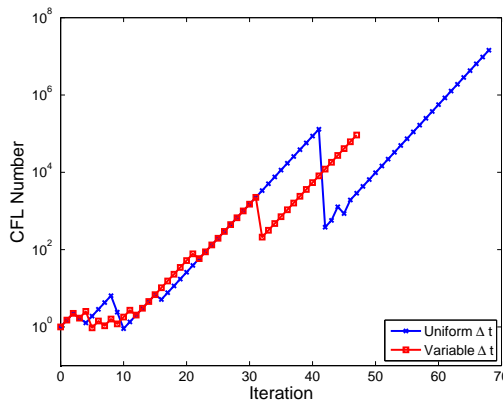
(b) Error estimate



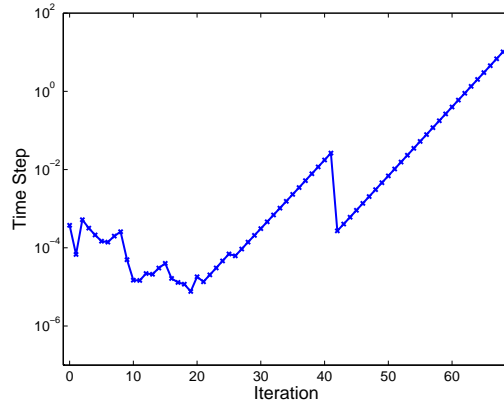
(c) Drag



(d) Spatial residual norm



(e) CFL Number



(f) Time step (Uniform Δt only)

Figure 6-2: Unsteady adaptation history for the flat plate test case

in 48 iterations, compared to 68 for the uniform time step case.

Meshes generated by the uniform time step algorithm are shown in Figure 6-3. Over the course of the adaptation run, the number of DOF increases by a factor of approximately 2.7 from the initial to the final mesh. The largest mesh generated during the run is larger than the final mesh by a factor of approximately 1.4.



(a) Finest mesh, 288 elements



(b) Final mesh, 208 elements

Figure 6-3: Finest and final meshes for unsteady adaptation using uniform time step applied to flow over a flat plate

The final mesh generated using the variable time step approach is shown in Figure 6-4. While this mesh has more elements than the final mesh produced by the uniform time step



Figure 6-4: Final mesh (252 elements) for unsteady adaptation using variable time step applied to flow over a flat plate

case, it also gives a smaller error estimate, although both solutions satisfy the requested tolerance.

To ensure that the unsteady algorithm is producing reasonable results, the final error, error estimate, and DOF can be compared to the results of the standard algorithm. At the converged steady state, the result from the unsteady algorithm with uniform time step has 2080 DOF. The maximal error estimate ($\max(\epsilon_{prim}, \epsilon_{dual})$) for the solution on the final mesh is 2.45×10^{-2} counts, and the actual error is 2.94×10^{-2} counts. Using the variable time step version of the algorithm, the final steady solution with 2520 DOF has a maximal error estimate of 1.37×10^{-2} counts and an actual error of 2.31×10^{-2} counts. Alternatively, after a single iteration, the standard algorithm for $p = 3$, started with 2340 DOF, gives 2630 DOF with a maximal error estimate of 1.55×10^{-2} and an actual error of 4.15×10^{-3} .

6.3.2 Ellipse

The second test case is $Re_c = 2 \times 10^7$, $M_\infty = 0.25$, $\alpha = 0$ flow over an ellipse. The geometry for this case is the same as that considered in Section 5.2.2, but the Reynolds number is twice as large. The $p = 3$, dual consistent discretization is used. The initial condition is the same as that used for the flat plate case, but the initial CFL was set to 0.1. For this case, only uniform time step results are presented.

As in the flat plate case, the initial mesh, shown in Figure 6-5, is extremely coarse. It contains 400 $q = 3$ elements. All meshes for this case are generated using the global mapping approach described in Section 5.1.3. In the figure, each edge of the true mesh is approximated by linear interpolation between the $q = 3$ nodes on the edge.

The adaptation history is shown in Figure 6-6. As with the flat plate case, the mesh is initially coarsened. After this initial coarsening, elements are added near the body until the error tolerance is satisfied at 21 iterations. The error estimate drops below the minimum error request after 24 iterations, and the mesh is coarsened again. The final mesh adaptation occurs at the 37th iteration, after which 26 additional iterations are required to converge the steady solution.

Figure 6-7 shows the finest mesh, generated after 21 iterations, and the final mesh. The finest mesh is significantly finer near the ellipse. This resolution is required to resolve the very thin boundary layer that forms after starting the calculation from uniform flow. As expected, the final mesh shows significant refinement in the boundary layer relative to the initial mesh. However, the wake is left somewhat under-resolved. It is expected that for a smaller error tolerance, refinement would occur in the wake as well.

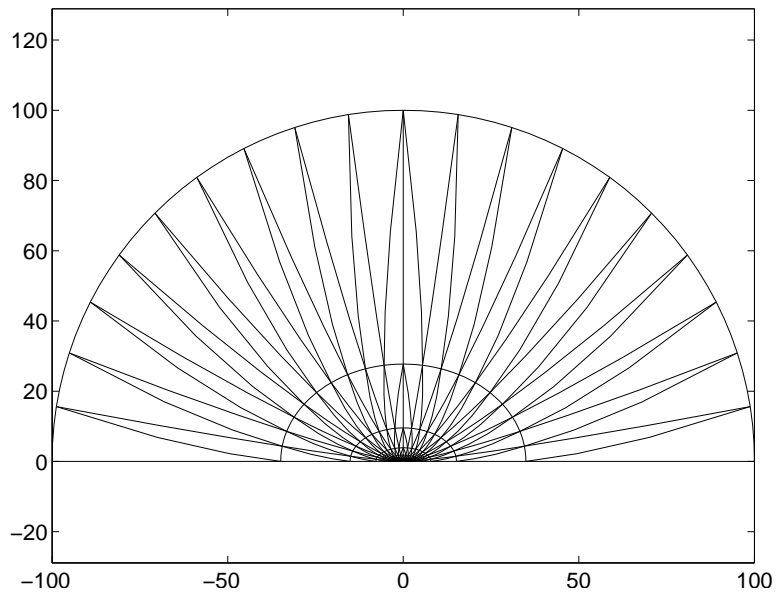
6.3.3 Advanced Energy Efficient Transport Three-Element Airfoil

The final test case is $M_\infty = 0.26$, $Re_c = 9 \times 10^6$ (based on the cruise configuration chord), $\alpha = 8^\circ$ flow around the advanced Energy Efficient Transport (EET) three-element airfoil. For this case, $p = 3$, isoparametric elements are used, and the curved elements are generated via the linear elasticity node movement approach. The uniform time step version of the algorithm is used, and the output of interest is drag.

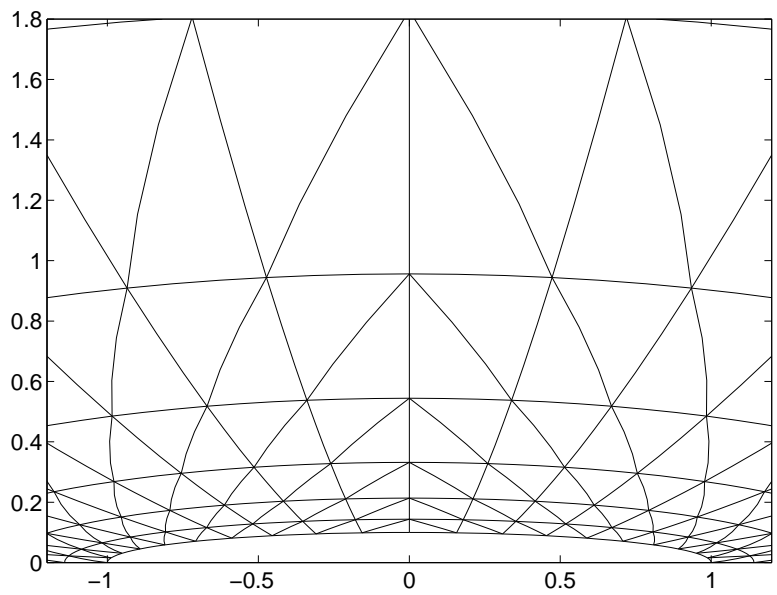
The initial mesh used here, shown in Figure 6-8, is an 11063 element, isotropic mesh. This mesh may be adequate for inviscid flow, but it is clearly not appropriate for a high accuracy RANS calculations.

To begin the calculation, the flow is initialized to zero flow. Specifically,

$$\frac{\rho}{\rho_\infty} = 1.0, \quad \frac{\rho u}{\rho_\infty u_\infty} = 0.0, \quad \frac{\rho v}{\rho_\infty u_\infty} = 0.0, \quad \frac{\rho E}{\rho_\infty u_\infty^2} = 26.416, \quad \frac{\rho \tilde{\nu}}{\mu_\infty} = 1.0 \times 10^{-2}.$$

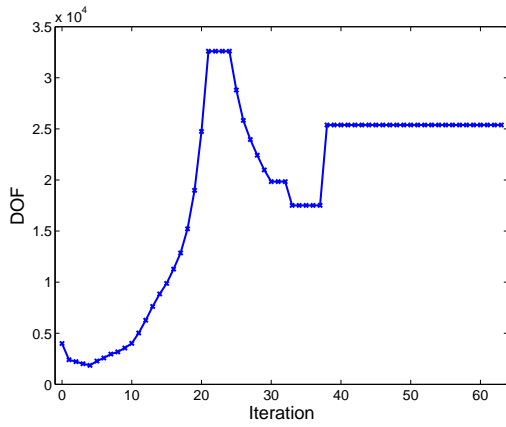


(a) Entire domain, 400 elements

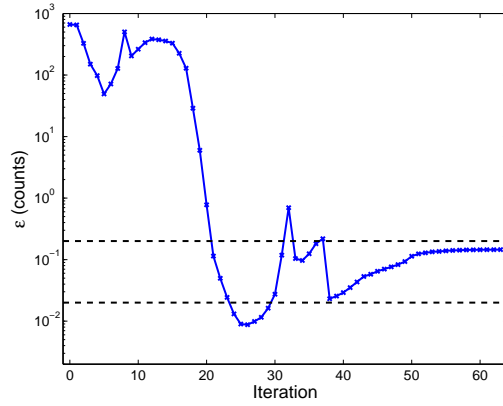


(b) Zoom

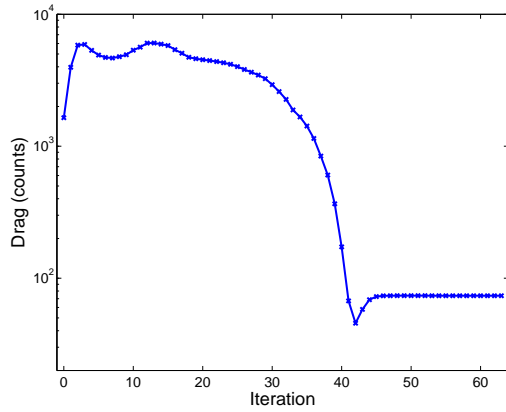
Figure 6-5: Initial mesh for unsteady adaptation applied to flow over an ellipse



