A Cut-Cell Method for Adaptive High-Order Discretizations of Conjugate Heat Transfer Problems

by

Steven Matthew Ojeda

B.S., Massachusetts Institute of Technology (2012)

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

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2014

Signature redacted

Author .....
Department of Aeronautics and Astronautics
May 22, 2014

Signature redacted

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

Signature redacted

Accepted by ....
Paulo C. Lozano
Associate Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

A Cut-Cell Method for Adaptive High-Order Discretizations of Conjugate Heat Transfer Problems

by

Steven Matthew Ojeda

Submitted to the Department of Aeronautics and Astronautics
on May 22, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Heat transfer between a conductive solid and an adjacent convective fluid is prevalent in many aerospace systems. The ability to achieve accurate predictions of the coupled heat interaction is critical in advancing thermodynamic designs. Despite their growing use, coupled fluid-solid analyses known as conjugate heat transfer (CHT) are hindered by the lack of automation and robustness. The mesh generation process is still highly dependent on user experience and resources, requiring time-consuming involvement in the analysis cycle. This thesis presents work toward developing a robust PDE solution framework for CHT simulations that autonomously provides reliable output predictions. More specifically, the framework is comprised of the following components: a simplex cut-cell technique that generates multi-regioned meshes decoupled from the design geometry, a high-order discontinuous Galerkin (DG) discretization, and an anisotropic output-based adaptation method that autonomously adapts the mesh to minimize the error in an output of interest.

An existing cut-cell technique is first extended to generate fully-embedded meshes with multiple sub-domains. Then, a coupled framework that combines separate disciplines is developed, while ensuring compatibility between the cut-cell and mesh adaptation algorithms. Next, the framework is applied to high-order discretizations of the heat, Navier-Stokes, and Reynolds-Averaged Navier-Stokes (RANS) equations to analyze the heat flux interaction. Through a series of numerical studies, high-order accurate outputs solved on autonomously controlled cut-cell meshes are demonstrated. Finally, the conjugate solutions are analyzed to gain physical insight to the coupled interaction.

Thesis Supervisor: David Darmofal
Title: Professor of Aeronautics and Astronautics

Acknowledgments

I would like to express my sincere gratitude to all people who have made this thesis possible. First, I would like to thank my advisor, Prof. David Darmofal, for providing me with the opportunity to learn from and contribute to the CFD community, and for encouraging me during my graduate study. In addition, I would like to thank Dr. Steven Allmaras for his very knowledgeable insight and continued commitment to my weekly meetings. I would also like to recognize Marshall for teaching me the tools of the trade, and for providing me with the continuous feedback I needed to learn.

This thesis would not have been possible without the relentless efforts of the entire ProjectX team. I would like to thank past members who had built the building blocks and who had contributed many years of their lives developing ProjectX. I would also like to thank: Huafei for helping me climb the learning curve, and for laying down the foundation of my work; Jun and Phil for their unbounded willingness to help solve problems, share ideas, and drink coffee with me; Carlee, Savi, Yixuan, and Jeff for bringing new perspectives to the team. Thank you all, and the best of luck to you.

I would like recognize people in the ACDL for creating a productive working environment: David M, Xevi, Ferran, and Hemant for sharing their ideas and encouragement; Patrick for making me take breaks for coffee hour; Eric for his admin support and sense of humor. I would like to thank all my friends outside the ACDL: the brothers of Theta Chi for always extending a helping hand; Giulia and Ed (team WindX) for encouraging me with their positivity and puns; Andras for his problem solving and cooking skills; among many more. I would also like to give a special thanks to Erica for inspiring me to summit my goals, strive for a healthy lifestyle, and reframe all frustrating scenarios into positive ones.

Lastly, I would like to thank my family – Mom, Dad, and Laura – for their unconditional love and support. The past few years have been quite difficult, and I would not be where I am today without their continuous guidance and encouragement. I wish

you many years of joy and health, and I am excited to be moving closer to home.

Finally, I would like to acknowledge the financial support provided by the Boeing Company (technical monitor Dr. Mori Mani), the MIT Graduate Work Program, and the MIT AeroAstro Department through fellowships.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Numerical simulation has become a critical tool for engineering analysis over the last several decades. In particular, Computational Fluid Dynamics (CFD) software has widely been used throughout academia and industry to analyze and drive aerospace design. As an alternative to experimental testing for certification, CFD offers relatively fast turnaround times and the ability to simulate a wide range of component designs and test conditions. Additionally, with recent advancements in algorithm development and increased computational power, CFD solvers are becoming far more capable in analyzing problems with complex geometry, physics, or both. However, despite their widespread use, many CFD software packages still lack efficiency, reliability, and autonomy, yielding unaffordable high-fidelity simulations, unreliable predictions in outputs of interest, and heavy user involvement.

Depending on the size of the problem and the computational resources being used, common high-fidelity simulations can take hours or even days to complete. Equally inhibiting, user involvement is dominated by two factors: (1) the process of understanding and determining where mesh refinement is necessary in order to achieve an accurate solution, and (2) the process of generating a corresponding mesh to conform with the modeled geometry, which can take weeks to even months depending on the

geometry complexity. Worse still, unreliable predictions in outputs of interest compounded with the absence of solution uncertainty quantification greatly impairs the validity of the solution, diminishes the value of the expensive analysis cycle, and can even lead to large-scale disasters. For example, the 44,000-ton Sleipner A offshore platform sank in 1991 due to a flawed design. Costing $700 million, this failure was caused by a finite element analysis that underestimated the shear stress in a concrete support structure by 45 % [38]. After further investigations, it was found that the underestimation resulted from an under-resolved mesh with poorly shaped elements. The analysis was performed again with a suitable mesh, and returned with a predicted structural failure occurring at a water depth of 62 meters, agreeing closely with the actual failure depth of 65 meters [24]. Hence, the lack of reliability in the mesh generation and error estimation process led to a catastrophic failure that could have been avoided with proper error control.

Fortunately, mesh adaptation offers a means toward mitigating mesh reliance on human experience, and instead provides a far more reliable output prediction through a systematic and autonomous control over the error. In conjunction, higher-order discretizations, which are becoming more prevalent in the CFD community, can further improve the solution and output accuracy. Despite these promises, however, the mesh generation process in an industry setting still serves as a primary 'bottleneck' in the CAD-to-mesh-to-solution cycle [26]. A couple difficulties contributing to this bottleneck involve: requiring curved elements to conform to the geometry surface in order to maintain the benefit of higher-order discretizations [7], and achieving different levels of refinement in specific areas within the domain.

In the context of multi-disciplinary simulations where multiple governing partial differential equations (PDE's) are solved simultaneously, many of the mesh generation and error control issues are exacerbated by the complexity of the coupled problem. The ability to sufficiently resolve important regions in the domain is non-trivial for single disciplines, let alone multiple disciplines. Often times, many industry simulations that involve solid bodies interacting with fluid flow simplify the problem by

assuming conditions that decouple the two domains, such as using adiabatic walls or a constant heat flux assumption, as oppose to solving the fully coupled conjugate heat transfer (CHT) problem. Certainly, if the thermal variation within the solid is negligible, these assumptions may apply; however, many aerospace applications consist of thermal-fluid interactions that are highly coupled, and reliable solutions cannot be obtained if the assumptions are maintained. Instead, a robust and autonomous method for generating multi-disciplinary, adapted meshes is necessary to conduct efficient CHT simulation. With the help of mesh adaptation, the ability to accurately capture the coupled interaction between a fluid and solid can foster and facilitate aerospace design by:

- mitigating crack development due to thermal shocks
- reducing hot spots in high-temperature environments
- allowing reductions in coolant flows
- determining performance metrics for heat exchangers
- increasing accuracy in predicted heat transfer coefficients
- optimizing highly coupled thermal-fluid component design

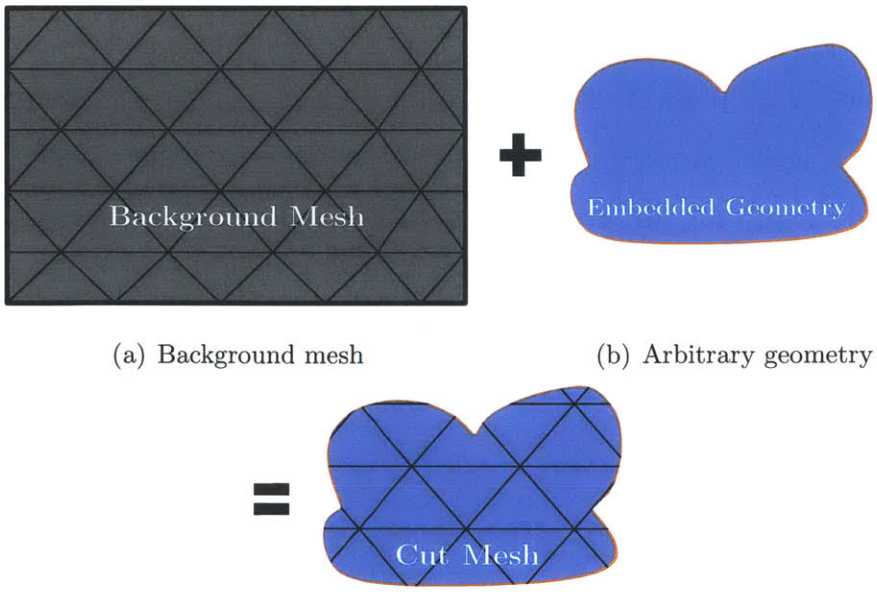
With these applications in mind, this thesis presents work toward: streamlining the CAD-to-mesh-to-solution cycle through a cut-cell technique, utilizing high order discretizations, and integrating an error estimation and adaptation process for multi-disciplinary conjugate heat transfer problems.

1.2 Background

1.2.1 Cut-Cell Methods

The mesh generation process for unstructured grids is often very time-consuming, and also suffers from robustness issues involving highly anisotropic elements around high-order curved geometries. One alternative for a streamlined and robust process is the use of cut cells, where the computational mesh is 'cut' from a background

mesh that is not required to conform to the geometry at hand. This process begins with the generation of a coarse background mesh, which is then intersected with a non-conforming geometry, or *embedded* geometry, to create a cut-cell mesh consisting of arbitrarily shaped elements. Depending on the type of problem, the embedded geometry may specify a solid body embedded in an external flow (in which case the external mesh becomes the computational domain), or it may specify the outer boundaries of a fully embedded domain (in which case the interior mesh becomes the computational domain). FIGURE 1-1 illustrates an example of the latter, where a fully embedded cut-cell mesh is formed from a background mesh intersecting with an arbitrary embedded geometry. Without the requirement of boundary conformity, the generation of a cut-cell mesh now only relies on an automated intersection algorithm. However, the burden of robustness is transferred to the PDE solver, which must now account for the resulting arbitrarily shaped elements within the cut-cell mesh. Despite this added complexity, the automated benefits of the cut-cell technique may still outweigh the large cost associated with a traditional boundary-conforming mesh generation process.



(a) Background mesh

(b) Arbitrary geometry

(c) Cut mesh from embedded geometry

FIGURE 1-1: Example of a cut-cell mesh

The initial concept of cut-cell methods was first considered by Purvis and Burkhalter [62] in 1979. In their work, the full potential equations were solved on cut Cartesian background meshes, which has more recently been extended to 3D meshes with complex boundaries. For example, Young *et al.* [80] demonstrated an accurate and reliable method for solving the 3D potential flow equations on complex geometry. Their Cartesian cut-cell finite element method used Stokes' theorem to carry out the volume integration of linear cut cells, and allowed for adaptation based on geometric or solution features. Furthermore, Karman [43] had developed software (SPLITFLOW) that used a Cartesian grid to solve the 3D Reynolds-Averaged Navier-Stokes (RANS) equations. However, this technique required *a priori* knowledge of the location and orientation of dominant flow features in order to properly align the Cartesian grid. As such, Cartesian grids are limited in the direction in which anisotropic elements are desired, and are therefore inefficient in resolving anisotropic features in Navier-Stokes and RANS problems.