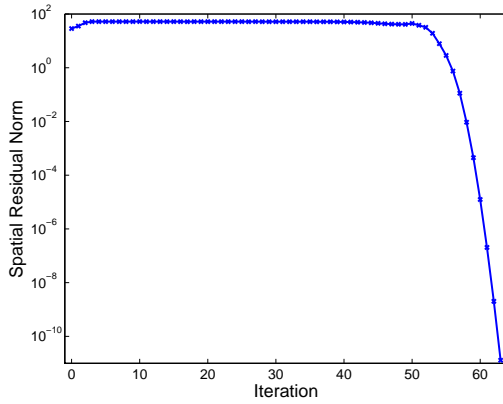
(a) DOF



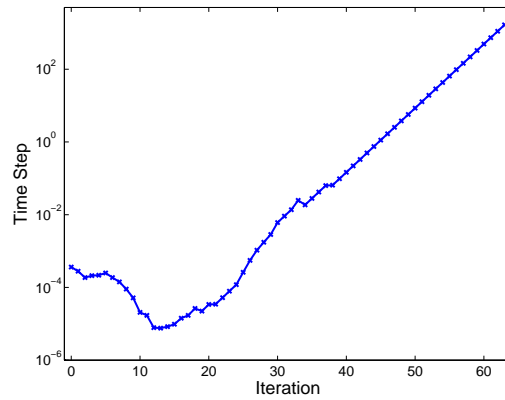
(b) Error estimate



(c) Drag

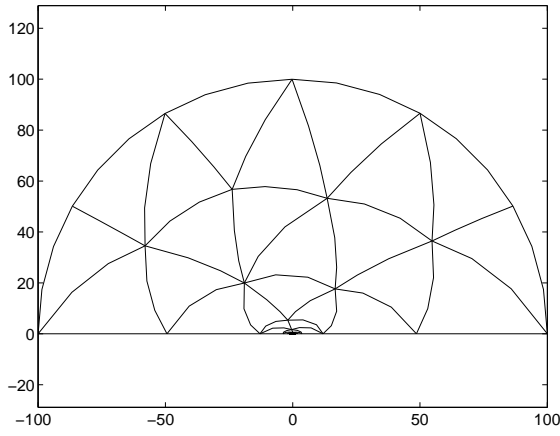


(d) Spatial residual norm

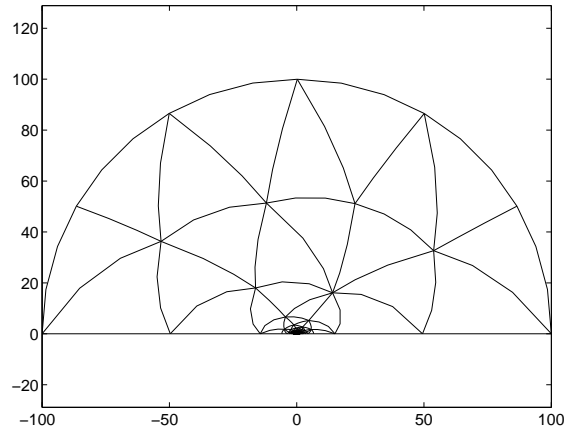


(e) Time step

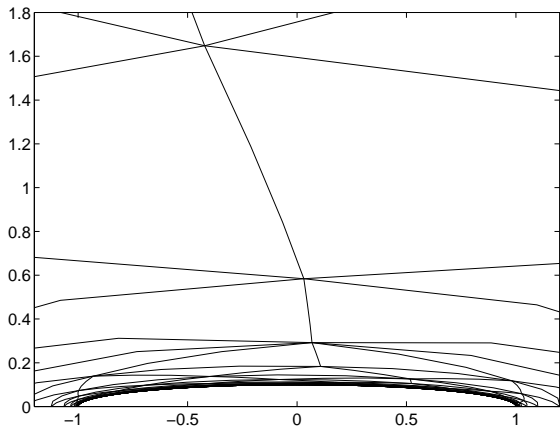
Figure 6-6: Unsteady adaptation history for the ellipse test case



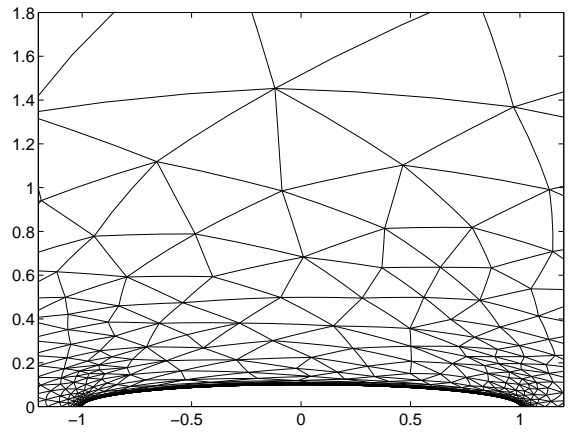
(a) Finest mesh, 3260 elements



(b) Final mesh, 2538 elements



(c) Finest mesh, zoom



(d) Final mesh, zoom

Figure 6-7: Finest and final meshes for unsteady adaptation applied to flow over an ellipse

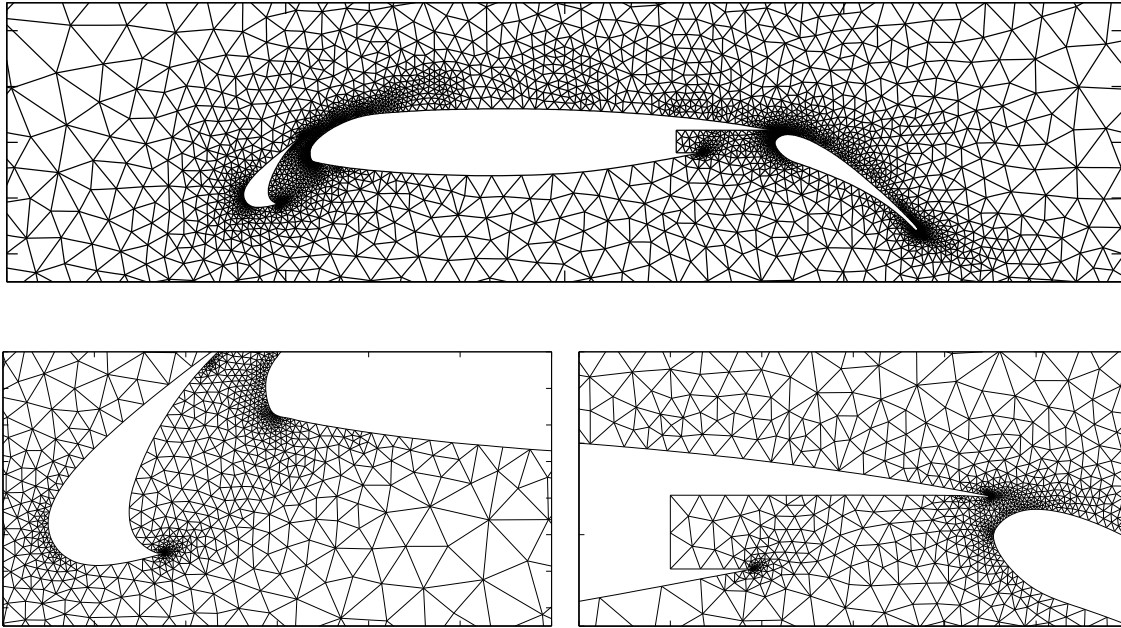


Figure 6-8: Initial mesh for unsteady adaptation applied to flow over the EET 3-element airfoil

This initial condition—as opposed to the freestream condition used in the flat plate and ellipse cases—is used to improve the robustness of the procedure. Using the freestream flow initial condition here leads to a mesh generation failure due to the very small, high aspect ratio elements requested to resolve the extremely thin transient boundary layer encountered at the beginning of the procedure. Starting from the zero flow condition relieves this transient somewhat. Instead, a different transient is encountered due to the mismatch between the farfield boundary conditions, which are consistent with the freestream flow, and the initial condition.

For this case, the unsteady adaptation algorithm is not used to converge the final steady state solution. Due to the large cost of the current algorithm and the length of the transient encountered when starting from zero flow, it is used only as a start-up procedure to obtain a reasonable mesh. In particular, the unsteady algorithm is used to march the solution forward in time approximately 85 freestream flow convective time scales. After this time, the mesh shown in Figure 6-9 is obtained. Clearly, this mesh, which contains 10332 elements, has significantly more resolution in the boundary layer than the initial mesh. Furthermore, it is possible to obtain the steady state solution on this mesh, allowing the standard adaptation to be applied. After a single standard adaptation iteration, the mesh shown in Figure 6-10 is generated. This mesh contains 11620 elements and is very similar to the mesh generated

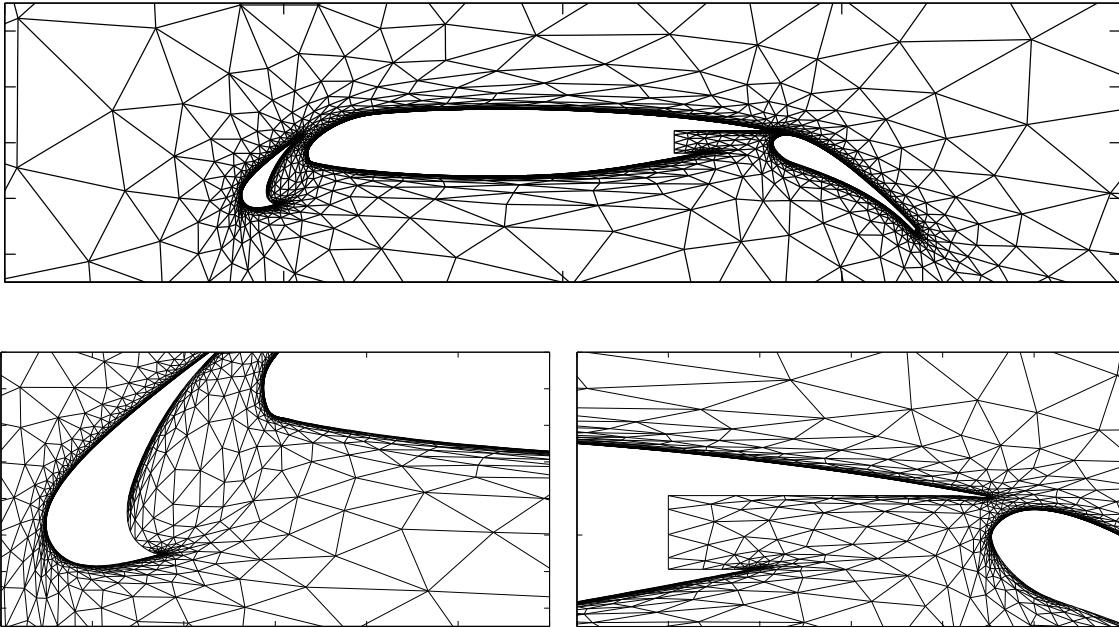


Figure 6-9: Mesh obtained by unsteady adaptation applied to flow over the EET 3-element airfoil

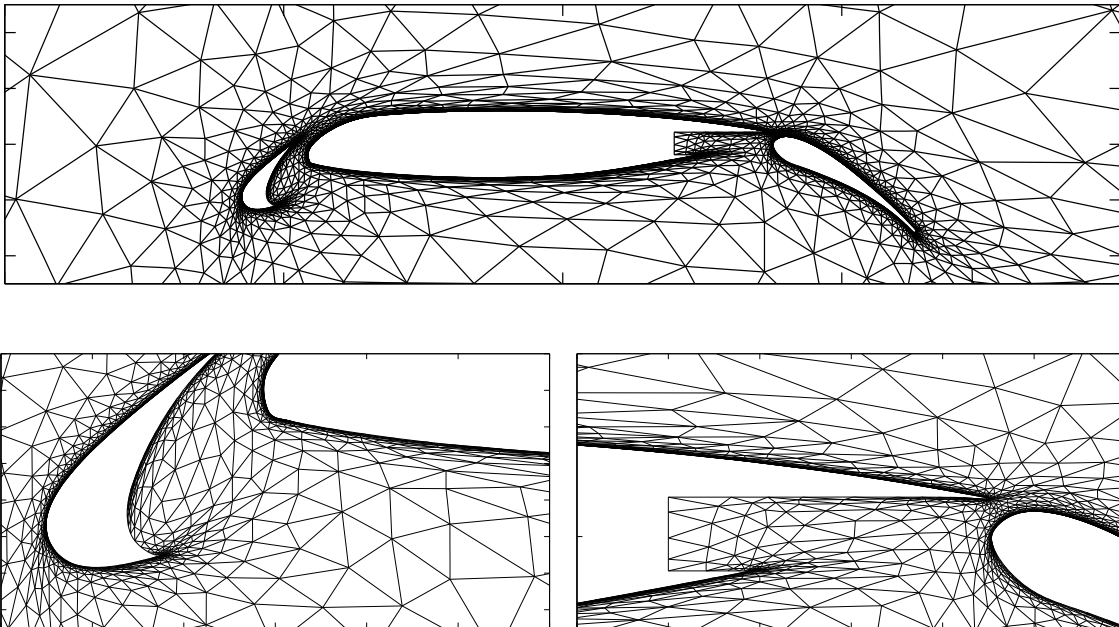
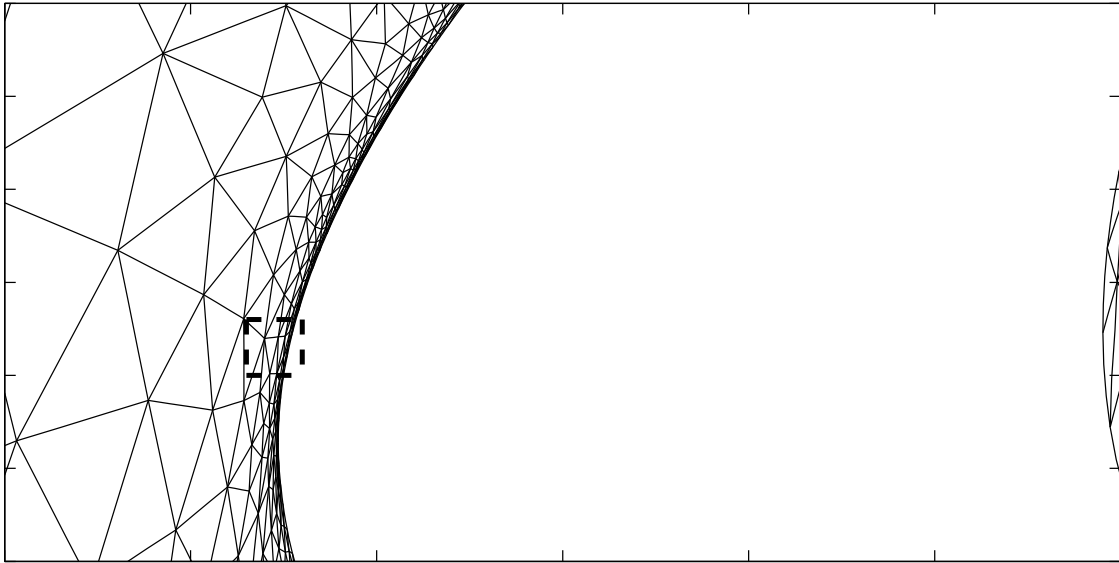


Figure 6-10: Mesh obtained by a standard adaptation iteration for flow over the EET 3-element airfoil

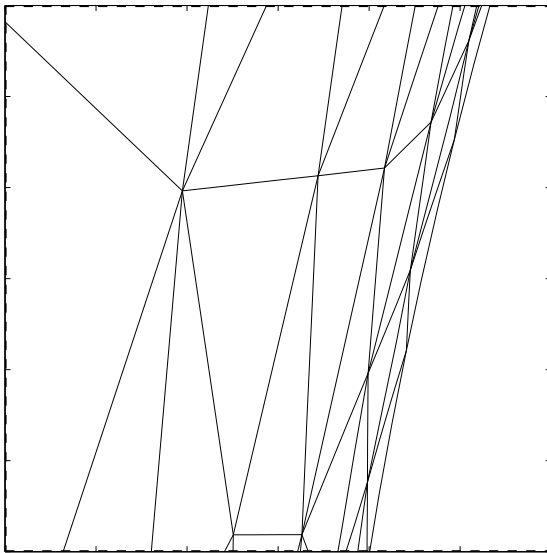
by the unsteady algorithm. However, it is significantly finer near the leading edge of the slat, as illustrated in Figure 6-11.

Figure 6-12 shows the solution obtained on the final mesh. The surface pressure coefficient is shown in Figure 6-13. For this solution, the lift coefficient is $c_\ell = 3.51$, which agrees well with experimental results and other computations [71, 5, 103]. Finally, the drag coefficient is $c_d = 436$ counts, and the error estimate is approximately 3 counts. This error estimate gives a percent error of approximately 0.7%.

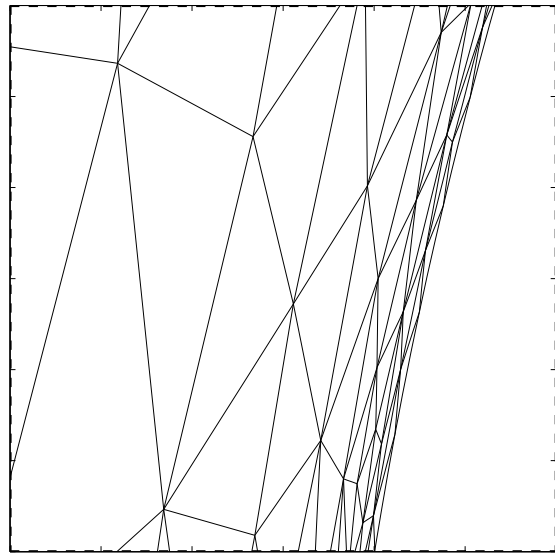
To summarize, while the unsteady procedure was not used to converge the steady solution in this case, it did enable a reasonable initial mesh to be automatically generated starting from an isotropic mesh which was too coarse for RANS simulations. This reasonable initial mesh then allowed a steady solution to be obtained and used to drive the standard adaptation algorithm. Thus, the primary benefit of the unsteady procedure for this case is the automated generation of a reasonable initial mesh. While this is an important step, future work should look to further refine and improve the algorithm.



(a) Indication of zoom region near slat leading edge

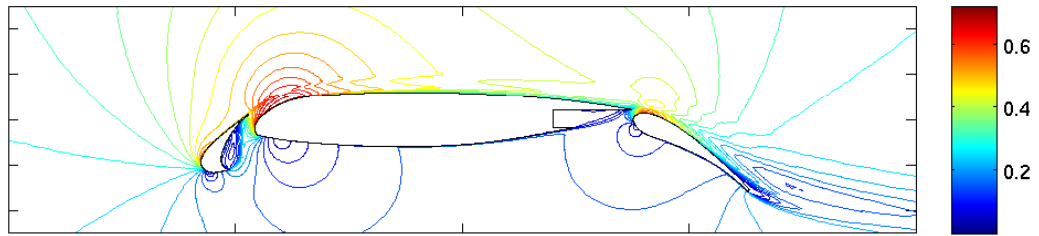


(b) Before standard adaptation

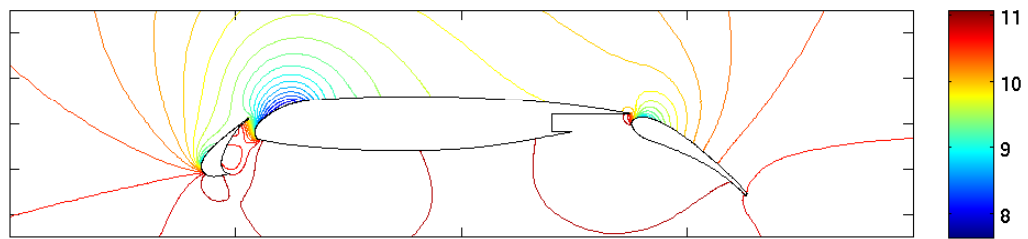


(c) After standard adaptation

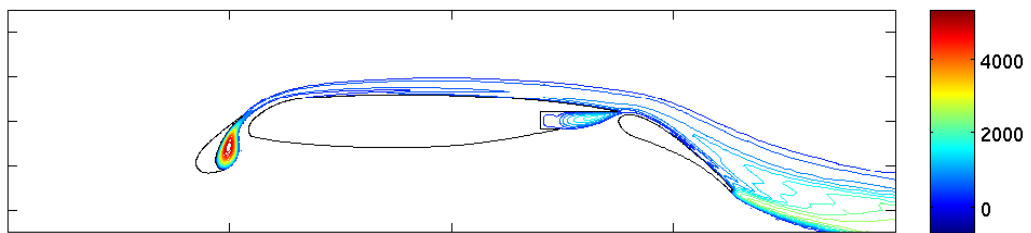
Figure 6-11: Boundary layer mesh on slat before and after standard adaptation for EET 3-element airfoil



(a) Mach number



(b) Normalized pressure ($p/\rho_\infty U_\infty^2$)



(c) Normalized SA working variable ($\tilde{\nu}/\nu_\infty$)

Figure 6-12: Finest solution ($p = 3$, 11620 elements) for the EET 3-element airfoil

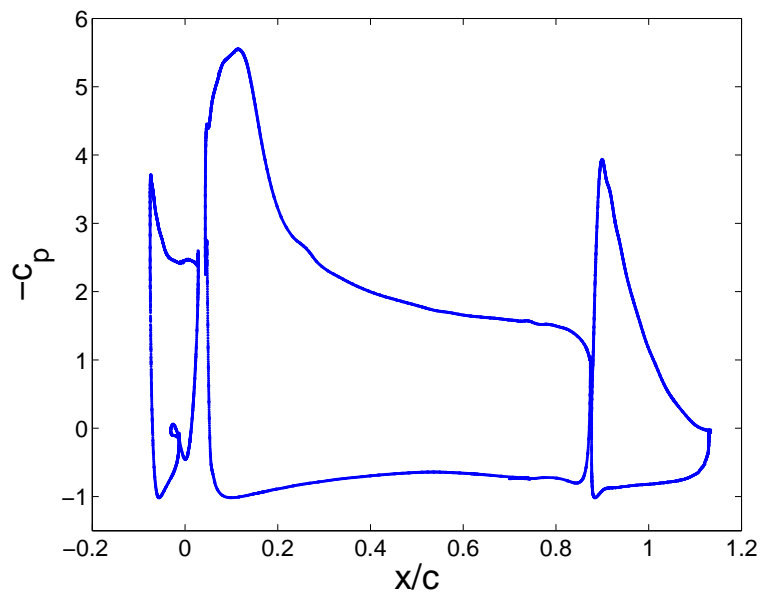


Figure 6-13: Surface pressure coefficient distribution for finest solution ($p = 3$, 11620 elements) for the EET 3-element airfoil

Chapter 7

Conclusions and Outlook

7.1 Summary and Conclusions

This thesis presents research toward a high-order, adaptive solution method for the RANS equations coupled with the SA turbulence model. The main contributions of this work are an analysis of the dual consistency of the DG discretization of the SA model source terms, an extension of output-based error estimation and adaptation to high-order discretizations of the RANS equations on curved, boundary conforming meshes, and a new, unsteady adaptation framework to eliminate the need to obtain converged steady state solutions prior to performing error estimation and mesh adaptation.

The dual consistency analysis finds that the straightforward standard weighting discretization of gradient dependent source terms results in a dual inconsistent discretization. However, a dual consistent discretization can be constructed by adding a dual consistency correction to the standard weighting scheme. Furthermore, discretizations based on the mixed formulation are shown to be asymptotically dual consistent. The impact of dual consistency on accuracy is illustrated for a scalar model problem. Specifically, the results show that the dual consistent and asymptotically dual consistent discretizations achieve optimal accuracy for the primal solution, adjoint solution, and a simple functional output. Alternatively, the standard weighting scheme produces suboptimal results for some p . For the RANS-SA system, the results for a simple flat plate test case demonstrate that the standard weighting discretization produces oscillatory adjoint solutions where the dual consistent and mixed formulation methods give smooth adjoints. These oscillations appear most clearly in the derivatives of the adjoint solution. This result agrees with the result from the model problem that the adjoint for the standard weighting discretization does not converge to the continuous adjoint in the broken H^1 norm. Furthermore, a grid refinement study for the NACA 0012 airfoil shows that, when the error is not dominated by the boundary layer edge

singularity, the dual consistent discretization is superior to the standard weighting method.

The method of Dual Weighted Residuals is applied to estimate the error in functional outputs of interest. While this method has appeared throughout the literature, to the author's knowledge, this work represents its first application to high-order DG discretizations of the RANS equations on curved meshes. The implementation here relies on an element-block Jacobi iterative procedure to generate the surrogates for the exact primal and adjoint solutions appearing in the error estimate. The results demonstrate that the error estimates are generally accurate for both the standard weighting and dual consistent discretizations. However, especially for $p = 2$, the standard weighting error estimates can be significantly smaller than the true error.

In addition to the functional output error estimate, the error estimation procedure provides an error indicator for each element in the mesh. This elemental error is used to drive a mesh adaptation procedure. Specifically, the error estimate is used to determine the size of elements desired in the adapted mesh, while the anisotropy is set by attempting to achieve equal interpolation error in all directions. This procedure has been successfully applied to multiple high Re test cases, showing that, given reasonable starting solutions, the algorithm is capable of improving the mesh and decreasing the output error. Moreover, it is observed that the $p = 2, 3$ discretizations are superior to the $p = 1$ in terms of DOF required to achieve a desired error tolerance.

Unfortunately, the requirement that a converged solution must be obtained before adapting is quite restrictive. In particular, it implies that one must converge the steady solution on the initial mesh. Of course, to minimize grid generation effort, it is beneficial to use very coarse initial meshes. Such coarse meshes can cause significant difficulty for high-order discretizations, particularly for the RANS equations. Thus, an unsteady adaptation procedure, where the need to converge steady state solutions prior to adapting is eliminated, has been developed. In this unsteady procedure, the error due to spatial discretization is estimated at every time step and used to drive the mesh adaptation algorithm. Numerical results demonstrate that this algorithm has the potential to significantly improve the robustness of the adaptation procedure when starting from very coarse initial meshes. In particular, starting from meshes that are clearly inappropriate for RANS computations, accurate steady state $p = 3$ solutions are obtained.

7.2 Future Work

Many areas for future work remain. A few ideas for further research are described here.

7.2.1 Turbulence Model Improvements

The modifications to the SA model described in Chapter 2 are designed to make the model equation stable for negative values of the working variable. However, these modifications do not address the behavior that is often the root cause of the appearance of negative values: the sharp transition in the eddy viscosity at the edge of the boundary layer. Neglecting the effect of the viscosity of the fluid, the first derivative of the turbulence model working variable is discontinuous at the boundary layer edge. This discontinuity strongly affects both the primal and adjoint solutions at the boundary layer edge. This behavior is apparent in the adjoint derivative profiles (Figures 3-8 and 3-9) shown in Chapter 3. These discontinuities at the boundary layer edge, while present in many turbulence models, are non-physical [108]. More importantly, they produce oscillations in high-order solutions that can cause the solution to diverge. Modifications to the model to increase the smoothness of the eddy viscosity in this region could lead to significant robustness improvements for high-order discretizations of the RANS-SA system.