To overcome this limitation, Fidkowski and Darmofal developed a simplex cut-cell method for 2D and 3D embedded boundary problems [32]. Combined with a high-order discontinuous Galerkin discretization, this method used cubic splines to represent embedded geometries in two dimensions, and quadratic patches in three dimensions. The method enabled anisotropic adaptation, though exhibited low quadrature quality in arbitrarily shaped cut cells, and ill-conditioning due to small volume ratios between element neighbors. Since then, robustness and automation improvements of the method have occurred. For instance, Modisette developed an algorithm to recognize canonical shapes for cut cells in two dimensions to improve their respective quadrature quality, as well as a merging technique to improve the overall conditioning [51]. Additionally, Sun introduced the use of magic points for cut-cell quadrature rules, and demonstrated the method on interface problems [72].

This improved cut-cell method becomes particularly useful in 'multi-regioned' simulations, where generating interface-conforming meshes becomes far more challenging as the number of region boundaries and interfaces increase. Several methods, such as the

immersed interface method [46] and ghost fluid method [30], have been developed to handle non-interface-conforming meshes, though are typically second-order accurate. In order to achieve high-accuracy solutions to interface problems, the method must be compatible with high-order discretizations [71]. Though the cut-cell method by Sun produces meshes to obtain high-order solutions to a single interface problem, the algorithm is not capable of generating cut-cell meshes for multi-disciplinary domains with an arbitrary number of boundaries and interfaces. Hence, Sun's cut-cell method is used as a starting point for developing a method to streamline the mesh generation process for multi-regioned, high-order conjugate heat transfer problems.

1.2.2 High-Order Discretizations

The primary goal of high-order methods is to achieve higher fidelity solutions at a fraction of the cost of low-order methods. Because of their ability to reduce discretization error, high-order methods become important in complex problems that require high accurate solutions. As an example, a heat transfer review by Peniguel illuminates common low-order methods that overestimates the average Nusselt number of a heated flat plate case by a factor of two [58]. Not only that, Peniguel expresses that this overestimation is typical for industry RANS solvers. With the added complexity of a strongly coupled conjugate heat transfer problem, the ability to control the discretization error is paramount.

Most aerospace industry CFD methods today can only achieve second order error convergence defined as $E \propto \mathcal{O}(h^2)$, where E is a measure of the error, and h is a measure of the mesh size. Generally speaking, *high-order* refers to a discretization's ability to obtain a higher convergence rate r in order to achieve improved accuracy. More specifically, high-order methods typically achieve error convergence rates that are higher than second order: $E \propto \mathcal{O}(h^{r>2})$, for an L^2-error measure.

In this work, finite element schemes are used to achieve high-order discretizations as they are applicable to unstructured meshes, which can more readily tessellate models

with complex boundary or interface geometry. They also offer an elegant extension to high-order accurate solutions by increasing the order of basis polynomials. The concept of increasing the polynomial order p (while maintaining a constant grid spacing, h) is known as a 'p-type' method, which had originally been applied to elasticity equations in 1981 by Babuska *et al.* [4]. Their study found that the p-type method required fewer degrees of freedom to achieve a similar level of accuracy when applied to smooth problems. In order to realize the benefits of high-order methods for problems with low regularity, however, h-type methods (one that adapts the grid spacing) are required to control the resulting discretization error. For instance, Yano *et al.* [79] demonstrated the affordability of high-order convergence by solving compressible Navier-Stokes and RANS problems with a mesh adaptation method.

To stabilize the finite element discretization for convection-dominated problems, the discontinuous Galerkin (DG) method is used. The DG method dates back to 1973, where Reed and Hill first introduced it for scalar hyperbolic equations [63]. Soon after, LeSaint and Raviart [45] proved that, assuming a smooth solution with a p-order polynomial basis, the L^2-error of the DG method is $\mathcal{O}(h^p)$, while Richter [64] proved a decade later that convergence rates of $\mathcal{O}(h^{p+1})$ can be obtained. The method was later extended to nonlinear hyperbolic problems by Chavent and Salzano [18] using Godunov's flux, followed by an extension to using a Runge-Kutta explicit time integration (RKDG) by Cockburn, Shu, and co-authors [21, 20, 19, 22]. For solving elliptic problems, Arnold and Wheeler [3, 76] developed interior penalty methods, while Bassi and Rebay later developed the so-called BR1 [8] and BR2 [9] discretizations. The BR2 scheme achieves stability for purely elliptic problems, and serves as the viscous discretization in this work.

1.2.3 Output-Based Error Estimation and Mesh Adaptation

Many engineering applications require the development of high-fidelity models to accurately predict an output of interest. This development typically involves the generation of a mesh that, through experience, is refined in areas that reduces the

output error. For complex problems (including CHT), experience is not enough to locate regions needing refinement *a priori*. Hence, an *output*-based adaptation framework that resolves appropriate domain features in the absence of human experience is desired. Combined with a high-order discretization and a cut-cell method framework, mesh adaptation could enable a user to specify a required error level and a maximum allotted time for an automated multi-disciplinary simulation. FIGURE 1-2 illustrates an ideal flow of information to achieve an output of interest within a desired error tolerance, $\mathcal{E}_{\max}$, and a desired solve time, $\tau_{\max}$. For a multi-disciplinary problem, the process begins with a cut-mesh generated by intersecting the multi-regioned geometry with an initial (coarse) background mesh. Next, the discretized PDE's are solved on the new cut-mesh, followed by an estimation of the resulting output error. If the error or time tolerances are not met, the estimated error is localized on an elemental level, and the *background* mesh is adapted to reduce the error. This process continues until the outputs reach an acceptable error level, or the run-time is exhausted. The three primary modules in this process are the cut-cell framework (as discussed earlier), the output error estimation, and the mesh adaptation procedure.

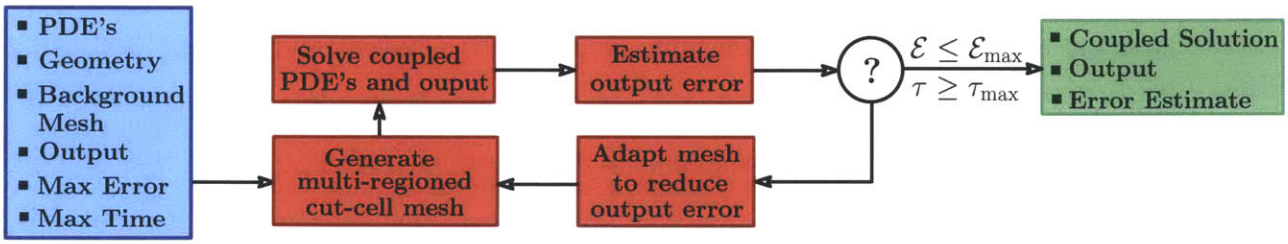


FIGURE 1-2: Automated process of the output-based adaptive PDE solver

Error Estimation

The purpose of error estimation in the adaptive process is to first determine a global error to assess the validity of an output of interest, then localize the error to identify which elements contribute to the global error the most. Several varieties of error estimation have been developed to achieve these goals. A common strategy is to

identify dominant solution features based on the areas marked by large gradients, as shown by Baker [5]. However, large gradients in a solution do not necessarily localize the area of greatest error. For example, small upstream perturbations can have an effect on downstream features, despite the largest gradients occurring in the downstream features [75]. Another method is residual-based error estimation, which relies on calculating residual norms that bound the error. Though this method was demonstrated to function well in one-dimensional transonic flows with shocks [81], it has not been proven successful in two dimensions.

An improved method consists of an output-based error estimation technique that localizes the output error by incorporating the corresponding adjoint solution from the dual problem. The adjoint serves as the sensitivity of an output of interest to perturbations in the primal residual, which links the output error to the local residual. Such method is given by the dual-weighted residual (DWR) method proposed by Becker and Rannacher [11, 12], and is used in this work to drive mesh adaptation. The DWR method has been extended and implemented for several mesh adaptation procedures, notably the first output-based, anisotropic adaptation method applied to RANS problems by Venditti and Darmofal [73].

Mesh Adaptation

Given a localized error estimate, the purpose of mesh adaptation is to minimize the output error by changing the mesh spacing (h-type adaptation). In other words, given an amount of resources (degress of freedom), the best approximation of the output of interest can be determined by controlling the error through mesh optimization. For purely isotropic adaptation, the localized error from the DWR method is sufficient to carry out a fixed fraction strategy. This technique first ranks all elements within the computational domain by their localized error, then performs a refinement of those with the highest error, and a coarsening of those with the lowest error. However, to generate efficient meshes for high-Reynolds flows where boundary layers, wakes, and shocks require a resolution with high aspect ratios, anisotropic meshes are needed.

One way to represent an anisotropic mesh is to formulate the mesh information as a metric tensor field that defines the size, stretching, and orientation of each element. Though an anisotropic mesh can be fully characterized by a conforming metric field, it still belongs to a family of metric-conforming meshes that all have similar approximation properties [48, 49]. Given a metric field, several anisotropic mesh generators such as BAMG [14, 37] can be used to generate corresponding meshes. This way, an anisotropic mesh that conforms to a desired distribution that reduces the total output error is possible. Peraire *et al.* [60] developed a method based on the Hessian of a solution field to control the anisotropic mesh configuration. Venditti and Darmofal [73] proposed an anisotropic adaptation method that relies on both the DWR method and an anisotropy detection based on the Hessian of the Mach number, while Fidkowski and Darmofal [32] extended this idea to high-order discretizations.

Beyond this, Yano and Darmofal [78] proposed the Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm, which determines adapted meshes by solving a continuous constrained optimization problem. This is done by formulating the objective as the output error, and defining the design variables as the degrees of freedom in a metric tensor field. The error-metric sensitivities are approximated by first constructing split configurations for each element, then solving for the resulting error for each configuration. As defined, this method is only capable of performing error sampling over triangular elements, and therefore cannot be used for arbitrarily shaped elements. However, Sun has extended this method to handle cut-cell meshes for embedded boundary problems [71], and performs the adaptation procedure on the background mesh. Additionally, Kudo [39] has modified the mesh adaptation method to use a gradient-based optimization algorithm.

In the context of multi-disciplinary problems, mesh adaptation is important in resolving anisotropic and perhaps non-intuitive features in each sub-domain. Since multi-regioned domains are tessellated using a cut-cell mesh, and the mesh adaptation process is performed on the *background* mesh, the modification and generation of a new optimized mesh is completely decoupled from the geometry at hand. Hence,

the combined cut-cell and mesh adaptation framework allows for efficient and autonomous output error minimization of multi-regioned domains, and is demonstrated in this thesis for conjugate heat transfer problems.

1.3 Thesis Overview

This thesis presents work toward developing a robust, PDE solution framework for CHT simulations that autonomously provides reliable output predictions. In particular, the framework is comprised of three primary components: a simplex cut-cell technique, a high-order DG discretization, and an anisotropic output-based adaptation method. The specific contributions of this thesis are as follows:

- Extension of cut-cell methods to fully-embedded, multi-regioned domains.
- Development of a coupled framework that combines separate disciplines, compatible with the integrated cut-cell and mesh adaptation methods.
- Demonstration of the CHT framework with high-order discretizations of the heat, Navier-Stokes, and RANS equations.

In this thesis, CHAPTER 2 details the cut-cell algorithm used to generate multi-regioned meshes. CHAPTER 3 provides background for the DG discretization, dual-weighted residual output error estimation, and mesh optimization with cut-cell extensions. In CHAPTER 4, the CHT framework for the coupled heat and Navier-Stokes equations is demonstrated, while CHAPTER 5 demonstrates the CHT framework for the coupled heat and RANS equations.