7.2.2 Discretization Robustness Improvement

More generally, robustness for high-order RANS discretizations is a major issue. While robustness problems are sometimes tied to under-resolution at the boundary layer edge, they can also be caused by under-resolution in other flow regions where the turbulence model is not at fault, e.g. near the leading edge of an airfoil where the boundary layer is still laminar. One idea to address such under-resolved features is the addition of artificial viscosity. The use of artificial viscosity has recently been developed for shock-capturing for high-order DG discretizations of the Euler, Navier-Stokes, and RANS equations [84, 10]. In related work, it has been successfully applied to improve the robustness of RANS calculations, even for flows without shocks [76]. This technique should be further investigated for high-order discretizations of the RANS equations, particularly in combination with adaptation.

7.2.3 Standard Adaptation Algorithm Modifications

Two aspects of the adaptive procedure could be modified to improve the performance of the algorithm. As noted in Section 5.1.1, the desired anisotropy is determined by a high-order analog of a Hessian-based technique. Specifically, approximations of the $(p + 1)$ th derivatives of the Mach number are used to attempt to equidistribute the interpolation error. One problem with this technique is that the anisotropy determined by the Mach number may not be appropriate for all solution and adjoint components over the entire domain. For example, for airfoil flows with drag as the output of interest, upstream of the airfoil,

the adjoint varies rapidly in the direction normal to the stagnation streamline. Generally, the Mach number does not capture this anisotropy, potentially leading to poor resolution of the adjoint, which can adversely affect both the error estimate and output accuracy. One potential method for addressing this problem is to compute the desired metric using multiple quantities and find an appropriate intersection of the computed metrics. This idea was explored by Castro-Diaz *et al.* [21] using the primal state vector as the quantities of interest. It should be investigated using the adjoint state as well.

A second potential deficiency of the algorithm is the reliance on an assumed order of accuracy. Even when the asymptotic order of accuracy is known, it is unlikely that this order will be achieved in the initial iterations of an adaptation run, when the solution is not well resolved. In this case, the adaptation scheme will give less than the desired reduction in the error at every iteration, leading to more iterations being required to achieve the desired tolerance. A larger drawback is that the order of accuracy depends on the regularity of the exact solution, which may not be known. Or, even if it is known, the elements affected by the singularities must be determined from the discrete solution. Thus, it would be beneficial to remove the order of accuracy from the adaptive algorithm. One appealing approach is to entirely avoid the metric specification step, instead driving the mesh adaptation with error directly, as described by Park and Darmofal [80].

7.2.4 Unsteady Adaptation Refinements

The unsteady adaptation algorithm presented here is but a beginning. Many issues remain to be resolved. First, the cases shown in Chapter 6 were started from a uniform flow initial condition. While these cases were successful, this initial condition causes a very violent transient that the solver must march through. Physically realistic initial conditions should be explored. Second, the tolerances used here were set at the levels desired for the final, steady state result. Again, while this setting worked in this case, it is probably not the most efficient selection. One obvious option is to set the error tolerance to some percentage of the computed output. This tolerance may allow the algorithm to march through the initial transient using fewer DOF, thus decreasing the total cost. Third, the stopping criterion should be improved. In the cases shown, a significant fraction of the total cost is allotted to fully solving each time step and estimating the error even after the output is changing very little and the adaptation has stopped. This time might be better spent simply obtaining the steady state solution. Fourth, only backward Euler time discretization has been considered. The method should be extended to high-order time discretizations. With high-order discretization in time, it may be possible to move through any transients

more quickly and reliably. Furthermore, this extension may be viewed as the first step toward a space-time adaptation algorithm for truly unsteady problems, where high-order time discretization will be desirable.

Appendix A

Derivation of the RANS Equations

The compressible Navier-Stokes equations in conservation form are given by

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0, \quad (\text{A.1})$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_j u_i + p \delta_{ji}) = \frac{\partial \tau_{ji}}{\partial x_j}, \quad (\text{A.2})$$

$$\frac{\partial}{\partial t} \left[\rho \left(e + \frac{1}{2} u_i u_i \right) \right] + \frac{\partial}{\partial x_j} \left[\rho u_j \left(h + \frac{1}{2} u_i u_i \right) \right] = \frac{\partial}{\partial x_j} (u_i \tau_{ij}) - \frac{\partial q_j}{\partial x_j}, \quad (\text{A.3})$$

where t denotes time, x_i is position, ρ is density, u_i is velocity, p denotes pressure, e is the internal energy per unit mass, h is the enthalpy per unit mass, τ_{ij} is the viscous stress tensor, q_i is the heat flux vector, and δ_{ij} is the Kronecker delta. For Newtonian fluids, the viscous stress tensor is given by

$$\tau_{ij} = 2\mu \left(s_{ij} - \frac{1}{3} \frac{\partial u_k}{\partial x_k} \delta_{ij} \right),$$

where $s_{ij} = \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$ is the strain rate tensor, μ is the dynamic viscosity, and the second viscosity is assumed to be $-\frac{2}{3}\mu$. Furthermore, the heat flux vector is given by Fourier's law:

$$q_j = -\kappa \frac{\partial T}{\partial x_j},$$

where T is the temperature and κ is the thermal conductivity of the fluid.

Finally, it is assumed that the fluid is an ideal gas with constant specific heats. Thus,

$$p = \rho R T, \quad e = c_v T, \quad h = c_p T,$$

where R is the ideal gas constant, c_v is the specific heat at constant volume, and c_p is the specific heat at constant pressure.

While (A.1) through (A.3) govern the motion of fluids in both laminar and turbulent regimes, the large range of spatial and temporal scales present in turbulent flows make direct solution of the compressible Navier-Stokes equations prohibitively expensive for the majority of flows of engineering interest [37, 74]. Thus, it is common to instead solve the RANS equations, which govern the turbulent mean flow.

Following Reynolds [91], the flow variables are written in terms of mean and fluctuating parts. Then, equations governing the mean flow are derived by averaging the Navier-Stokes equations. A summary of the relevant definitions and resulting equations is given here. For more details, see [108].

Given a flow variable, $v(\mathbf{x}, t)$, define the time average, $\bar{v}(\mathbf{x})$, by

$$\bar{v}(\mathbf{x}) \equiv \lim_{T \rightarrow \infty} \int_{t_0}^{t_0+T} v(\mathbf{x}, t) dt.$$

Then, the flow variables can be decomposed into mean and fluctuating parts. For example, the density can be written

$$\rho = \bar{\rho} + \rho',$$

where the fluctuating part, ρ' , is defined by this relationship.

In addition, for compressible flows, it is convenient to define a density-weighted time average. The density-weighted average of a quantity $v(\mathbf{x}, t)$ is denoted by $\tilde{v}(\mathbf{x})$, where

$$\tilde{v}(\mathbf{x}) \equiv \frac{1}{\bar{\rho}} \lim_{T \rightarrow \infty} \int_{t_0}^{t_0+T} \rho(\mathbf{x}, t) v(\mathbf{x}, t) dt.$$

This averaging procedure is known as Favre averaging. As with the conventional average, the density-weighted average allows the flow variables to be written in terms of mean and fluctuating parts. For example,

$$u_i = \tilde{u}_i + u_i''.$$

Applying the time averaging procedure to the compressible Navier-Stokes equations, one obtains the following equations governing the mean flow:

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial}{\partial x_i} (\bar{\rho} \tilde{u}_i) = 0, \quad (\text{A.4})$$

$$\frac{\partial}{\partial t} (\bar{\rho} \tilde{u}_i) + \frac{\partial}{\partial x_j} (\bar{\rho} \tilde{u}_j \tilde{u}_i + \bar{p} \delta_{ji}) = \frac{\partial}{\partial x_j} (\bar{\tau}_{ji} - \overline{\rho u_j'' u_i''}), \quad (\text{A.5})$$

$$\begin{aligned} & \frac{\partial}{\partial t} \left[\bar{\rho} \left(\tilde{e} + \frac{1}{2} \tilde{u}_i \tilde{u}_i \right) + \frac{1}{2} \overline{\rho u_i'' u_i''} \right] + \frac{\partial}{\partial x_j} \left[\bar{\rho} \tilde{u}_j \left(\tilde{h} + \frac{1}{2} \tilde{u}_i \tilde{u}_i \right) + \tilde{u}_j \frac{1}{2} \overline{\rho u_i'' u_i''} \right] \\ & = \frac{\partial}{\partial x_j} \left[-\bar{q}_j - \overline{\rho u_j'' h''} + \overline{\tau_{ji} u_i''} - \overline{\rho u_j'' \frac{1}{2} u_i'' u_i''} \right] + \frac{\partial}{\partial x_j} \left[\tilde{u}_i \left(\bar{\tau}_{ij} - \overline{\rho u_i'' u_j''} \right) \right]. \quad (\text{A.6}) \end{aligned}$$

Equations (A.4) through (A.6) contain multiple correlation terms—e.g. $\overline{\rho u_j'' u_i''}$ in the momentum equation—that do not appear in (A.1) through (A.3) and are not known in terms of the mean flow variables. Thus, there are more unknowns than equations, which implies that the system is not closed. To close the system, it is necessary to relate these correlation terms to the mean flow via closure approximations.

In general, one must provide closure approximations for five terms: the Reynolds stress tensor, $\overline{\rho u_j'' u_i''}$; the turbulent kinetic energy per unit volume, $\bar{\rho}k \equiv \frac{1}{2}\overline{\rho u_i'' u_i''}$; the turbulent heat flux vector, $\overline{\rho u_j'' h''}$; the molecular diffusion term, $\overline{\tau_{ji} u_i''}$; and the turbulent transport term, $\overline{\rho u_j'' \frac{1}{2} u_i'' u_j''}$. In this work, the turbulent kinetic energy, molecular diffusion, and turbulent transport terms are ignored. This approximation is reasonable whenever $k \ll \tilde{h}$, which holds for most engineering flows into the supersonic regime [108]. Thus, only two closure approximations are required. For the Reynolds stress tensor, the Boussinesq approximation is applied. The Boussinesq approximation relates the Reynolds stress to the mean flow viscous stress tensor:

$$-\overline{\rho u_j'' u_i''} = 2\mu_t \left(\tilde{s}_{ji} - \frac{1}{3} \frac{\partial \tilde{u}_k}{\partial x_k} \delta_{ji} \right) - \frac{2}{3} \bar{\rho} k \delta_{ji}, \quad (\text{A.7})$$

where μ_t is the eddy viscosity. The addition of $-2\bar{\rho}k\delta_{ji}/3$ is made to guarantee that the trace of the Reynolds stress is $-2\bar{\rho}k$. However, in keeping with the assumption that the turbulent kinetic energy is small compared to the mean flow enthalpy, this term is neglected here.

Then, following the Reynolds analogy, the turbulent heat flux vector is given by

$$\overline{\rho u_j'' h''} = -\frac{\mu_t c_p}{Pr_t} \frac{\partial \tilde{T}}{\partial x_j}, \quad (\text{A.8})$$

where Pr_t denotes the turbulent Prandtl number, which is taken to be $Pr_t = 0.9$.

Appendix B

Laminar NACA 0012 Results

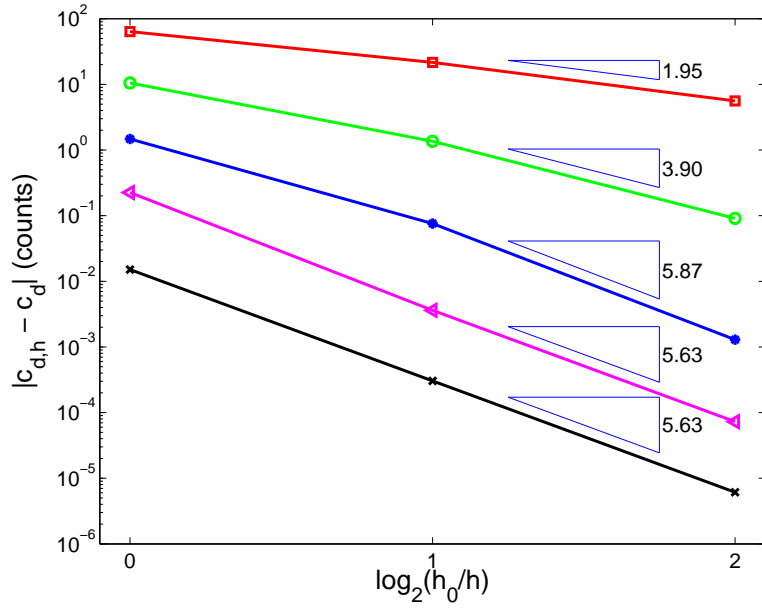
The test case is $M_\infty = 0.25$, $Re_c = 1 \times 10^4$, $\alpha = 0$ flow over a NACA 0012 airfoil. The computational domain and meshes used for this case are described in detail in Section 3.4.2.

Figure B-1 shows the drag error results. The error is computed relative to an “exact” drag value, which is computed by extrapolating the $p = 5$ drag results, assuming that the error convergence rate is the same for both refinements. The resulting drag value is 375.2615065 counts.

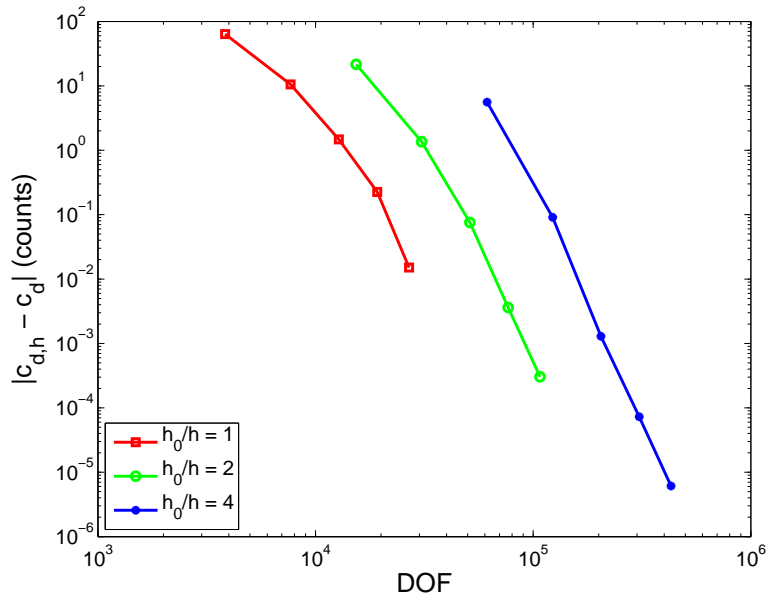
The figure shows that $p = 1, 2, 3$ achieve nearly optimal order of accuracy—i.e. h^{2p} —for the second mesh refinement. While $p = 4, 5$ are not achieving their optimal rates, there is no evidence that the accuracy will asymptote to $O(h^3)$ as in the RANS-SA case. Furthermore, by examining the elemental error estimates, it is possible to confirm that the boundary layer edge region is not the source of the suboptimal $p = 4, 5$ results. Figure B-2 shows the near-airfoil $p = 4$ elemental error estimates on the coarse and fine meshes. The figure shows that on the coarse mesh, the error is dominated by a few elements near the edge of the boundary layer and wake. However, as expected for smooth solutions, these errors are effectively decreased by h refinement with the high-order discretization, and these regions do not appear to contribute significantly to the error on the fine mesh.

The $p = 4$ elemental error estimates for entire domain are shown in Figure B-3. As expected, the coarse mesh errors in the outer regions of the domain are small compared to those near the airfoil. However, for the fine mesh, the error is largest on the outer boundary at the top and bottom points of the circle. These points are where the boundary condition switches from inflow to outflow conditions. Thus, it appears that it is the boundary conditions which cause the loss of accuracy for $p = 4, 5$ in this case.

To summarize, given that this case is run using the same meshes as the turbulent NACA 0012 case, it reinforces the argument that the RANS-SA solution behavior at the boundary layer edge is responsible for the suboptimal accuracy observed in Section 3.4.2.

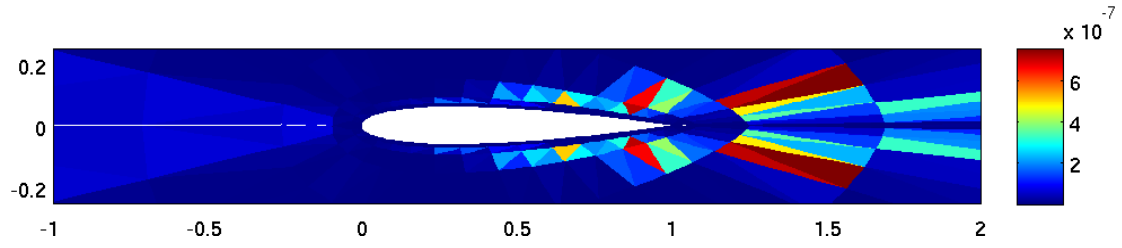


(a) Drag error versus h for $p = 1$ through $p = 5$ ($p = 1 : \square$, $p = 2 : \circ$, $p = 3 : *$, $p = 4 : \triangleleft$, $p = 5 : \times$)

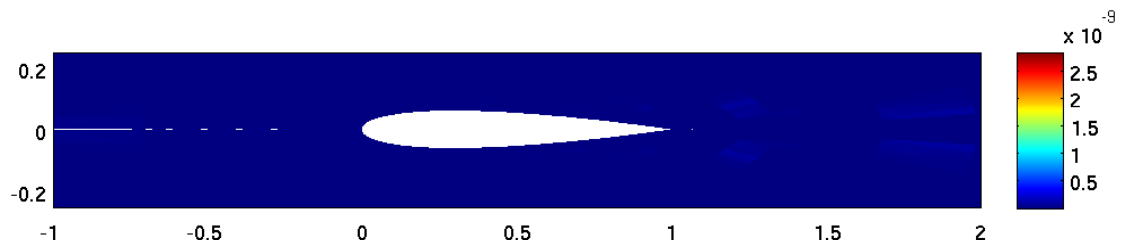


(b) Drag error versus DOF

Figure B-1: Drag error for laminar flow over a NACA 0012

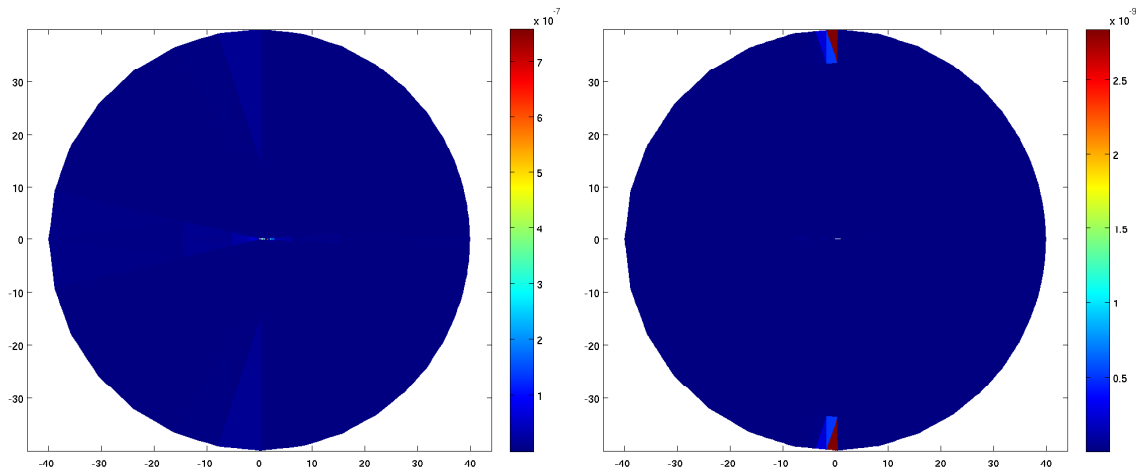


(a) Coarse mesh



(b) Fine mesh

Figure B-2: Elemental error estimate for the $p = 4$ discretization on the coarse and fine meshes for the laminar NACA 0012 test case in the near field



(a) Coarse mesh

(b) Fine mesh

Figure B-3: Elemental error estimate for the $p = 4$ discretization on the coarse and fine meshes for the laminar NACA 0012 test case for the entire domain

Appendix C

A Priori Output Error Estimation

This appendix provides *a priori* error bounds for functional outputs computed from finite element solutions. The analysis demonstrates the importance of dual consistency for accurately computing functional outputs. Furthermore, the bounds obtained are used to motivate the convergence rates used by the mesh optimization algorithm discussed in Section 5.1.2. For simplicity, a linear problem and linear functional output are considered first. Then, the analysis is extended to the nonlinear case.

C.1 Linear Analysis

Consider the following continuous primal problem: compute $\mathcal{J}(u)$ where $u \in \mathcal{V}$ satisfies

$$B(u, v) = \ell(v), \quad \forall v \in \mathcal{V}.$$

Here, $B(\cdot, \cdot)$ is a bilinear functional and ℓ , and \mathcal{J} are linear functionals. The corresponding dual problem is given by the following: find $\psi \in \mathcal{V}$ such that

$$B(v, \psi) = \mathcal{J}(v), \quad \forall v \in \mathcal{V}.$$

Assume that both the continuous primal and dual problems are well-posed.

Furthermore, consider the following discretization of the primal problem: compute $\mathcal{J}(u_h)$ where $u_h \in \mathcal{V}_h$ satisfies

$$B_h(u_h, v_h) = \ell(v_h), \quad \forall v_h \in \mathcal{V}_h,$$

where \mathcal{V}_h is an appropriate finite dimensional space. Then, the discrete dual problem is as

follows: find $\psi_h \in \mathcal{V}_h$ such that

$$B_h(v_h, \psi_h) = \mathcal{J}(v_h), \quad \forall v_h \in \mathcal{V}_h.$$

As with the continuous problems, assume that both the discrete primal and dual problems are well-posed.

Then, using the definitions of the continuous and discrete dual problems, the error in the computed functional output can be written as

$$\mathcal{J}(u) - \mathcal{J}(u_h) = B(u, \psi) - B_h(u_h, \psi_h).$$

To continue, assume that, for the exact solutions u and ψ , $B(u, \psi) = B_h(u, \psi)$. Then,

$$\begin{aligned} \mathcal{J}(u) - \mathcal{J}(u_h) &= B_h(u, \psi) - B_h(u_h, \psi_h) \\ &= B_h(u, \psi) - B_h(u_h, \psi) + B_h(u_h, \psi) - B_h(u_h, \psi_h) \\ &= B_h(u - u_h, \psi) + B(u_h, \psi - \psi_h) \\ &= B_h(u - u_h, \psi) - B_h(u - u_h, \psi_h) + B_h(u - u_h, \psi_h) + B(u_h, \psi - \psi_h) \\ &= B_h(u - u_h, \psi - \psi_h) + B_h(u - u_h, \psi_h) + B_h(u_h, \psi - \psi_h). \end{aligned}$$

Thus, if the discretization is consistent and dual consistent,

$$\mathcal{J}(u) - \mathcal{J}(u_h) = B_h(u - u_h, \psi - \psi_h).$$

Finally, if the bilinear form B_h is continuous,

$$|\mathcal{J}(u) - \mathcal{J}(u_h)| \leq C \|u - u_h\| \|\psi - \psi_h\|,$$

where $\|\cdot\|$ is an appropriate energy norm. Assuming that the discretization achieves $\|u - u_h\| \leq Ch^s$ and $\|\psi - \psi_h\| \leq Ch^t$,

$$|\mathcal{J}(u) - \mathcal{J}(u_h)| \leq Ch^{s+t}.$$

For DG discretizations of hyperbolic problems, assuming the exact solutions u and ψ are sufficiently regular, $r = s = p + 1/2$ (see Johnson and Pitkaranta [63]). Thus,

$$|\mathcal{J}(u) - \mathcal{J}(u_h)| \leq Ch^{2p+1}.$$

For elliptic problems, assuming the exact solution is sufficiently regular, the BR2 scheme,

as well as many other DG methods, achieve $r = s = p$ (see Arnold *et al.* [7]). Thus,

$$|\mathcal{J}(u) - \mathcal{J}(u_h)| \leq Ch^{2p}.$$

C.2 Nonlinear Analysis

Consider the following continuous primal problem: compute $\mathcal{J}(u)$ where $u \in \mathcal{V}$ satisfies

$$R(u, v) = \ell(v), \quad \forall v \in \mathcal{V},$$

where $R(\cdot, \cdot)$ is a semi-linear (linear in the second argument) functional, ℓ is a linear functional, and \mathcal{J} is a nonlinear functional. The corresponding dual problem is given by the following: find $\psi \in \mathcal{V}$ such that

$$R'[u](v, \psi) = \mathcal{J}'[u](v), \quad \forall v \in \mathcal{V}.$$

Assume that both the continuous primal and dual problems are well-posed.

Furthermore, consider the following discretization of the primal problem: compute $\mathcal{J}(u_h)$ where $u_h \in \mathcal{V}_h$ satisfies

$$R_h(u_h, v_h) = \ell(v_h), \quad \forall v_h \in \mathcal{V}_h.$$

Then, define the discrete dual problem is as follows: find $\psi_h \in \mathcal{V}_h$ such that

$$R'_h[u](v_h, \psi_h) = \mathcal{J}'[u](v_h), \quad \forall v_h \in \mathcal{V}_h.$$

As with the continuous problems, assume that both the discrete primal and dual problems are well-posed.

The central idea of the nonlinear analysis shown here is the examination of Taylor series expansions of the output and residual forms about the exact solution, u . In particular, consider the Taylor series expansion of the function output:

$$\mathcal{J}(u_h) = \mathcal{J}(u) + \mathcal{J}'[u](u_h - u) + r_{out}(u, u_h),$$

where

$$r_{out}(u, u_h) = \int_0^1 \mathcal{J}''[u + \theta(u_h - u)](u_h - u, u_h - u)(1 - \theta) d\theta.$$

For the conditions required for this statement to be valid, see [47].