Chapter 2

Multi-Disciplinary Cut Cell Methods

Many aerospace applications rely on different analysis models to support and drive multi-disciplinary design. When coupled together, these analyses typically require computational meshes that are capable of representing different regions separated by arbitrary interfaces. However, generating meshes that conform to the domain boundary and interfaces can be very time-consuming if the geometry representation is complex. Cut-cell methods, on the other hand, do not require a mesh to conform to the interface geometry, and instead offer an alternative mesh generation process for multi-regioned domains. In this chapter, a two-dimensional cut-cell method for multi-regioned domains is presented.

2.1 Geometry Definition

For multi-disciplinary problems, the geometry consists of multiple boundaries and interfaces that form separate closed loop regions. FIGURE 2-1 illustrates an example of a multi-regioned domain consisting of three regions, Ω_1, Ω_2, and Ω_3, separated by interfaces, Σ_{13}, Σ_{12}, and Σ_{23}. In two dimensions, these geometric features are uniquely characterized by a set of nodes, faces, and face groups as defined in TABLE 2.1.

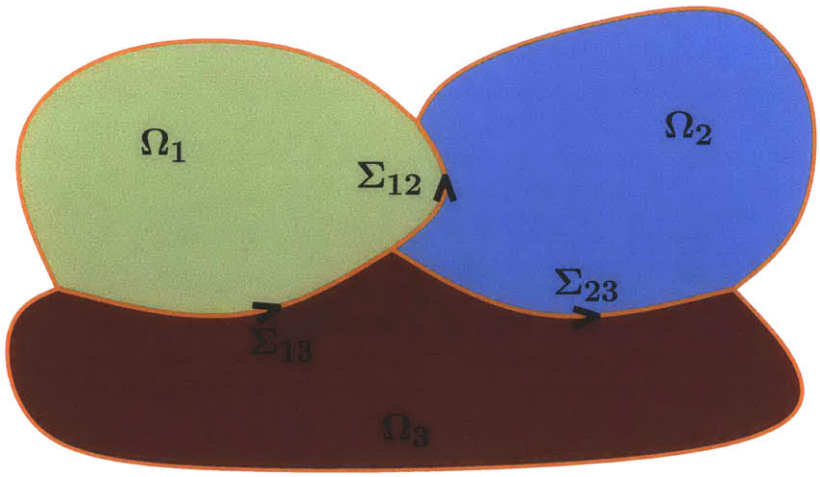


FIGURE 2-1: Multi-region interface example

Attribute	Symbol	Definition
Nodes	n_i	(x_i, y_i)
Faces	f_{ij}	(node i, node j)
Face Groups	F_i	$\{f_{jk}\}, (l_i, r_i)$

TABLE 2.1: 2D multi-region geometry attributes

The nodes are defined by a set of coordinates (x_i, y_i), and the faces are defined by an ordered nodal pair. The face groups are defined by both a set of faces, $\{f_{jk}\}$, and a left and right region index, l_i and r_i, which corresponds to a user-specified material. Each face group F_i represents either an interface or boundary of the computational domain. As such, the user-supplied interface and boundary conditions for all interfaces and boundaries in the domain are applied through the defined face groups. Each face f_{jk} in face group F_i must be oriented in the same direction such that the left and right region on each face when traversing from node j to node k is equal to F_i's respective left l_i and right r_i region index. Additionally, all face groups are required to form a 'water-tight' geometry by establishing closed loops around each region. FIGURE 2-2 shows an example of the geometry representation for a multi-region interface domain. Here, F_2 consists of the set of faces: f_{23}, f_{34}, and f_{45}, and region information: $l_2 = 1$, $r_2 = 2$, that uniquely defines the interface between domain Ω_1 and Ω_2. Alternatively, boundaries define the interface between a valid region and a null region. To signify

a null region, the corresponding region index is set to zero. For example, F_3 in FIGURE 2-2 has an associated left and right region index: $l_3 = 1$ and $r_3 = 0$, since the left domain is Ω_1 and the right domain is null. Note that face group directions can be defined in either direction, as long as its set of faces are oriented in the same direction, and its region information is consistent with the left and right domains.

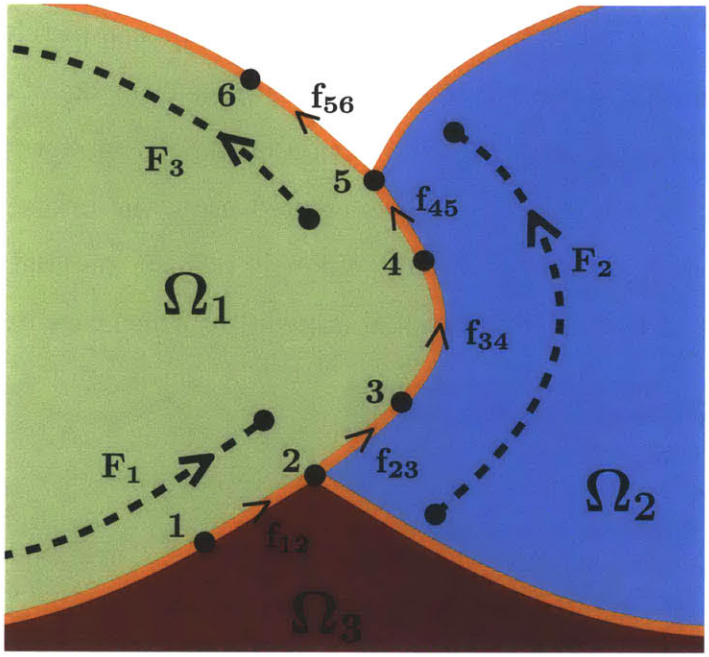


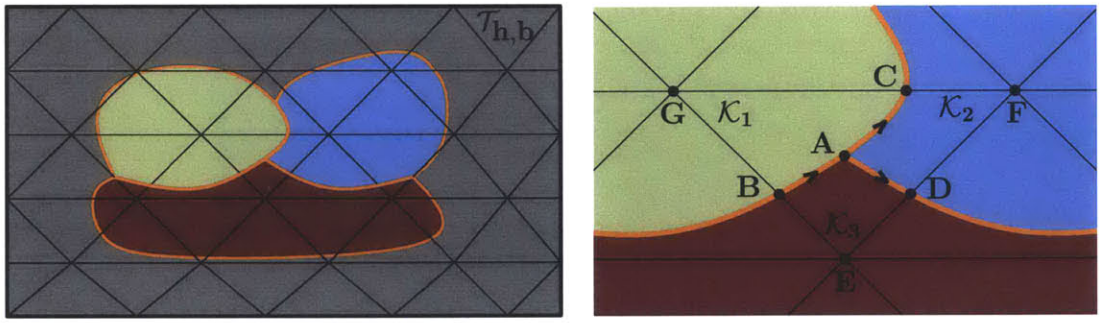
FIGURE 2-2: Multi-region geometry representation

To approximate curvature in the geometry, cubic splines are used to represent all boundaries and interfaces defined by the face groups. At geometric corners or interface junctions where multiple regions intersect at a node, multiple splines are required to represent the geometry. Once all face groups are represented by splines, the geometry is used to generate a cut-cell mesh.

2.2 Cut-Cell Technique

The grid generation process for a cut-cell mesh begins with a *background mesh*, denoted as $\mathcal{T}_{h,b}$ consisting of elements $\mathcal{K}_b$, and an embedded geometry definition. Both

the background mesh and embedded geometry definitions are not required to conform, and are instead used to efficiently construct cut-cell meshes consisting of multiple complex regions. Though conformity is not required, the geometry definition is required to be fully embedded within the background mesh. FIGURE 2-3(a) shows an example of an embedded domain within a non-conforming background mesh $\mathcal{T}_{h,b}$. To create the cut-cell mesh, the geometry definition is intersected with $\mathcal{T}_{h,b}$, and $\mathcal{T}_{h,b}$ is separated or 'cut' into individual parts ($\mathcal{T}_h^{(i)}$) that reside completely within a single region Ω_i. For instance, FIGURE 2-3(b) illustrates three elements of different regions, $\mathcal{K}_1$, $\mathcal{K}_2$, and $\mathcal{K}_3$, resulting from the intersection of the geometry with a single background element. Once a cut-cell mesh is created using the intersection algorithm, each element is either associated with a uniquely defined material or considered a null element (if outside the computational domain) in which case it is disregarded.



(a) Non-geometry-conforming background mesh (grey signifies null region)

(b) Example of elements cut from a background mesh

FIGURE 2-3: Background and cut mesh

2.3 Multi-Region Intersection Algorithm

The generation of multi-region cut-cell meshes presented here is an extension of the intersection algorithm developed by Modisette [51], which creates cut elements on only one side of an interface to define a domain boundary. The algorithm also leverages the implementation provided by Sun [71] where cut elements are formed on both sides of an interface, creating two distinct domains. However, when generating cut meshes for n-regioned domains, *junction* points, which join multiple regions (ie. point

$\mathcal{A}$ in FIGURE 2-3(b)), are encountered, warranting additional region information to precisely define the topology of the cut mesh. Hence, the algorithm utilizes region information (l_i, r_i) that is specified for each face group F_i in order to identify and create cut elements of unique regions.

To generate the cut mesh topology, the intersection algorithm starts by finding all intersection points between the set of splines, indexed $S_j = 1, 2, ..., n_{\text{spline}}$, and background mesh edges (ie. points $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ in FIGURE 2-3(b)). For each spline, S_j, a bounding box method is used to detect all background mesh edges that are candidates for an intersection. In particular, the global coordinates of both the spline endpoints and any non-endpoint extrema (determined by solving a quadratic equation) are determined, and the maximum distance between the set of coordinates defines the diagonal endpoints of the bounding box. The box is expanded slightly with a tolerance of $\max(0.01d, 100\text{MP})$ where d is the box diagonal length and MP represents machine precision, to ensure no intersections are missed. Then, the set of edges within the bounding box are determined by testing whether the edge endpoints are both within the box or, if not, whether the edges intersect the lines of the bounding box. For each edge within the bounding box, the intersection points are determined by solving a cubic-root problem with double precision arithmetic. Though it is possible to erroneously detect an intersection due to floating point errors, the intersection between a spline and an edge is performed only once in order to prevent an inconsistent topology based on the ordering of the intersection calculations.

Junction points are determined by finding the nodes that share multiple face groups. Since multiple spline segments meet at junction points, the spline index, S_j, and spline end parameter, s_{end}, of each joined spline are stored on the corresponding junction point. Both intersection and junction points, along with all other background grid points (ie. points $\mathcal{E}$,$\mathcal{F}$, and $\mathcal{G}$), are stored and referred to as *zerod* objects. Background faces that are "cut" by the intersection points are then formed into separate faces called "cut" faces (ie. face $\overrightarrow{\mathcal{GB}}$, $\overrightarrow{\mathcal{BE}}$, etc. in FIGURE 2-3(b)). Additionally, spline segments that are "cut" by the intersection points are also formed into separate faces

called "embedded" faces (ie. face $\overrightarrow{\mathcal{BA}}$, $\overrightarrow{\mathcal{AD}}$, and $\overrightarrow{\mathcal{AC}}$). These faces, along with all other uncut background faces, or "whole" faces, are stored and referred to as *oned* objects. TABLE 2.2 and TABLE 2.3 outline the different *zerod* and *oned* object types along with their stored information, while FIGURE 2-4 shows an illustration of both object types on a cut mesh.

zerod **object**	**Stored Information**
Grid Point	– Background grid node – Single region index m
Junction Point	– $\{S_j\}$, set of spline indices of joined spline segments – $\{s_{\text{end}}\}$, set of spline parameters of joined spline ends
Spline-face intersection	– Face index of background face – S_j, spline index of background spline segment – s_{int}, spline parameter of intersection

TABLE 2.2: Information stored for *zerod* objects

oned **object**	**Stored Information**
Embedded Face	– S_j, spline segment index – Index of *zerod* endpoints – Background element index – Left and right region index, l and r
Cut Face	– Background face index – Index of *zerod* endpoints – Single region index m
Whole Face	– Background face index – Index of *zerod* endpoints – Single region index m
Null Face	None, this face is not relevant for solving, but is used in forming the null vertex layer (See SECTION 3.5.2)

TABLE 2.3: Information stored for *oned* objects

Oned objects are combined into elements by forming closed loops around single regions. Each loop that is formed is referred to as a *twod* object, which defines a new cut element and its associated region. To create all loops within an intersected background element, each "embedded" face must be traversed twice in opposite directions. The multi-regioned loop generation algorithm is detailed in ALGORITHM 1.