Assuming that the bilinear functional $\mathcal{J}''[u + \theta(u_h - u)](\cdot, \cdot)$ is continuous for all $\theta \in (0, 1)$,

$$|r_{out}(u, u_h)| \leq \int_0^1 |\mathcal{J}''[u + \theta(u_h - u)](u_h - u, u_h - u)(1 - \theta)| d\theta \leq C \|u_h - u\|^2.$$

Thus,

$$|\mathcal{J}(u_h) - \mathcal{J}(u)| \leq |\mathcal{J}'[u](u_h - u)| + C \|u_h - u\|^2.$$

To complete the analysis, it is necessary to analyze the linear term, $\mathcal{J}'[u](u_h - u)$. By the definitions of the dual problems,

$$\mathcal{J}'[u](u_h - u) = R'_h[u](u_h, \psi_h) - R'[u](u, \psi).$$

Assuming that, for the exact solutions u and ψ , $R'[u](u, \psi) = R'_h[u](u, \psi)$, gives

$$\begin{aligned} \mathcal{J}'[u](u - u_h) &= R'_h[u](u, \psi) - R'_h[u](u_h, \psi_h) \\ &= R'_h[u](u, \psi) - R'_h[u](u_h, \psi) + R'_h[u](u_h, \psi) - R'_h[u](u_h, \psi_h) \\ &= R'_h[u](u - u_h, \psi) + R'_h[u](u_h, \psi - \psi_h) \\ &= R'_h[u](u - u_h, \psi) - R'_h[u](u - u_h, \psi_h) + R'_h[u](u - u_h, \psi_h) + R'_h[u](u_h, \psi - \psi_h) \\ &= R'_h[u](u - u_h, \psi - \psi_h) + R'_h[u](u - u_h, \psi_h) + R'_h[u](u_h, \psi - \psi_h). \end{aligned}$$

If the scheme is dual consistent, $R'_h[u](u_h, \psi - \psi_h) = 0$. The term $R'_h[u](u - u_h, \psi_h)$ corresponds to the consistency error term from the linear analysis. However, unlike the linear case, this term is not zero, even for a consistent discretization. To bound this term, begin by writing a Taylor series for the functional $R_h(\cdot, \psi_h)$ about u . That is,

$$R_h(u_h, \psi_h) = R_h(u, \psi_h) + R'_h[u](u_h - u, \psi_h) + r_{res}(u, u_h),$$

where

$$r_{res}(u, u_h) = \int_0^1 R''_h[u + \theta(u_h - u)](u_h - u, u_h - u, \psi_h)(1 - \theta) d\theta.$$

Thus, assuming that tri-linear functional $R''_h[u + \theta(u_h - u)](\cdot, \cdot, \cdot)$ is continuous for all $\theta \in (0, 1)$,

$$r_{res}(u, u_h) \leq \int_0^1 |R''_h[u + \theta(u_h - u)](u_h - u, u_h - u, \psi_h)| d\theta \leq C \|u_h - u\|^2 \|\psi_h\|.$$

If the discretization is consistent, $R_h(u_h, \psi_h) - R_h(u, \psi_h) = 0$. Thus,

$$|R'_h[u](u_h - u, \psi_h)| \leq C \|u_h - u\|^2 \|\psi_h\|.$$

Further assuming that the bilinear form $R'_h[u](\cdot, \cdot)$ is continuous,

$$|\mathcal{J}'[u](u - u_h)| \leq C \|u_h - u\| (\|\psi_h - \psi\| + \|u_h - u\|).$$

Thus,

$$|\mathcal{J}(u_h) - \mathcal{J}(u)| \leq C \|u_h - u\| (\|\psi_h - \psi\| + \|u_h - u\|).$$

Finally, if the scheme achieves $\|u_h - u\| \leq Ch^s$ and $\|\psi_h - \psi\| \leq Ch^t$

$$\mathcal{J}'[u](u - u_h) \leq Ch^{\min(s+t, 2s)}.$$

Appendix D

Error Convergence Rates for Adaptation

The results shown in Chapter 5 demonstrate that the $p = 2, 3$ adaptive discretizations are superior to the $p = 1$ in terms of DOF required to achieve a given accuracy. However, the results should also be compared to the optimal adaptive performance. The optimal performance of the adaptive scheme is not immediately obvious. Thus, to get an idea of what to expect, two different regimes are considered.

In the first regime, all flow features are relatively well resolved. Specifically, assume that the error is approximately equidistributed throughout the domain and that the mesh anisotropy is appropriate—i.e. there are no regions of the mesh where decreasing the mesh spacing in one direction is significantly more effective at driving down the error than decreasing the spacing in an orthogonal direction. Then, adaptation and global uniform refinement should provide the same error reduction with DOF. Thus, assuming that the optimal order of accuracy for the output of interest is $O(h^r)$, and noting that, for isotropic refinement, $N_{dof} \propto h^{-n}$, where n is the dimension of the domain, the expected convergence rate is given by

$$E \propto N_{dof}^{-r/n}.$$

In the second regime, assume that the output error is dominated by error in a particular region of the mesh. In this case, one would expect the error reduction with DOF to be dictated by the details of the region where the error is large. For example, if the total error and DOF in the mesh are dominated by some anisotropic flow feature—e.g. a shock or a turbulent boundary layer—it is likely that the grid spacing normal to this feature is too large. Thus, one might expect that the error would achieve the optimal rate with respect to the grid spacing normal to the feature. That is, if h_1 is the grid spacing normal to the

feature, $E \propto h_1^r$. Then, if the mesh spacing is decreased only in this direction, $N_{dof} \propto h_1^{-1}$. Thus,

$$E \propto N_{dof}^{-r}.$$

For shock-free, attached flow, RANS calculations on coarse meshes—in particular meshes where the near wall spacing is too large—one expects the drag error to be dominated by errors in the boundary layer and, more specifically, errors in the laminar sublayer. Furthermore, the solution in the laminar sublayer is smooth and anisotropic. Thus, one might expect $E \propto N_{dof}^{-r}$. However, given that $E \propto h^r$ is an asymptotic convergence rate, it is not clear that this rate will be achieved until the boundary layer is at least somewhat well resolved. This fact may lead to suboptimal convergence. On the other hand, if other regions of the flow are over-resolved, the adaptation algorithm will coarsen the mesh in these regions. This subtraction of DOF in other regions may lead to better than expected performance in terms of error versus total DOF.

Once the near wall solution is well resolved, the error may be dominated by errors in the boundary layer edge region. Given that, for the grid resolution typical in engineering calculations, the solution in this region is singular, optimal convergence in this regime should not be expected.

To examine the error reduction versus DOF obtained by the adaptation algorithm, results for the ellipse and NACA 0012 test cases are shown in Tables D.1 through D.3. In the tables, the column *Rate* refers to the convergence rate of the error with DOF—i.e. $E \propto N_{dof}^{Rate}$. The notation *Rate* = NA indicates that either the total DOF decreased or the error increased at that iteration. A star (*) in the column labeled “BL Edge” indicates that the error estimate in the boundary layer edge region contributes significantly to the total error estimate. Specifically, if there is an element in the boundary layer edge region where the elemental error estimate is equal to or greater than one half of the maximum elemental error estimate in the mesh, then (*) appears in the BL Edge column. This data is included to give a rough feel for whether the boundary layer edge is playing an important role in the total error.

For $p = 1$, the adaptation generally gives better than the rate expected for isotropic refinement, and the boundary layer edge region is never a significant contributor to the total error. For $p = 2, 3$ it is difficult to draw any firm conclusions. In general, it appears that the initial adaptation iterations perform well in that they are able to decrease the error or hold it fixed while decreasing the total DOF. In this regime, the elemental error estimates are largest at the wall, and DOF are added near the wall. The decrease in total DOF is due to coarsening of other regions of the domain that initially contain more resolution

than necessary. As the adaptation progresses, the boundary layer edge begins to play a significant role. However, it is not clear that this fact is responsible for the degradation of the convergence rate observed in both the ellipse and NACA 0012, $Re_c = 1 \times 10^6$ cases. Finally, it should be pointed out that the most consistent result in terms of the observed convergence rate is that for the $p = 3$, NACA 0012 at $Re_c = 1 \times 10^7$, where the BAMG grid spacing ratio was decreased to provide a smoother grid.

Table D.1: Error convergence versus DOF for the ellipse test case

p	Adapt Iter	N_{dof}	Error (counts)	Rate	BL Edge
1	0	4800	3.16e1	—	
1	1	13605	8.69e0	-1.238	
1	2	21999	3.03e-2	-3.501	
2	0	9600	2.478e1	—	
2	1	6078	9.870e0	NA	
2	2	5928	2.871e-2	NA	
2	3	7926	1.828e-1	NA	*
2	4	14904	6.140e-3	-5.374	
2	5	31074	1.135e-3	-2.297	*
3	0	16000	1.315e1	—	
3	1	8400	5.164e0	NA	
3	2	4450	7.686e0	NA	
3	3	4740	1.755e-1	NA	
3	4	5220	5.583e-1	NA	
3	5	6280	2.133e-2	-17.659	
3	6	9470	3.160e-3	-4.649	
3	7	14310	2.766e-3	-0.323	
3	8	21880	1.122e-3	-2.126	*

Given that the results do not agree with the expected optimal rates and do not support any obvious conclusion, further work is required to more fully understand the performance of the algorithm in this respect.

Table D.2: Error convergence versus DOF for the NACA 0012 ($Re_c = 1 \times 10^6$) test case

p	Adapt Iter	N_{dof}	Error (counts)	r	BL Edge
1	0	4428	8.734e1	—	
1	1	8115	7.467e0	-4.060	
1	2	28533	4.113e-2	-4.137	
2	0	8856	1.964e1	—	
2	1	5670	1.171e0	NA	
2	2	10536	3.064e-1	-2.164	
2	3	27168	9.569e-3	-3.660	*
2	4	75846	8.980e-4	-2.305	*
3	0	14760	6.375e0	—	
3	1	6740	1.689e0	NA	
3	2	8710	1.176e0	-1.411	
3	3	12110	7.501e-2	-8.351	
3	4	21460	8.155e-4	-7.903	*
3	5	40820	7.803e-4	-0.069	*

Table D.3: Error convergence versus DOF for the NACA 0012 ($Re_c = 1 \times 10^7$) test case

p	Adapt Iter	N_{dof}	Error (counts)	r	BL Edge
1	0	6438	5.872e1	—	
1	1	19659	8.288e0	-1.754	
1	2	56664	2.430e-1	-3.334	
2	0	12876	3.054e1	—	
2	1	12774	1.512e0	NA	
2	2	20562	1.948e-1	-4.304	
2	3	55884	2.874e-2	-1.914	*
2	4	152082	1.059e-3	-3.298	*
3	0	21460	5.437e0	—	
3	1	30530	1.471e0	-3.708	
3	2	46490	1.661e-1	-5.187	
3	3	76380	1.748e-2	-4.535	
3	4	126170	1.676e-3	-4.672	*

Appendix E

Boundary Conditions

This appendix details the boundary conditions used in this work. All boundary conditions are enforced weakly through the fluxes, \mathcal{F}^b and \mathcal{F}_v^b , across the domain boundary. In general, these fluxes are computed by evaluating the flux using boundary state and state gradient vectors that depend on both the interior state and gradient and the boundary condition information. The calculation of the boundary state and state gradient as well as departures from this framework are described for various boundary condition options below.

E.1 Farfield, Full State Boundary

At a farfield “full state” boundary, the boundary state vector, \mathbf{u}^b , is specified by the user. Then, the inviscid flux, \mathcal{F}^b , is computed using the Roe flux:

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

The viscous flux is calculated using the user specified boundary state and the state gradient evaluated from the interior:

$$\mathcal{F}_v^b = \mathcal{A}(\mathbf{u}^b) \nabla \mathbf{u}_h^+.$$

As shown by Lu [72], the specification of the inviscid flux in this manner renders the discretization dual inconsistent. Thus, this boundary condition is only used for farfield boundaries where this dual inconsistency should have little impact on the flow solution.

E.2 Subsonic Inflow: T_t , p_t , α , $\rho\tilde{v}$

The subsonic inflow conditions are set using Riemann invariants. For 2-D flows, at subsonic inflow boundaries, analysis of the convective terms of the RANS-SA system shows that there

are four incoming and one outgoing Riemann invariants. Thus, for convection dominated flow, to appropriately specify a subsonic inflow condition, four quantities must be specified. In this work, the total temperature, T_t , total pressure, p_t , inflow angle, α , and turbulence working variable $\rho\tilde{\nu}$ are specified. This information is combined with the outgoing Riemann invariant to compute the boundary state. Then, the inviscid flux, \mathcal{F}^b , is given by

$$\mathcal{F}^b = \mathcal{F}(\mathbf{u}^b).$$

The viscous flux is calculated using boundary state and the state gradient evaluated from the interior,

$$\mathcal{F}_v^b = \mathcal{A}(\mathbf{u}^b)\nabla\mathbf{u}_h^+.$$

E.3 Subsonic Outflow: p

The subsonic outflow boundary condition is also specified using Riemann invariants. In this case, there is one incoming invariant and four outgoing. Thus, to appropriately specify a subsonic outflow condition, one quantity must be specified. In this work, the static pressure, p , is specified. This quantity is combined with the outgoing Riemann invariants to compute the boundary state. Then, the inviscid flux, \mathcal{F}^b , is given by

$$\mathcal{F}^b = \mathcal{F}(\mathbf{u}^b).$$

The viscous flux is calculated using boundary state and the state gradient evaluated from the interior,

$$\mathcal{F}_v^b = \mathcal{A}(\mathbf{u}^b)\nabla\mathbf{u}_h^+.$$

E.4 No Slip, Adiabatic Wall

At a no slip, adiabatic wall, the desired conditions are zero velocity, heat flux, and eddy viscosity at the wall. Using the this information, the boundary state is given by

$$\mathbf{u}^b = [\rho^+, 0, 0, \rho E^+, 0]^T,$$

and the inviscid flux, \mathcal{F}^b , is given by

$$\mathcal{F}^b = \mathcal{F}(\mathbf{u}^b).$$

When combined with the no slip condition, the adiabatic condition requires that the

viscous flux in the energy equation is zero. Thus, the viscous flux is computed using the interior gradient, except for the energy equation component, which is set to zero. In particular,

$$\left(\mathcal{F}_v^b - \eta_f \tilde{\mathbf{r}}_f^b(\mathbf{u}_h)\right) \cdot \vec{n}^+ = \begin{bmatrix} \left(\mathcal{A}(\mathbf{u}^b) \nabla \mathbf{u}_h^+ - \eta_f \tilde{\mathbf{r}}_f^b(\mathbf{u}_h)\right)_1 \cdot \vec{n}^+ \\ \left(\mathcal{A}(\mathbf{u}^b) \nabla \mathbf{u}_h^+ - \eta_f \tilde{\mathbf{r}}_f^b(\mathbf{u}_h)\right)_2 \cdot \vec{n}^+ \\ \left(\mathcal{A}(\mathbf{u}^b) \nabla \mathbf{u}_h^+ - \eta_f \tilde{\mathbf{r}}_f^b(\mathbf{u}_h)\right)_3 \cdot \vec{n}^+ \\ 0 \\ \left(\mathcal{A}(\mathbf{u}^b) \nabla \mathbf{u}_h^+ - \eta_f \tilde{\mathbf{r}}_f^b(\mathbf{u}_h)\right)_5 \cdot \vec{n}^+ \end{bmatrix}.$$

E.5 Symmetry Plane

The symmetry plane boundary condition requires that the state be symmetric about the plane and that its derivatives normal to the plane are continuous. These conditions require that the flow is tangent to the plane. Thus, the boundary state is set to the interior state with the normal velocity subtracted:

$$\begin{aligned} \rho^b &= \rho^+, \\ \rho u_1^b &= \rho u_1^+ - (\rho u_1^+ n_1 + \rho u_2^+ n_2) n_1, \\ \rho u_2^b &= \rho u_2^+ - (\rho u_1^+ n_1 + \rho u_2^+ n_2) n_2, \\ \rho E^b &= \rho E^+, \\ \rho \tilde{v}^b &= \rho \tilde{v}^+. \end{aligned}$$

Then, $\mathcal{F}^b = \mathcal{F}(\mathbf{u}^b)$.

For the state derivatives, the symmetry plane implies that the normal derivatives of scalar quantities—i.e. ρ , ρE , and $\rho \tilde{v}$ —are zero. The normal derivative of the tangential velocity is zero as well. The condition gives no information about the normal derivative of the normal velocity or the tangential derivative of any quantity. The boundary state gradient is set using the given information about the normal derivatives combined with

interior derivatives. Thus, for the density,

$$\begin{aligned}
\begin{bmatrix} \frac{\partial \rho}{\partial x_1} \\ \frac{\partial \rho}{\partial x_2} \end{bmatrix}^b &= \begin{bmatrix} n_1 & n_2 \\ -n_2 & n_1 \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial \rho}{\partial n} \\ \frac{\partial \rho}{\partial t} \end{bmatrix}^b \\
&= \begin{bmatrix} n_1 & n_2 \\ -n_2 & n_1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \left(\frac{\partial \rho}{\partial t}\right)^b \end{bmatrix} \\
&= \begin{bmatrix} n_1 & n_2 \\ -n_2 & n_1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ -n_2 \left(\frac{\partial \rho}{\partial x_1}\right)^+ + n_1 \left(\frac{\partial \rho}{\partial x_2}\right)^+ \end{bmatrix} \\
&= \begin{bmatrix} n_2^2 \left(\frac{\partial \rho}{\partial x_1}\right)^+ + n_2 n_1 \left(\frac{\partial \rho}{\partial x_2}\right)^+ \\ -n_1 n_2 \left(\frac{\partial \rho}{\partial x_1}\right)^+ + n_1^2 \left(\frac{\partial \rho}{\partial x_2}\right)^+ \end{bmatrix} \\
&= \begin{bmatrix} n_2^2 & -n_1 n_2 \\ -n_1 n_2 & n_1^2 \end{bmatrix} \begin{bmatrix} \frac{\partial \rho}{\partial x_1} \\ \frac{\partial \rho}{\partial x_2} \end{bmatrix}^+.
\end{aligned}$$

The energy and turbulence model working variable gradients are set analogously. The velocity gradients are specified using $\frac{\partial \rho u_t}{\partial n} = 0$ and interior gradient information. Thus,

$$\begin{aligned}
\begin{bmatrix} \frac{\partial \rho u_1}{\partial x_1} \\ \frac{\partial \rho u_1}{\partial x_2} \\ \frac{\partial \rho u_2}{\partial x_1} \\ \frac{\partial \rho u_2}{\partial x_2} \end{bmatrix}^b &= \begin{bmatrix} n_1 & 0 & -n_2 & 0 \\ 0 & n_1 & 0 & -n_2 \\ n_2 & 0 & n_1 & 0 \\ 0 & n_2 & 0 & n_1 \end{bmatrix} \begin{bmatrix} \frac{\partial \rho u_1}{\partial n} \\ \frac{\partial \rho u_1}{\partial t} \\ \frac{\partial \rho u_2}{\partial n} \\ \frac{\partial \rho u_2}{\partial t} \end{bmatrix}^b \\
&= \begin{bmatrix} n_1^2 & -n_1 n_2 & -n_1 n_2 & n_2^2 \\ n_1 n_2 & n_1^2 & -n_2^2 & -n_1 n_2 \\ n_1 n_1 & -n_2^2 & n_1^2 & -n_1 n_2 \\ n_2^2 & n_1 n_2 & n_1 n_2 & n_1^2 \end{bmatrix} \begin{bmatrix} \frac{\partial \rho u_n}{\partial n}^+ \\ 0 \\ \frac{\partial \rho u_t}{\partial t}^+ \\ \frac{\partial \rho u_t}{\partial t}^+ \end{bmatrix} \\
&= \begin{bmatrix} (1 - n_1^2 n_2^2) & (n_1^3 n_2) & (-n_1 n_2^3) & (n_1^2 n_2^2) \\ (n_1^3 n_2) & (1 - n_1^4) & (n_1^2 n_2^2) & (-n_1^3 n_2) \\ (-n_1 n_2^3) & (n_1^2 n_2^2) & (1 - n_2^4) & (n_1 n_2^3) \\ (n_1^2 n_2^2) & (-n_1^3 n_2) & (n_1 n_2^3) & (1 - n_1^2 n_2^2) \end{bmatrix} \begin{bmatrix} \frac{\partial \rho u_1}{\partial x_1} \\ \frac{\partial \rho u_2}{\partial x_1} \\ \frac{\partial \rho u_1}{\partial x_2} \\ \frac{\partial \rho u_2}{\partial x_2} \end{bmatrix}^+.
\end{aligned}$$

Then,

$$\mathcal{F}_v^b = \mathcal{A}(\mathbf{u}^b) \nabla \mathbf{u}_h^b.$$

Bibliography

- [1] M. Ainsworth and J. T. Oden. *A Posteriori Error Estimation in Finite Element Analysis*. John Wiley and Sons, 2000.
- [2] 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.
- [3] S. R. Allmaras. Personal Communication via email, August 2007.
- [4] S. R. Allmaras and M. B. Giles. A second-order flux split scheme for the unsteady 2-D Euler equations on arbitrary meshes. AIAA Paper 1987-1119-CP, 1987.
- [5] W. Anderson, R. Rausch, and D. Bonhaus. Implicit multigrid algorithms for incompressible turbulent flows on unstructured grids. Number AIAA 95-1740-CP. In Proceedings of the 12th AIAA CFD Conference, San Diego CA, 1995.
- [6] D. N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19:742–760, 1982.
- [7] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptical problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.
- [8] I. Babuska, B. A. Szabo, and I. N. Katz. The p-version of the finite element method. *SIAM Journal on Numerical Analysis*, 18(3):515–545, 1981.
- [9] T. J. Baker. Mesh adaptation strategies for problems in fluid dynamics. *Finite Elements in Analysis and Design*, 25:243–273, 1997.
- [10] G. E. Barter and D. L. Darmofal. Shock capturing with higher-order, PDE-based artificial viscosity. AIAA Paper 2007-3823, 2007.
- [11] T. J. Barth. Recent developments in high-order k-exact reconstruction on unstructured meshes. AIAA Paper 1993-0668, 1993.
- [12] F. Bassi, A. Crivellini, S. Rebay, and M. Savini. Discontinuous Galerkin solution of the Reynolds averaged Navier-Stokes and $k - \omega$ turbulence model equations. *Computers and Fluids*, 34:507–540, May-June 2005.
- [13] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2d Euler equations. *Journal of Computational Physics*, 138(2):251–285, 1997.

- [14] F. Bassi and S. Rebay. A high-order discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 131:267–279, 1997.
- [15] F. Bassi and S. Rebay. GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations. In Cockburn, Karniadakis, and Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 197–208. Springer, Berlin, 2000.
- [16] F. Bassi and S. Rebay. Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 40:197–207, 2002.
- [17] R. Becker and R. Rannacher. A feed-back approach to error control in finite element methods: Basic analysis and examples. *East-West Journal of Numerical Mathematics*, 4:237–264, 1996.
- [18] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. In A. Iserles, editor, *Acta Numerica*. Cambridge University Press, 2001.
- [19] H. Borouchaki, P. George, F. Hecht, P. Laug, and E. Saltel. Maillageur bidimensionnel de Delaunay gouverné par une carte de métriques. Partie I: Algorithmes. INRIA-Rocquencourt, France. Tech Report No. 2741, 1995.
- [20] F. Brezzi, G. Manzini, D. Marini, P. Pietra, and A. Russo. Discontinuous Galerkin approximations for elliptic problems. *Numerical Methods for Partial Differential Equations*, 16(4):365–378, July 2000.
- [21] M. J. Castro-Diaz, F. Hecht, B. Mohammadi, and O. Pironneau. Anisotropic unstructured mesh adaptation for flow simulations. *International Journal for Numerical Methods in Fluids*, 25:475–491, 1997.
- [22] S. Catris and B. Aupoix. Density corrections for turbulence models. *Aerospace Science and Technology*, 4:1–11, 2000.
- [23] A. Celic and E.H. Hirschel. Comparison of eddy-viscosity turbulence models in flows with adverse pressure gradient. *AIAA Journal*, 44(10), 2006.
- [24] G. Chavent and G. Salzano. A finite element method for the 1D water flooding problem with gravity. *Journal of Computational Physics*, 42:307–344, 1982.
- [25] B. Cockburn, S. Hou, and C. W. Shu. Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case. *Mathematics of Computation*, 54:545–581, 1990.
- [26] B. Cockburn, G. Karniadakis, and C. Shu. The development of discontinuous Galerkin methods. In *Lecture Notes in Computational Science and Engineering*, volume 11. Springer, 2000.
- [27] B. Cockburn, S. Y. Lin, and C. W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One dimensional systems. *Journal of Computational Physics*, 84:90–113, 1989.