Algorithm: *Multi-Regioned Loop Generation*

for *each background element with an embedded geometry intersection* $\mathcal{T}^b_{cut}$ **do**
- Set the *used* counter of all embedded faces to zero: $e_j.used = 0$;
- **while** *not all embedded faces in* $\mathcal{T}^b_{cut}$ *are used twice* **do**
  - Select an embedded face e_0 from a list of valid *oned* objects, OneD, associated with the background element that has not been used twice;
  - **if** $e_0.used = 0$ **then**
    - (e_0 unused)
    - Start a loop by traversing from the initial *zerod* on the embedded face, setting $z_{start} = \text{OneD}[e_0].z_0$, and $z_{cur} = \text{OneD}[e_0].z_1$;
    - Set current region $r_{cur} = OneD[e_0].l$;
  - **else**
    - (e_0 used once)
    - Start a loop by traversing opposite to the direction previously used, setting $z_{start} = \text{OneD}[e_0].z_1$, and $z_{cur} = \text{OneD}[e_0].z_0$;
    - Set current region $r_{cur} = OneD[e_0].r$;
  - **end**
  - Set $e_{cur} = e_0$ and $z_{cur} = \text{OneD}[e_0].z_1$;
  - **while** $(z_{cur}! = z_{start})$ **do**
    - Add e_{cur} to the Loop and increment used counter: $e_{cur}.used$++;
    - **if** z_{cur} *is a junction* zerod **then**
      - **for** *all embedded faces* $e^e_j \in$ *OneD used less than twice* **do**
        - **if** $(OneD[e^e_j].z_0 == z_{cur})$ && $(OneD[e^e_j].l == r_{cur})$ **then**
          - Set $e_{cur} = e^e_j$;
          - Set z_{cur} to $\text{OneD}[e^e_j].z_1$;
        - **else**
          - **if** $(OneD[e^e_j].z_1 == z_{cur})$ && $(OneD[e^e_j].r == r_{cur})$ **then**
            - Set $e_{cur} = e^e_j$;
            - Set z_{cur} to $\text{OneD}[e^e_j].z_0$;
          - **end**
        - **end**
      - **end**
    - **else**
      - **for** *all faces* $e_j \in$ *OneD that is either an embedded face used less than twice or an unused cut face* **do**
        - **if** $(OneD[e_j].z_0 == z_{cur})$ **then**
          - Set $e_{cur} = e_j$;
          - Set z_{cur} to $\text{OneD}[e_j].z_1$;
        - **else**
          - **if** $(OneD[e_j].z_1 == z_{cur})$ **then**
            - Set $e_{cur} = e_j$;
            - Set z_{cur} to $\text{OneD}[e_j].z_0$;
          - **end**
        - **end**
      - **end**
    - **end**
  - Assign region r_{cur} to the cut faces in the loop and to the newly constructed *twod*;
- **end**

end

Algorithm 1: Multi-region loop generation algorithm

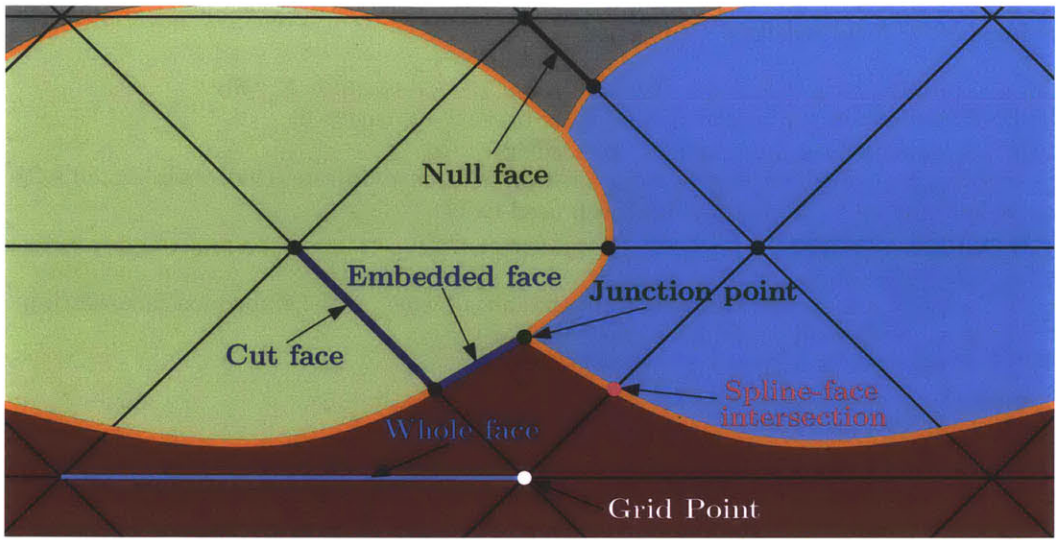
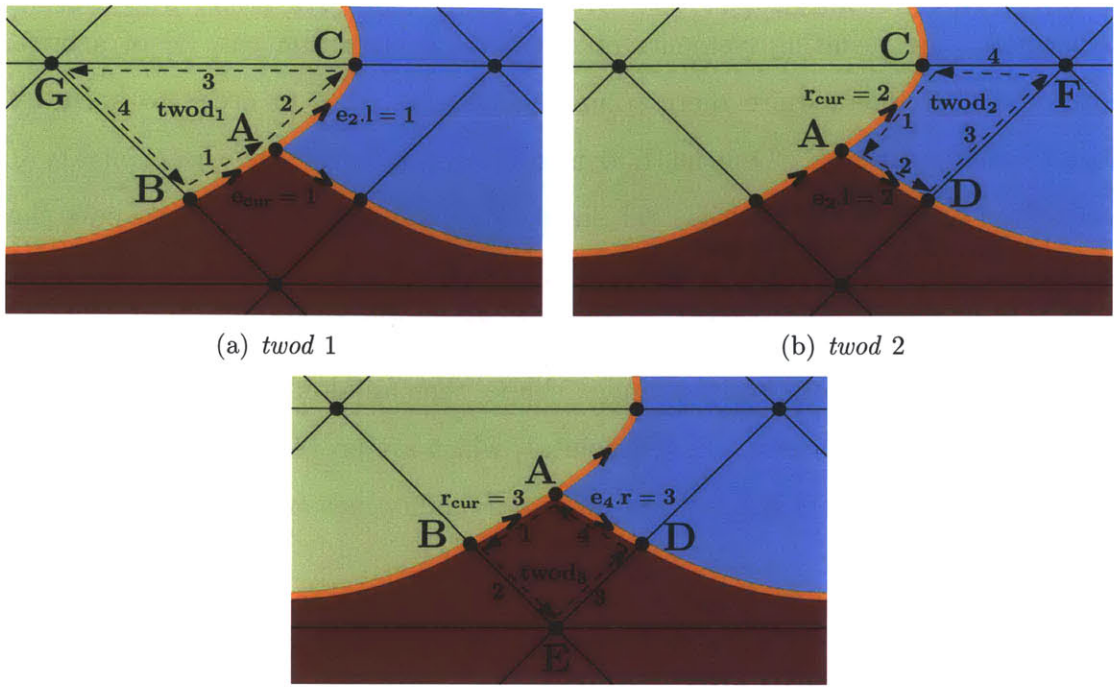


FIGURE 2-4: Example of *zerod* and *oned* objects in a multi-region intersection

Since the algorithm can start with any embedded face e_0, the order of the loop generation is not unique, though the algorithm guarantees a precise definition of the element topology when completed. As an example, FIGURE 2-5 shows the progression of the loop generation algorithm as the *twod* objects in a multi-regioned background element are constructed. To create $twod_1$, the algorithm starts with an embedded face, $\overrightarrow{\mathcal{BA}}$, and steps to junction point $\mathcal{A}$. At the junction point, all *oned* objects within the element are looped over, and the embedded face corresponding to the current region, which in this case is $\overrightarrow{\mathcal{AC}}$ with $l = 1$, is selected. This is continued until the closed loop is formed. Next, the cut faces, $\overrightarrow{\mathcal{CG}}$ and $\overrightarrow{\mathcal{GB}}$, and the *twod* object, $\mathcal{BACG}$, is assigned with the region index $m = r_{cur} = 1$. This entire process is repeated until all embedded faces have been traversed twice, as seen in the formation of $twod_2$ and $twod_3$. Note that the algorithm, once initialized with a valid *oned*, always forms loops existing only within the computational domain.

At this point, all elements with a cut face belong to a specific region, though region information still needs to be defined on all other non-cut faces and elements within the domain. This is required in order to distinguish the choice of physics for residual evaluation on faces and elements of different regions. The propagation of region information is performed by an algorithm that uses the embedded face group's left



(a) *twod* 1

(b) *twod* 2

(c) *twod* 3

FIGURE 2-5: Formation of *twod* types in a multi-regioned cut element

and right region information and mesh connectivity to spread the region to all nodes and faces in the cut mesh. Once this is completed, the defined region of a non-cut element is trivially set by the surrounding face region information. The cut-mesh is complete once all features within the domain has region identification.

2.3.1 Merging and Quadrature

With this method, it is possible to create an arbitrarily small area ratio between neighboring elements. This large difference in area negatively affects the linear system condition number and inhibits solution accuracy. To mitigate this issue, two elements of the same region with a large area ratio are merged into a single larger element as proposed by Modisette [51] and demonstrated by Sun [71]. This process is carried through for all neighbors with high area ratios, though is not used as part of local solves on cut elements for error synthesis (See SECTION 3.5.2).

Furthermore, the resulting elements of the cut-cell algorithm can be of arbitrary shape, and therefore require more robust quadrature rules in order to accurately calculate residual terms. One method is to attempt to convert the arbitrary cut element into a triangle or quadrilateral, since the arbitrary shapes typically have three or four edges. This allows the cut element to be referenced to a master element so that standard integration rules can be applied [51]. If a cut element cannot be converted to a canonical element due to its complex shape, then an alternative quadrature rule must be applied. In this case, "magic points", which are proven to be asymptotically the same as Fekete points but with an improved quality measure as demonstrated by Sun [71], are employed.

2.3.2 Multi-Region Simulation

This work focuses on designing a tool to efficiently create and solve CHT simulations. The same solver that would be used in a single region case is used for the multi-region case, though the residual calculation changes depending on which region is being calculated. In order to perform the correct calculations based on the physics involved, a region *key* listing all elements and corresponding regions is defined. This allows for the correct function calls and material properties in each element when evaluating the global residual.

Chapter 3

Discretization, Error Estimation and Output-Based Adaptation

This chapter first summarizes the discontinuous Galerkin (DG) method for general conservation laws. Then the dual-weighted residual method, proposed by Becker and Rannacher [11, 12], is shown as a means for output error estimation. Lastly, the metric optimization framework for mesh adaptation, proposed by Yano and Darmofal [77] and extended to handle cut cells by Sun [71], is presented.

3.1 Governing Equations

Let $\Omega \in \mathbb{R}^d$ be an arbitrary, bounded domain in a d−dimensional space. The strong form of a general time-dependent conservation law in the domain, Ω, can be expressed as:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathcal{F}^i(\mathbf{u}, \mathbf{x}, t) - \nabla \cdot \mathcal{F}^v(\mathbf{u}, \nabla \mathbf{u}, \mathbf{x}, t) = \mathcal{S}(\mathbf{u}, \nabla \mathbf{u}, \mathbf{x}, t), \quad \forall \mathbf{x} \in \Omega, \ t \in I \quad (3.1)$$

with initial condition:

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega$$

and boundary conditions:

$$\mathcal{B}(\mathbf{u}, \mathcal{F}^v(\mathbf{u}, \nabla\mathbf{u}, \mathbf{x}, t) \cdot \mathbf{n}, \mathbf{x}, t; BC) = 0 \quad \forall \mathbf{x} \in \partial\Omega, \;\; t \in I$$

where $\mathbf{u}(\mathbf{x}, t) : \mathbb{R}^{m_r}$ is the m_r-state solution vector in region r, $\mathcal{F}^i(\mathbf{u}, \mathbf{x}, t) : \mathbb{R}^{m_r \times d}$ is the inviscid flux, $\mathcal{F}^v(\mathbf{u}, \nabla\mathbf{u}, \mathbf{x}, t) : \mathbb{R}^{m_r \times d}$ is the viscous flux, $\mathcal{S}(\mathbf{u}, \nabla\mathbf{u}, \mathbf{x}, t) : \mathbb{R}^{m_r}$ is the source term, and $\mathcal{B}$ imposes the boundary condition. Note that the state rank and residual term definitions are region dependent. For solving a conjugate heat transfer solution, both the fluid and the solid governing equations are expressed in the general conservative form and solved simultaneously. In this work, both the Navier-Stokes and Reynolds-Averaged Navier-Stokes equations are coupled with the heat equation for conjugate simulation. The formulation of these governing equations are detailed in APPENDIX A.

3.2 Discontinuous Galerkin Discretization

Since the governing equations for fluid flow and heat conduction can be expressed in the general conservative form, the DG discretization can be applied to the entire conjugate domain, regardless of the number or type of regions. This allows for finite element discretizations to extend to multi-regioned problems that are governed by PDE's of the same general form.

For the discontinuous Galerkin discretization, let $\mathcal{T}_h$ be a triangulation of the domain Ω with elements, κ. Also define a function space $\mathcal{V}_{h,p}$ as:

$$\mathcal{V}_{h,p} \equiv \{\mathbf{v} \in \left(L^2(\Omega)\right)^{m_r} : \mathbf{v}|_\kappa \in \left(\mathcal{P}^p(\kappa)\right)^{m_r}, \forall \kappa \in \mathcal{T}_h\}, \tag{3.2}$$

where $\mathcal{P}^p$ represents the solution space of p-th degree polynomials on a physical element κ. Taking the product of EQUATION 3.1 with a test function $\mathbf{v}_{h,p} \in \mathcal{V}_{h,p}$, and integrating by parts yields the weak formulation of the governing equation. Solving the weak formulation finds a solution $\mathbf{u}_{h,p} \in \mathcal{V}_{h,p}$ such that:

$$\sum_{\kappa \in \mathcal{T}_h} \int_\kappa \mathbf{v}_{h,p}^T \frac{\partial \mathbf{u}_{h,p}}{\partial t} + \mathcal{R}_{h,p}(\mathbf{u}_{h,p}, \mathbf{v}_{h,p}) = 0 \quad \forall \mathbf{v}_{h,p} \in \mathcal{V}_{h,p}. \tag{3.3}$$

where the weighted residual $\mathcal{R}_{h,p}$ is comprised of inviscid ($\mathcal{R}^i$), viscous ($\mathcal{R}^v$), and source ($\mathcal{R}^s$) discretization terms:

$$\mathcal{R}_{h,p}(\mathbf{w}_{h,p}, \mathbf{v}_{h,p}) = \mathcal{R}^i_{h,p}(\mathbf{w}_{h,p}, \mathbf{v}_{h,p}) + \mathcal{R}^v_{h,p}(\mathbf{w}_{h,p}, \mathbf{v}_{h,p}) + \mathcal{R}^s_{h,p}(\mathbf{w}_{h,p}, \mathbf{v}_{h,p}) \tag{3.4}$$

3.2.1 Inviscid Discretization

The DG discretization of the inviscid term is given by:

$$\begin{aligned} \mathcal{R}^i_{h,p}(\mathbf{w}, \mathbf{v}) = &- \sum_{\kappa \in \mathcal{T}_h} \int_\kappa \nabla \mathbf{v}^T \cdot \mathcal{F}^i(\mathbf{w}) \\ &+ \sum_{f \in \Gamma_i} \int_f (\mathbf{v}^+ - \mathbf{v}^-)^T \mathcal{H}(\mathbf{w}^+, \mathbf{w}^-; \mathbf{n}^+) \\ &+ \sum_{f \in \Gamma_b} \int_f \mathbf{v}^{+^T} \mathcal{H}^b(\mathbf{w}^+, \mathbf{u}^b(\mathbf{w}^+; BC); \mathbf{n}^+) \\ &+ \mathcal{R}^{inv}_\Sigma(\mathbf{w}, \mathbf{v}) \end{aligned} \tag{3.5}$$

where $(\cdot)^+$ and $(\cdot)^-$ denote trace values taken from opposite sides of a face f, $\mathbf{n}^+$ is the normal vector pointing from the (+) side to the (-) side, $\mathcal{H}$ and $\mathcal{H}^b$ are numerical flux functions on interior and boundary faces respectively, $\mathbf{u}^b$ is the boundary state constructed from the interior state and a specified boundary condition, and Γ_i, Γ_b, and Σ are the interior, boundary, and interface faces, respectively. $\mathcal{R}^{inv}_\Sigma(\mathbf{w}, \mathbf{v})$ is the equation-specific inviscid interface residual term that is defined for the coupled Navier-Stokes and heat equation interface in SECTION 4.1, and for the coupled RANS and heat equation interface in SECTION 5.1. In this work, the numerical flux function $\mathcal{H}$ uses the Roe flux [66] to approximate the Riemann problem. The inviscid boundary flux $\mathcal{H}^b$, is calculated by evaluating the flux at a boundary state, $\mathbf{u}_b$, which is a function of both the interior state, $\mathbf{w}^+$, and a user-specified boundary condition, BC.

3.2.2 Viscous Discretization

The viscous terms are discretized using the second method of Bassi and Rebay (BR2) [10]. For compactness, the jump $[[\cdot]]$ and average $\{\cdot\}$ operators are used. For a scalar s and vector $\mathbf{v}$, the jump and averages on interior faces are defined as:

$$\{s\} = \frac{1}{2}(s^+ + s^-), \qquad \{\mathbf{v}\} = \frac{1}{2}(\mathbf{v}^+ + \mathbf{v}^-),$$
$$[[s]] = (s^+\mathbf{n}^+ + s^-\mathbf{n}^-), \quad [[\mathbf{v}]] = (\mathbf{v}^+ \cdot \mathbf{n}^+ + \mathbf{v}^- \cdot \mathbf{n}^-)$$

and on boundary faces as:

$$\{s\} = s^+ \qquad \{\mathbf{v}\} = \mathbf{v}^+$$
$$[[s]] = s^+\mathbf{n}^+ \quad [[\mathbf{v}]] = \mathbf{v}^+ \cdot \mathbf{n}^+$$

The viscous discretization is:

$$\begin{aligned}\mathcal{R}^v_{h,p}(\mathbf{w},\mathbf{v}) \equiv & \sum_{\kappa\in\mathcal{T}} \int_\kappa \nabla\mathbf{v}^T \cdot (\mathcal{A}(\mathbf{w})\nabla\mathbf{w}) \\ & - \sum_{f\in\Gamma_i} \int_f [[[\mathbf{w}]]^T \cdot \{\mathcal{A}^T(\mathbf{w})\nabla\mathbf{v}\} + [[\mathbf{v}]]^T \cdot (\{\mathcal{A}(\mathbf{w})(\nabla\mathbf{w} - \eta_f \mathbf{r}_f([[\mathbf{w}]]))\})] \\ & - \sum_{f\in\Gamma_b} \int_f \left[(\mathbf{w}^+ - \mathbf{u}^b)^T \mathcal{A}_b^T \nabla\mathbf{v}^+) \cdot \mathbf{n}^+ + \mathbf{v}^{+^T} \left(\mathcal{A}_b \left(\nabla\mathbf{u}^b - \eta_f \mathbf{r}_f^b(\mathbf{w}^+ - \mathbf{u}^b)\right)\right) \cdot \mathbf{n}^+\right] \\ & - \mathcal{R}^{visc}_\Sigma(\mathbf{w},\mathbf{v})\end{aligned} \tag{3.6}$$

where $\mathbf{u}^b(\mathbf{w}^+, BC)$, $\mathcal{A}_b(\mathbf{u}^b; BC)$, and $\nabla\mathbf{u}^b(\nabla\mathbf{w}^+; BC)$ are chosen to specify the boundary viscous flux, $\mathbf{r}_f$ and $\mathbf{r}_f^b$ are the lifting operators on an interior and boundary face respectively, and η_f is a stabilizing coefficient. $\mathcal{R}^{visc}_\Sigma(\mathbf{w},\mathbf{v})$ is the equation-specific inviscid interface residual term that is defined for the coupled Navier-Stokes and heat equation interface in SECTION 4.1, and for the coupled RANS and heat equation interface in SECTION 5.1. For this work, the stabilization parameter is set conservatively to $\eta_f = 6$ (a value $\eta_f > 3$ implies stability of the BR2 discretization for triangular

meshes). The lifting operators, which are used to penalize jumps in the solution, are defined in the following way: for every face f, find $\mathbf{r}_f \in [\mathcal{V}_{h,p}]^d$ such that for interior faces

$$\sum_{\kappa \in \mathcal{T}_h} \int_\kappa \tau^T \cdot \mathbf{r}_f(\phi) = \int_f \phi^T \cdot \{\tau\} \quad \forall \tau \in [\mathcal{V}_{h,p}]^d \tag{3.7}$$

and for boundary faces

$$\sum_{\kappa \in \mathcal{T}_h} \int_\kappa \tau^T \cdot \mathbf{r}_f^b(\phi) = \int_f \phi^T \tau^+ \cdot \mathbf{n}^+ \quad \forall \tau \in [\mathcal{V}_{h,p}]^d \tag{3.8}$$

3.2.3 Source Discretization

The discretization of the source term uses the formulation shown by Bassi *et al.* [6], which uses a lifting operator to solve for the state gradient. More specifically:

$$\mathcal{R}_{h,p}^s(\mathbf{w}, \mathbf{v}) = \sum_{\kappa \in \mathcal{T}_h} \int_\kappa \mathbf{v}^T \mathcal{S}(\mathbf{w}, \nabla \mathbf{w} + \mathbf{r}_g(\mathbf{w})) \tag{3.9}$$

where the global lifting operator is $\mathbf{r}_g : \mathcal{V}_{h,p} \to [\mathcal{V}_{h,p}]^d$ such that:

$$\mathbf{r}_g(\mathbf{w}) = \sum_{f \in \Gamma_i} \mathbf{r}_f([[\mathbf{w}]]) + \sum_{f \in \Gamma_b} \mathbf{r}_f((\mathbf{w}^+ - \mathbf{u}^b)\mathbf{n}^+) \tag{3.10}$$

where $\mathbf{r}_f$ is the local, face-wise lifting operator. Oliver [57] proved this method to be asymptotically dual-consistent, allowing for super-convergence of an output of interest.