- [28] B. Cockburn and C. W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for scalar conservation laws II: General framework. *Mathematics of Computation*, 52:411–435, 1989.
- [29] B. Cockburn and C. W. Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6):2440–2463, December 1998.
- [30] B. Cockburn and C. W. Shu. The Runge-Kutta discontinuous Galerkin finite element method for conservation laws V: Multidimensional systems. *Journal of Computational Physics*, 141:199–224, 1998.
- [31] B. Cockburn and C. W. Shu. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, pages 173–261, 2001.
- [32] D.E. Coles and E.A. Hirst. Computation of turbulent boundary layers. 1968 AFOSR-IFP-Stanford Conference, Volume II, 1969.
- [33] S. Collis and M. Heinkenschloss. Analysis of the streamline upwind/Petrov Galerkin method applied to the solution of optimal control problems. Technical Report 02-01, Rice University Department of Computational and Applied Mathematics, Houston, Texas, March 2002.
- [34] L. T. Diosady. A linear multigrid preconditioner for the solution of the Navier-Stokes equations using a discontinuous Galerkin discretization. Masters thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, May 2007.
- [35] L.T. Diosady and D.L. Darmofal. Discontinuous Galerkin solutions of the Navier-Stokes equations using linear multigrid preconditioning. AIAA Paper 2007-3942, 2007.
- [36] J. Elliot and J. Peraire. Practical three-dimensional aerodynamic design and optimization using unstructured meshes. *AIAA Journal*, 35(9):1479–1485, 1997.
- [37] H. W. Emmons. Critique of numerical modeling of fluid-mechanics phenomena. *Annual Review of Fluid Mechanics*, 2:15–36, 1970.
- [38] A. Ern and J. L. Guermond. *Theory and Practice of Finite Elements*. Springer, New York, 2004.
- [39] K. J. Fidkowski. *A Simplex Cut-Cell Adaptive Method for High-Order Discretizations of the Compressible Navier-Stokes Equations*. PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2007.
- [40] K. J. Fidkowski and D. L. Darmofal. Output-based adaptive meshing using triangular cut cells. M.I.T. Aerospace Computational Design Laboratory Report. ACDL TR-06-2, 2006.
- [41] K. J. Fidkowski and D. L. Darmofal. An adaptive simplex cut-cell method for discontinuous Galerkin discretizations of the Navier-Stokes equations. AIAA Paper 2007-3941, 2007.

- [42] K. J. Fidkowski and D. L. Darmofal. A triangular cut-cell adaptive method for higher-order discretizations of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 225:1653–1672, 2007.
- [43] M. B. Giles and N. Pierce. Adjoint error correction for integral outputs. In *Lecture Notes in Computational Science and Engineering: Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, volume 25. Springer, Berlin, 2002.
- [44] M. B. Giles and E. Süli. Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality. In *Acta Numerica*, volume 11, pages 145–236, 2002.
- [45] M.B. Giles, N.A. Pierce, and E. Suli. Progress in adjoint error correction for integral functionals. *Computing and Visualisation in Science*, 6(2-3):113–121, 2004.
- [46] P. Godin, D. W. Zingg, and T. E. Nelson. High-lift aerodynamic computations with one- and two-equation turbulence models. *AIAA Journal*, 35(2), 1997.
- [47] L. M. Graves. Riemann integration and Taylor’s theorem in general analysis. *Transactions of the American Mathematical Society*, 29(1):163–177, 1927.
- [48] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M. G. Vallet. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. part I: general principles. *International Journal for Numerical Methods Fluids*, 32:725–744, 2000.
- [49] D. W. Halt. *A compact higher-order Euler solver for unstructured grids*. PhD thesis, Washington University, 1992.
- [50] K. Harriman, D. Gavaghan, and E. Süli. The importance of adjoint consistency in the approximation of linear functionals using the discontinuous Galerkin finite element method. Technical Report 04/18, Oxford University Computing Laboratory, Numerical Analysis Group, Oxford, England, July 2004.
- [51] K. Harriman, P. Houston, B. Senior, and E. Süli. *hp*-version discontinuous Galerkin methods with interior penalty for partial differential equations with nonnegative characteristic form. Technical Report Technical Report NA 02/21, Oxford University Computing Lab Numerical Analysis Group, 2002.
- [52] R. Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 51:1131–1156, 2006.
- [53] R. Hartmann. Adjoint consistency analysis of discontinuous Galerkin discretizations. *SIAM Journal of Numerical Analysis*, 45(6):2671–2696, 2007.
- [54] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. *SIAM Journal on Scientific Computing*, 24:979–1004, 2002.
- [55] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2):508–532, 2002.

- [56] R. Hartmann and P. Houston. Goal-oriented a posteriori error estimation for multiple target functionals. In T.Y. Hou and E. Tadmor, editors, *Hyperbolic Problems: Theory, Numerics, Applications*, pages 579–588. Springer-Verlag, 2003.
- [57] R. Hartmann and P. Houston. Symmetric interior penalty DG methods for the compressible Navier-Stokes equations II: Goal-oriented a posteriori error estimation. *International Journal of Numerical Analysis and Modeling*, 3(2):141–162, 2006.
- [58] F. Hecht. *BAMG: Bidimensional Anisotropic Mesh Generator*. INRIA, Rocquencourt, France, 1998. <http://www-rocq1.inria.fr/gamma/cdrom/www/bamg/eng.htm>.
- [59] P. Houston and E. Süli. hp -adaptive discontinuous Galerkin finite element methods for first-order hyperbolic problems. *SIAM Journal on Scientific Computing*, 23(4):1226–1252, 2001.
- [60] P. Houston and E. Süli. A note on the design of hp -adaptive finite element methods for elliptic partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 194:229–243, 2005.
- [61] T. J. R. Hughes, L.P. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: I Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics. *Computer Methods in Applied Mechanics and Engineering*, 54:223–234, 1986.
- [62] A. Jameson. Aerodynamic design via control theory. *Journal of Science and Computation*, 3:233–260, 1988.
- [63] C. Johnson and J. Pitkaranta. An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation. *Mathematics of Computation*, 46:1–26, 1986.
- [64] K. Z. Korczak and A. T. Patera. An isoparametric spectral element method for solution of the Navier-Stokes equations in complex geometry. *Journal of Computational Physics*, 62:361–382, 1984.
- [65] K. R. Laffin, J. C. Vassberg, R. A. Wahls, J. H. Morrison, O. Brodersen, M. Rakowitz, E. N. Tinoco, and J. L. Godard. Summary of data from the second AIAA CFD drag prediction workshop. AIAA Paper 2004-0555, 2004.
- [66] B. Landmann. *A parallel discontinuous Galerkin code for the Navier-Stokes and Reynolds-averaged Navier-Stokes equations*. PhD thesis, University of Stuttgart, 2008.
- [67] T. Leicht and R. Hartmann. Anisotropic mesh refinement for discontinuous Galerkin methods in two-dimensional aerodynamic flow simulations. *International Journal for Numerical Methods in Fluids*, 56(11):2111–2138, April 2008.
- [68] S. K. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103:16–42, 1992.
- [69] 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.

- [70] D. W. Levy, T. Zickuhr, J. Vassberg, S. Agrawal, R. A. Wahls, S. Pirzadeh, and M. J. Hensch. Data summary from the First AIAA Computational Fluid Dynamics Drag Prediction Workshop. *Journal of Aircraft*, 40(5):875–882, 2003.
- [71] J. C. Lin and C. J. Dominik. Optimization of an advanced design three-element airfoil at high reynolds numbers. AIAA Paper 1995-1858, 1995.
- [72] J. Lu. *An a Posteriori Error Control Framework for Adaptive Precision Optimization Using Discontinuous Galerkin Finite Element Method*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2005.
- [73] D. J. Mavriplis. Results from the 3rd drag prediction workshop using the NSU3D unstructured mesh solver. AIAA Paper 2007-256, 2007.
- [74] P. Moin and K. Mahesh. Direct numerical simulation: A tool in turbulence research. *Annual Review of Fluid Mechanics*, 30:539–578, 1998.
- [75] J. H. Morrison and M. J. Hensch. Statistical analysis of CFD solutions from the third AIAA drag prediction workshop. AIAA Paper 2007-254, 2007.
- [76] N.C. Nguyen, P.O. Persson, and J. Peraire. RANS solutions using high order discontinuous Galerkin methods. AIAA Paper 2007-0914, 2007.
- [77] E. J. Nielsen and W. K. Anderson. Recent improvements in aerodynamic design optimization on unstructured meshes. *AIAA Journal*, 40(6):1155–1163, 2002.
- [78] J. T. Oden, I. Babuska, and C. E. Baumann. A discontinuous *hp* finite element method for diffusion problems. *Journal of Computational Physics*, 146:491–519, 1998.
- [79] T. A. Oliver and D. L. Darmofal. An unsteady adaptation algorithm for discontinuous Galerkin discretizations of the RANS equations. AIAA Paper 2007-3940, 2007.
- [80] M. A. Park and D. L. Darmofal. Parallel anisotropic tetrahedral adaptation. AIAA Paper 2008-917, 2008.
- [81] A. T. Patera. A spectral element method for fluid dynamics: laminar flow in a channel expansion. *Journal of Computational Physics*, 54:468–488, 1984.
- [82] J. Peraire and P. O. Persson. The compact discontinuous Galerkin (CDG) method for elliptic problems. *SIAM Journal on Scientific Computing*, 30:1806–1824, 2008.
- [83] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz. Adaptive remeshing for compressible flow computations. *Journal of Computational Physics*, 72:449–466, 1987.
- [84] P. O. Persson and J. Peraire. Sub-cell shock capturing for discontinuous Galerkin methods. AIAA Paper 2006-0112, 2006.
- [85] N. A. Pierce and M. B. Giles. Adjoint recovery of superconvergent functionals from PDE approximations. *SIAM Review*, 42(2):247–264, 2000.
- [86] S. Z. Pirzadeh. An adaptive unstructured grid method by grid subdivision, local remeshing, and grid movement. AIAA Paper 99-3255, 1999.

- [87] S. Prudhomme and J. T. Oden. Computable error estimators and adaptive techniques for fluid flow problems. In Timothy J. Barth and Herman Deconinck, editors, *Lecture Notes in Computational Science and Engineering: Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, volume 25, pages 207–268. Springer, Berlin, 2003.
- [88] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer, New York, 1997.
- [89] S. De Rango and D. W. Zingg. High-order spatial discretization for turbulent aerodynamic computations. *AIAA Journal*, 39(7), 2001.
- [90] 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.
- [91] O. Reynolds. On the dynamical theory of incompressible viscous fluids and the determination of the criterion. *Philosophical Transactions of the Royal Society of London. A*, 186:123–164, 1895.
- [92] G. R. Richter. An optimal-order error estimate for the discontinuous Galerkin method. *Mathematics of Computation*, 50:75–88, 1988.
- [93] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.
- [94] C. W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77:439–471, 1988.
- [95] P. R. Spalart and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. *La Recherche Aéronautique*, 1:5–21, 1994.
- [96] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method*. Wellesley-Cambridge Press, Wellesly, MA, 1988.
- [97] E. F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*. Springer, New York, 1999.
- [98] B. van Leer. Towards the ultimate conservative difference scheme. IV - a new approach to numerical convection. *Journal of Computational Physics*, 23:276–299, 1977.
- [99] B. van Leer and Shohei Nomura. Discontinuous Galerkin for diffusion. AIAA Paper 2005-5108, 2005.
- [100] J. C. Vassberg, M. A. DeHaan, and T. J. Scalfani. Grid generation requirements for accurate drag predictions based on OVERFLOW calculations. AIAA Paper 2003-4124, 2003.
- [101] J. C. Vassberg, E. N. Tinoco, M. Mani, O. P. Brodersen, B. Eisfeld, R. A. Wahls, J. H. Morrison, T. Zickuhr, K. R. Laffin, and D. J. Mavriplis. Summary of the Third AIAA CFD Drag Prediction Workshop. AIAA Paper 2007-260, 2007.

- [102] D. A. Venditti. *Grid Adaptation for Functional Outputs of Compressible Flow Simulations*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2002.
- [103] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *Journal of Computational Physics*, 187(1):22–46, 2003.
- [104] M. R. Visbal and D. V. Gaitonde. On the use of higher-order finite-difference schemes of curvilinear and deforming meshes. *Journal of Computational Physics*, 181:155–185, 2002.
- [105] Z. J. Wang. Spectral (finite) volume method for conservation laws on unstructured grids. Basic formulation. *Journal of Computational Physics*, 178:210–251, 2002.
- [106] G. P. Warren, W. K. Anderson, J. T. Thomas, and S. L. Krist. Grid convergence for adaptive methods. AIAA Paper 1991-1592, 1991.
- [107] M. Wheeler. An elliptic collocation-finite element method with interior penalties. *SIAM Journal on Numerical Analysis*, 15:152–161, 1978.
- [108] D. C. Wilcox. *Turbulence Modeling for CFD*. DCW Industries, Inc., La Cañada, California, 2006.
- [109] X. D. Zhang, M. G. Vallet, J. Dompierre, P. Labbe, D. Pelletier, J. Y. Trepanier, R. Camarero, J. V. Lassaline, L. M. Manzano, and D. W. Zingg. Mesh adaptation using different error indicators for the Euler equations. AIAA Paper 2001-2549, 2001.
- [110] O. C. Zienkiewicz and J. Z. Zhu. A simple error estimation and adaptive procedure for practical engineering analysis. *International Journal for Numerical Methods in Engineering*, 24:337–357, 1987.
- [111] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery and *a posteriori* error estimates. Part 1: The recovery technique. *International Journal for Numerical Methods in Engineering*, 33:1331–1364, 1992.
- [112] O. C. Zienkiewicz and J. Z. Zhu. The superconvergent patch recovery and *a posteriori* error estimates. Part 2: Error estimates and adaptivity. *International Journal for Numerical Methods in Engineering*, 33:1365–1382, 1992.
- [113] D. W. Zingg, S. De Rango, M. Nemeć, and T. H. Pulliam. Comparison of several spatial discretizations for the Navier-Stokes equations. *Journal of Computational Physics*, 160:683–704, 2000.