3.3 Solution Technique

With the choice of a basis in the function space $\mathcal{V}_{h,p}$, a solution to the discrete equation can be obtained. Specifically, the steady discrete equation can be expressed as a system of algebraic equations, which allows for finding $\mathbf{U}$ such that:

$$\mathbf{R}_s(\mathbf{U}) = 0 \tag{3.11}$$

where $\mathbf{R}_s(\mathbf{U})$ is the discrete spatial residual vector. This equation is solved using a pseudo-time continuation and backward Euler time integration. Given an initial discrete solution, $\mathbf{U}^n$, a new solution after one time step, $\mathbf{U}^{n+1}$, is determined by solving

$$\mathbf{R}_t(\mathbf{U}^{n+1}) \equiv \mathbf{M}^t(\mathbf{U}^{n+1} - \mathbf{U}^n) + \mathbf{R}_s(\mathbf{U}^{n+1}) = 0 \tag{3.12}$$

where $\mathbf{R}_t$ is the pseudo-unsteady residual, and $\mathbf{M}^t$ is the mass matrix weighted by a local elemental time step Δt_κ. This time step is calculated based on a global CFL number defined as:

$$\mathrm{CFL} = \frac{\Delta \mathrm{t}_\kappa \lambda_\kappa}{\mathrm{h}_\kappa} \tag{3.13}$$

where h_κ is a measure of the element's size, and λ_κ is the maximum characteristic speed within the element κ. At each time step, Newton's method is used to solve EQUATION 3.12 such that:

$$\mathbf{U}^{n+1} - \mathbf{U}^n \approx \Delta\mathbf{U} \equiv -\left(\mathbf{M}^t + \left.\frac{\partial \mathbf{R}_s}{\partial \mathbf{U}}\right|_{\mathbf{U}^n}\right)^{-1} \mathbf{R}_s(\mathbf{U}^n) \tag{3.14}$$

The pseudo-time is advanced until the spatial residual's 2-norm $||\mathbf{R}_s(\mathbf{U}^{n+1})||_2$ is less than a user-specified tolerance. Additionally, the CFL number is updated and strategically limited on each iteration to improve the robustness of the solver. This is done by preventing large updates to select states, and by using a line search that controls the unsteady residual, $\mathbf{R}_t$ [51].

EQUATION 3.14 is solved using a restarted GMRES algorithm [68, 69], which is preconditioned with an in-place block-ILU(0) factorization [28] with minimum discarded fill ordering [61], unless otherwise stated.

3.4 Output Error Estimation

The estimation of output errors relies on the dual-weighted resdual (DWR) method proposed by Becker and Rannacher [11, 12]. This method utilizes the dual problem associated with an output of interest J in order to tie local primal residuals to the output error. In other words, it allows for an approximation of the local output error from numerical solutions to the primal and dual problem. First, let the output of interest be a function of the state: $J = \mathcal{J}(u)$ where $u \in \mathcal{V}$ is the exact solution to the governing PDE, and $\mathcal{J}(\cdot) : \mathcal{V} \to \mathbb{R}$ is the output functional. For conjugate heat transfer, $\mathcal{J}$ is typically an integral quantity on surfaces (heat flux, mass flow) or in the domain (average temperature). Once a discrete DG solution $u_{h,p}$ that satisfies

$$\mathcal{R}_{h,p}(u_{h,p}, v) = 0, \quad \forall v \in \mathcal{V}_{h,p} \tag{3.15}$$

is obtained, the desired output is estimated by

$$J_{h,p} = \mathcal{J}_{h,p}(u_{h,p}), \tag{3.16}$$

where $\mathcal{J}_{h,p}$ is the discrete output functional that maintains dual consistency [57, 50]. The true output error is then expressed as

$$\mathcal{E}_{true} \equiv \mathcal{J} - \mathcal{J}_{h,p} = -\mathcal{R}_{h,p}(u_{h,p}, \psi) \tag{3.17}$$

where the adjoint ψ is the sensitivity of the output $\mathcal{J}$ with respect to perturbations in the solution residual $R_{h,p}$. More formally, the adjoint solution $\psi \in \mathcal{W} \equiv \mathcal{V} + \mathcal{V}_{h,p}$ satisfies

$$\overline{\mathcal{R}}'_{h,p}[u, u_{h,p}](w, \psi) = \overline{\mathcal{J}}'_{h,p}[u, u_{h,p}](w), \quad \forall w \in \mathcal{W}. \tag{3.18}$$

where $\overline{\mathcal{R}}'_{h,p}[u, u_{h,p}] : \mathcal{W} \times \mathcal{W} \to \mathbb{R}$ and $\overline{\mathcal{J}}'_{h,p}[u, u_{h,p}] : \mathcal{W} \to \mathbb{R}$ are the mean-value linearizations defined as

$$\overline{\mathcal{R}}'_{h,p}[u, u_{h,p}](w, v) \equiv \int_0^1 \mathcal{R}'_{h,p}[\theta u + (1-\theta)u_{h,p}](w, v)d\theta \tag{3.19}$$

$$\overline{\mathcal{J}}'_{h,p}[u, u_{h,p}](w) \equiv \int_0^1 \mathcal{J}'_{h,p}[\theta u + (1-\theta)u_{h,p}](w)d\theta \tag{3.20}$$

Here, $\mathcal{R}'_{h,p}[z](\cdot,\cdot)$ and $\mathcal{J}'_{h,p}[z](\cdot)$ denote the Fréchet derivative of $\mathcal{R}_{h,p}(\cdot,\cdot)$ and $\mathcal{J}_{h,p}(\cdot)$ with respect to the first argument evaluated about the state z. Since the adjoint solution ψ generally cannot be computed from EQUATION 3.18, it is instead approximated by $\psi_{h,p+1}$ in an enriched space $\mathcal{V}_{h,p+1} \supset \mathcal{V}_{h,p}$, computed from a linearization about $u_{h,p}$:

$$\mathcal{R}'_{h,p+1}[u_{h,p}](v, \psi_{h,p+1}) = \mathcal{J}'_{h,p+1}[u_{h,p}](v), \quad \forall v \in \mathcal{V}_{h,p+1}. \tag{3.21}$$

Thus, the output error can be estimated by

$$\mathcal{E}_{true} \approx -\mathcal{R}_{h,p}(u_{h,p}, \psi_{h,p+1}). \tag{3.22}$$

The error in approximating the adjoint in an enriched space could become significant for coarse meshes, though Yano [77] showed that it is sufficiently accurate for the purpose of mesh adaptation. Additional details of the method and derivation can be found in several reviews [34, 44, 31].

3.4.1 Error Localization

In order to achieve error gradients for mesh optimization, a localized error estimate is calculated on every element κ via

$$\eta_\kappa \equiv |\mathcal{R}_{h,p}(u_{h,p}, \psi_{h,p+1}|_\kappa)| \tag{3.23}$$

This localized error can be interpreted as a weighted product of the local primal error

and the local adjoint error, implying that accuracy in the error estimation requires accurate solutions to both the primal and dual problem.

Summing the local error estimate across all elements gives an error estimate for the output $\mathcal{J}$:

$$\mathcal{E} \equiv \sum_{\kappa \in \mathcal{T}_h} \eta_\kappa, \tag{3.24}$$

which can be used for an error model in mesh adaptation.

3.5 Output-Based Mesh Adaptation

The goal of mesh adaptation is to optimize a triangulation $\mathcal{T}_h$ in order to achieve greater accuracy in output predictions. Since many aerospace analyses involve physical features that warrant highly anisotropic meshes (ie. boundary layers, shocks, etc.), it is important for mesh adaptation schemes to handle and manipulate anisotropic information for simplex elements. One method of representing this information is through the formulation of a metric tensor $\mathcal{M}_\kappa$, which is a symmetric positive definite matrix that encodes the size and orientation of an element κ [73].

Though a single metric tensor $\mathcal{M}_\kappa$ can be discretely defined for each element in a mesh, a continuous metric field $\{\mathcal{M}(\mathbf{x})\}_{\mathbf{x}\in\Omega}$ can also be created to facilitate the optimization process. FIGURE 3-1 illustrates the mapping between a discrete mesh and a continuous metric tensor field. For a given mesh, there exists a unique conforming metric field; however, given a metric field, several non-unique discrete meshes can conform. Furthermore, the metric field, as proven by Yano [77], has a direct influence over the output error for the DG discretization. Thus, the error in heat flux for a conjugate heat transfer problem, for example, can be controlled by manipulating the metric field.

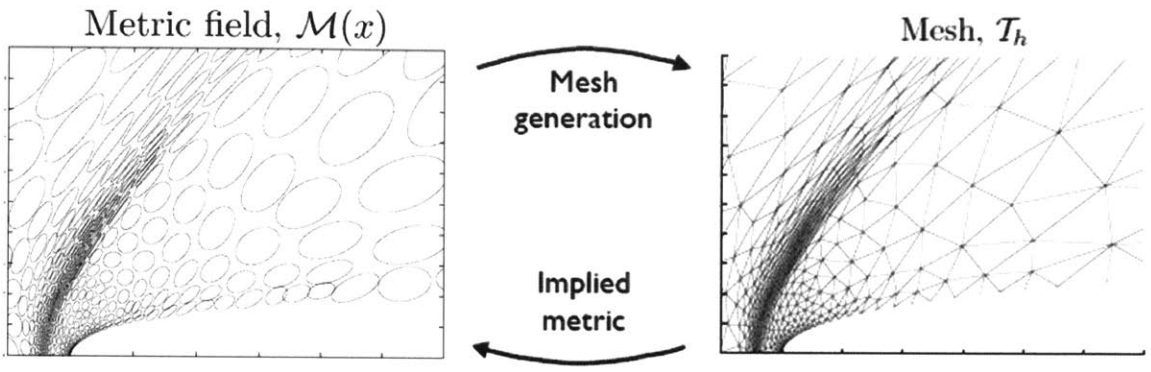


FIGURE 3-1: Example of mesh to continuous metric field mapping

In this work, the metric optimization framework proposed by Yano and Darmofal [77] and extended to handle cut-cell meshes by Sun [71] is applied to multi-region conjugate problems. The Bidimensional Anisotropic Mesh Generator (BAMG) [15, 37], developed by INRIA, is selected for generating all two-dimensional metric conforming meshes, and a gradient based method, described by Kudo [39], is chosen for optimizing the metric field.

3.5.1 Mesh Optimization via Error Sampling and Synthesis

In this section, a short review of the Mesh Optimization via Error Sampling and Synthesis (MOESS) algorithm developed by Yano and Darmofal [77] and the extension to cut cells developed by Sun [71] is presented.

The goal of mesh adaptation is to manipulate the triangulation $\mathcal{T}_\kappa$ in order to achieve higher accuracy in output predictions. This can be done by finding the optimal triangulation $\mathcal{T}_h^*$ where

$$\mathcal{T}_h^* = \arg\inf_{\mathcal{T}_h} \mathcal{E}(\mathcal{T}_h) \quad \text{subject to} \quad \mathcal{C}(\mathcal{T}_\mathrm{h}) \leq \mathrm{dof}_\mathrm{target} \tag{3.25}$$

where $\mathcal{E}(\cdot)$ is the error functional, and $\mathcal{C}(\cdot)$ measures the cost or number of degrees of freedom (DOF) within $\mathcal{T}_h$. Since the triangulation $\mathcal{T}_h$ is defined by nodal coordinates and connectivity, the above discrete optimization problem is generally intractable. Using Loseille and Alauzet's proposed continuous relaxation of the optimization problem

[47], the continuous metric field, $\boldsymbol{\mathcal{M}} \equiv \{\mathcal{M}(\mathbf{x})\}_{\mathbf{x}\in\Omega}]$, is optimized to find the optimal metric field, $\boldsymbol{\mathcal{M}}^*$, where

$$\boldsymbol{\mathcal{M}}^* \equiv \arg\inf_{\boldsymbol{\mathcal{M}}} \mathcal{E}(\boldsymbol{\mathcal{M}}) \quad \text{subject to} \quad \mathcal{C}(\boldsymbol{\mathcal{M}}) \leq \mathrm{N} \tag{3.26}$$

The cost functional $\mathcal{C}(\boldsymbol{\mathcal{M}})$ is defined as

$$\mathcal{C}(\boldsymbol{\mathcal{M}}) = \int_\Omega c_p \sqrt{\det(\mathcal{M}(\mathbf{x}))} d\mathbf{x} \tag{3.27}$$

where c_p is the reference element degrees of freedom normalized by its size. Note that for multi-disciplinary applications where the number of conservative states between regions are different, the cost model could be adjusted to reflect the differences in degrees of freedom. Here, all regions have equal weightings in the cost model, and extensions to varying weights is left for future work. Since each elemental error contribution η_κ is assumed to be a function of the elemental metric tensor, $\eta_\kappa = \eta_\kappa(\mathcal{M}_\kappa)$, the output functional $\mathcal{E}(\boldsymbol{\mathcal{M}})$ can be approximated by

$$\mathcal{E}(\boldsymbol{\mathcal{M}}) \approx \sum_{\kappa\in\mathcal{T}_h} \eta_\kappa(\mathcal{M}_\kappa) \tag{3.28}$$

where the local error function $\eta_\kappa(\mathcal{M}_\kappa)$ is constructed by a surrogate model due to the analytical expression generally being unknown.

Local Error Sampling

A local error surrogate model is created by detecting the change in error when solving on different split configurations of a particular element. FIGURE 3-2 displays the original configuration $\mathcal{C}_0$, the different types of split configurations $\mathcal{C}_i$ considered, and the respective implied metrics tensors $\mathcal{M}_{\mathcal{C}_i}$. For each configuration $\mathcal{C}_i$, the local problem is solved by setting the boundary fluxes on $\mathcal{C}_i$ assuming the solution on neighboring elements do not change. From this, a localized error estimate associated to $\mathcal{C}_i$ is calculated from the local dual weighted residual:

$$\eta_{\mathcal{C}_i} = |\mathcal{R}_{h,p}(\mathbf{u}_{h,p}^{\mathcal{C}_i}, \psi_{h,p+1}|_\kappa)| \tag{3.29}$$

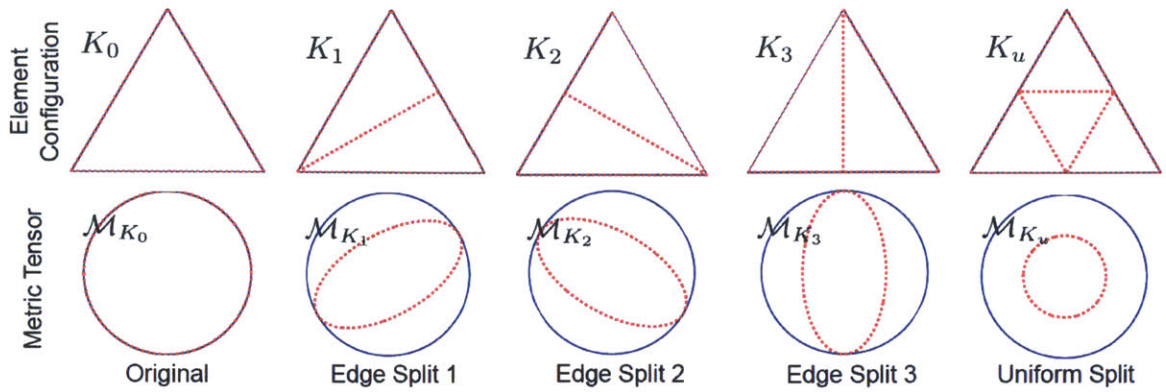


FIGURE 3-2: Example split configurations with respective metric tensors (Yano [77])

Error Model Synthesis and Metric Optimization

The surrogate error model is synthesized once the set of metric-error pairs $(\mathcal{M}_{\mathcal{C}_i}, \eta_{\mathcal{C}_i})|_{i=1}^{n_{\text{config}}}$ are obtained. As proposed by Pennec *et al.* [59], the change in the metric tensor between the original $\mathcal{C}_0$ and new $\mathcal{C}_i$ configuration, or the *step matrix* $\mathcal{S}_{\mathcal{C}_i}$, is determined based on the affine-invariant logarithmic map:

$$\mathcal{S}_{\mathcal{C}_i} \equiv \log\left(\mathcal{M}_{\mathcal{C}_0}^{-1/2}\mathcal{M}_{\mathcal{C}_i}\mathcal{M}_{\mathcal{C}_0}^{-1/2}\right) \quad i = 0, ..., n_{\text{config}} \tag{3.30}$$

where the aggregated metric of the new configuration, $\mathcal{M}_{\mathcal{C}_i}$, is determined by an affine invariant interpolation of the sub-elemental metrics, as shown by Caplan [16]. Additionally, the corresponding change in error is:

$$f_{\mathcal{C}_i} \equiv \log(\eta_{\mathcal{C}_i}/\eta_{\mathcal{C}_0}) \tag{3.31}$$

Once the step pairing samples $(\mathcal{S}_{\mathcal{C}_i}, f_{\mathcal{C}_i})|_{i=0}^{n_{\text{config}}}$ are obtained, a linear model is constructed via least squares: find a semi-definite positive matrix $\mathcal{R}_\kappa \in \text{Sym}_{\text{d}}^+$ such that:

$$\mathcal{R}_\kappa = \arg\min_{\mathcal{R}' \in \text{Sym}_{\text{d}}^+} \sum_{i=1}^{n_{\text{config}}} (\mathcal{R}' : \mathcal{S}_{\mathcal{C}_i} - f_{\mathcal{C}_i})^2 \quad \forall \kappa \in \mathcal{T}_h \tag{3.32}$$

where $A : B \equiv \sum_{i,j=1}^{d} A_{ij}B_{ij}$ and $\mathcal{R}_\kappa$ is called the *rate matrix* corresponding to an

element κ. From this, the local error model is constructed as:

$$\eta_\kappa(\mathcal{S}_\kappa) = \eta_{\mathcal{C}_0} \exp(\mathrm{tr}(\mathcal{R}_\kappa \mathcal{S}_\kappa)). \tag{3.33}$$

Given a local error and cost model, the constrained optimization problem in EQUATION 3.26 is solved using a gradient-based optimization algorithm as shown by Kudo [39].

3.5.2 Extension to Cut Cells

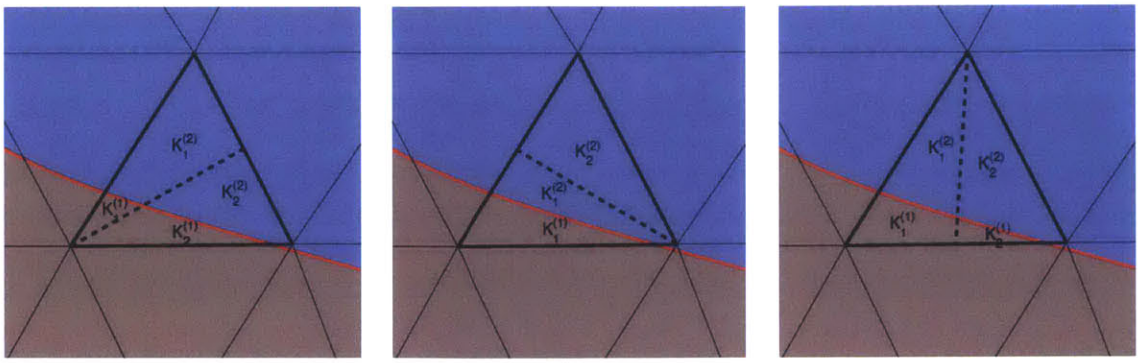
As provided by Sun [71], the same mesh adaptation framework is extended to handle cut cells for interface problems. The cut mesh $\mathcal{T}_h$ is generated from the intersection of a background mesh $\mathcal{T}_{h,b}$ and the curved embedded geometry. Since the continuous metric representation of a triangulation breaks down for a cut mesh $\mathcal{T}_h$, the metric of the background triangulation $\boldsymbol{\mathcal{M}}_b$ is instead optimized in the same manner:

$$\boldsymbol{\mathcal{M}}_b^* = \arg\inf_{\boldsymbol{\mathcal{M}}_b} \mathcal{E}(\boldsymbol{\mathcal{M}}_b) \text{ subject to } \mathcal{C}(\boldsymbol{\mathcal{M}}_\mathrm{b}) \leq \mathrm{N} \tag{3.34}$$

where $\mathcal{E}(\boldsymbol{\mathcal{M}}_b)$ and $\mathcal{C}(\boldsymbol{\mathcal{M}}_b)$ are evaluated on the cut mesh $\mathcal{T}_h$. This is the same formulation as before except that the error functional $\mathcal{E}(\boldsymbol{\mathcal{M}}_b)$ is approximated by

$$\mathcal{E}(\boldsymbol{\mathcal{M}}_b) = \sum_{\kappa_b \in \mathcal{T}_{h,b}} \eta_{\kappa_b}(\mathcal{M}_{\kappa_b}) \tag{3.35}$$

where η_{κ_b} is the sum of the error contributions from all cut elements created from the background element κ_b. Furthermore, the local sampling is performed in the same way except that elements intersecting with the geometry require the generation of sub-cut-meshes to determine the error sample pairs $(\mathcal{S}_{\mathcal{C}_i}, f_{\mathcal{C}_i})|_{i=1}^{n_{\mathrm{config}}}$ for each background element configuration. FIGURE 3-3 shows an example of different configurations for cut elements. Note that for background elements on the boundary with most of its area dominated by a null region, a particular split configuration may not intersect with the computational domain, yielding a zero change in the computed error.



(a) Configuration $\mathcal{C}_1$ (b) Configuration $\mathcal{C}_2$ (c) Configuration $\mathcal{C}_3$

FIGURE 3-3: Example split configurations for cut elements (Sun [71])

Once the local error pairings are computed, the error model synthesis and optimization is performed in the same manner as before, though special treatment is required for vertices and elements outside the computational domain.

Null Vertex Metric

For null background elements, which are completely outside the computational domain Ω, the local error is set to zero, and the corresponding vertex metric is not included in the optimization. Nevertheless, the null vertices still require a defined metric, $\mathcal{M}_{v\in\mathcal{V}^{null}}$, in order to generate an adapted background mesh, $\mathcal{T}_{h,b}$. For improved efficiency, null elements should be as coarse as possible. However, since the metric of null vertices near boundaries can have an impact on the metrics of valid vertices within the computational domain, the null vertex metrics are calculated to be as large as possible without having a strong influence over the valid vertices' metric. This is done with the help of vertex layers that control the growth of metrics defined on null vertices based on the vertices' degree of separation from the computational domain [71]. An example of a vertex layer is shown in FIGURE 3-4. Using this method, each vertex is identified with a vertex layer number $\mathcal{L}(v)$, and only those in the zero layer $\mathcal{L}(v) = 0$ have metrics that are optimized. Note that vertices that are connected to faces with a portion inside the computational domain are also assigned to the zero vertex layer, $\mathcal{L}(v) = 0$, despite their location being outside the computational do-

main. For the remaining vertices, $\mathcal{L}(v) > 0$, the metric is determined from an affine averaging of neighboring metrics, as defined by Pennec *et al.* [59], compounded with a growth factor γ. In particular,

$$\mathcal{M}_v = \gamma^{L(v)-1}\overline{\mathcal{M}}_{v_{neighbors}} \tag{3.36}$$

where $\overline{\mathcal{M}}_{v_{neighbors}}$ is the affine average of all neighboring metrics at vertices with two degrees of separation or less: $\mathcal{L}(v_{neighbors}) \in \{\max(0, \mathcal{L}(v)-1), \max(0, \mathcal{L}(v)-2)\}$, and γ is set to 0.9. Note that the characteristic length scales of a metric scale inversely with the magnitude of the eigenvalues.

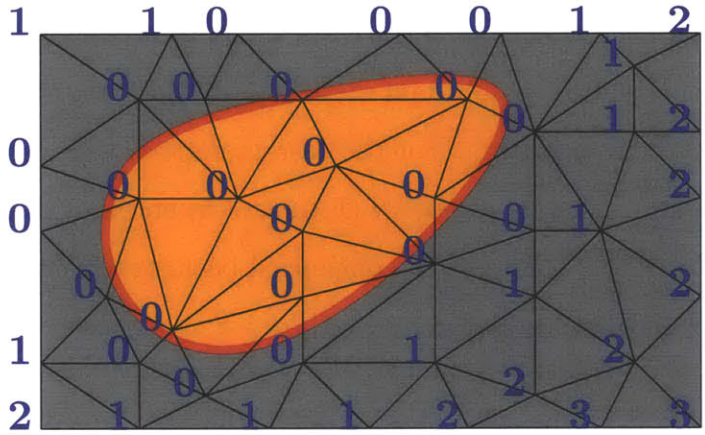


FIGURE 3-4: Example of vertex layer (grey represents null region)

The above method for integrating cut cells with mesh adaptation is leveraged for conjugate heat transfer models. Since the mesh adaptation process is performed on the background mesh $\mathcal{T}_{h,b}$ for cut-meshes, the algorithm is completely decoupled from the geometry. This allows for efficient application of the mesh adaptation method to general models consisting of multiple regions defined by complex geometry.

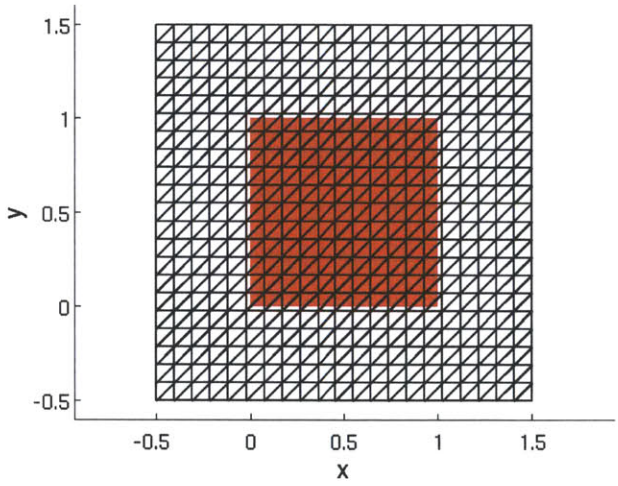
3.5.3 Cut-Cell r^{α}-Type Corner Singularity

Here, optimization of a cut mesh using the mesh adaptation procedure defined in SECTION 3.5.1 with the cut-cell extensions described in SECTION 3.5.2 is demonstrated.

To test for mesh optimality, a computed mesh distribution is compared to that of an analytically derived distribution. In particular, the optimal anisotropic element size distribution for a 2D function with a r^α-type corner singularity (as derived by Yano [77]) is used. For elliptic problems, this type of singularity appears at geometric corners, and is defined by

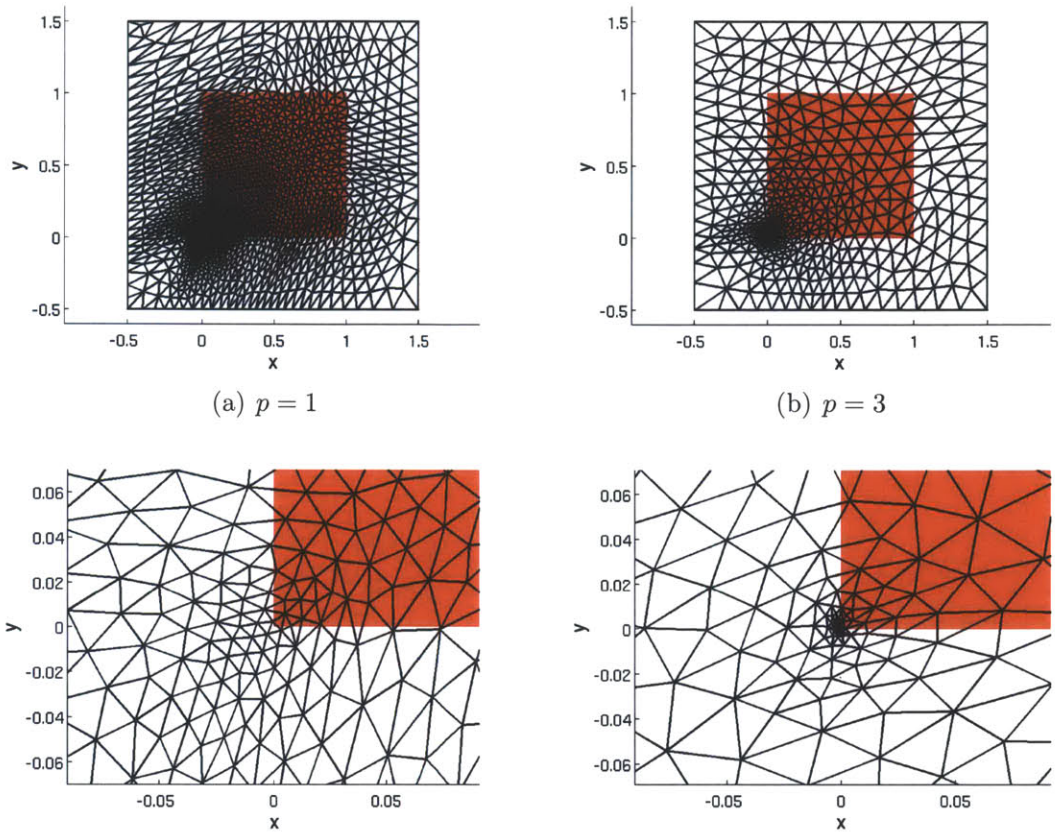
$$u(r, \theta) = r^\alpha \sin[\alpha(\theta + \theta_0)] \tag{3.37}$$

where $r = \sqrt{x_1^2 + x_2^2}$ is the radius from the origin, $\tan(\theta) = x_2/x_1$, $\alpha > 0$ is the singularity strength constant, and θ_0 is the offset angle. Yano [77] proved that the optimal mesh grading near the corner becomes stronger as α decreases or as the order of polynomial basis functions p increases. Yano also demonstrated that, for a boundary conforming mesh, the optimal mesh distribution is achieved when using a similar mesh adaptation algorithm. For the case presented here, a background mesh is intersected with a unit square geometry (FIGURE 3-5) and is optimized by adapting to the solution's L_2 error. The resulting optimized background meshes for $p = 1$ and $p = 3$ with 4000 DOF and singularity strength $\alpha = 2/3$ are shown in FIGURE 3-6.



(a) Initial Background Mesh

FIGURE 3-5: Background mesh for r^α singularity problem



(a) $p = 1$

(b) $p = 3$

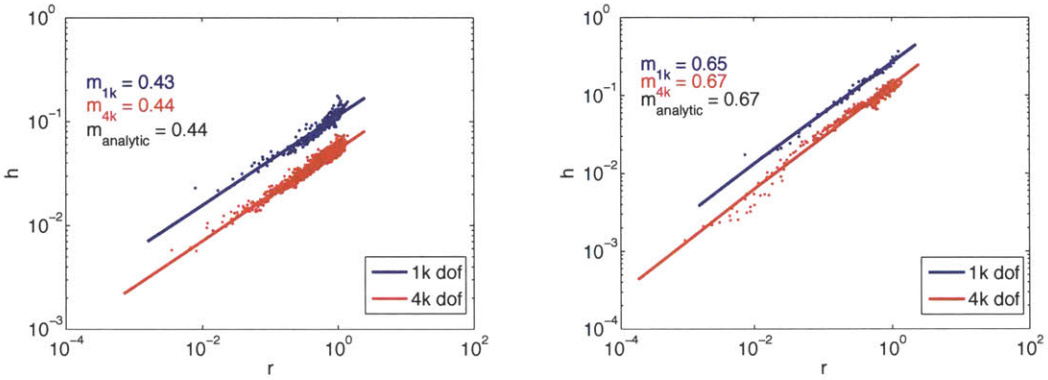
(c) $p = 1$ zoom on singularity

(d) $p = 3$ zoom on singularity

FIGURE 3-6: Optimized meshes for r^{α} singularity problem with 4000 DOF

The optimized meshes correctly show a stronger mesh grading toward the singularity located at the origin, especially for $p = 3$, and have nearly isotropic elements within the computational domain. For this problem, the optimal mesh is fully isotropic, though occasional anisotropic elements are observed near the geometry boundary, which propagate through the null region. The development of anisotropic elements is due to a poor construction of the error model on particular cut cells. If the cut element occupies only a small fraction of the background element, the error of certain split configurations (see SECTION 3.5.2) could result in a zero change in error, yielding insufficient information to effectively construct a useful error model. Since the metrics at null vertices are determined from their neighboring vertex metrics, the anisotropy propagates through the null region. Nevertheless, the elements within the computational domain still closely achieve the optimal distribution.

FIGURE 3-7 shows the distribution of the element size, h, within the computational domain versus the distance from the singularity, r, for $p = 1$ and 3 for 1000 and 4000 DOF. The element size h is determined by the elemental implied metric: $h = \det(\boldsymbol{\mathcal{M}}_\kappa)^{-1/4}$, and the distance r is measured from the singularity at the origin to the centroid of each element. The optimal values of h and r vary linearly in a log-log space with a slope of $k_{\text{analytic}} = 0.44$ and 0.67 for $p = 1$ and 3, respectively. From the distribution data, a linear regression is computed to show that the mesh adaptation procedure with cut elements achieves optimal distributions.



(a) $p = 1$ (b) $p = 3$

FIGURE 3-7: Metric distribution for r^α singularity problem with $\alpha = 2/3$

Chapter 4

Conjugate Navier-Stokes Heat Transfer

Conjugate Heat Transfer (CHT) refers to the process of heat exchange between a conductive solid and an adjacent convective fluid. Several engineering applications, such as turbine blade cooling, high-speed re-entry thermal control, and regenerative cooling, rely on accurate predictions of both fluid and solid temperature and heat fluxes. In this chapter, both the cut-cell method developed in CHAPTER 2 and output-based adaptation described in SECTION 3.5 are utilized to accurately model conjugate heat transfer between a solid and laminar flow. SECTION 4.1 details the application of the DG formulation to a NS-heat interface, as proposed by Sun [71], while the following sections demonstrate the framework's capability for CHT problems.

4.1 Interface Conditions for Navier-Stokes CHT

For the 2D conjugate heat transfer problem, interfaces define the boundary between a fluid and solid sub-domain. The fluid is governed by the Navier-Stokes equations with a conservative state vector $\mathbf{u}^{(1)} = [\rho, \rho\mathbf{u}, \rho E]^T$, while the solid is governed by the heat conduction equation with a state vector $\mathbf{u}^{(2)} = T$. See APPENDIX A.1 and APPENDIX A.2 for a detailed formulation of the heat and Navier-Stokes equations.

At the interface, three constraints are imposed:

$$\begin{aligned} u = v &= 0, && \text{no} - \text{slip condition} \\ T^{(1)} &= T^{(2)}, && \text{temperature continuity} \\ \kappa_T^{(1)} \nabla T^{(1)} \cdot \mathbf{n}^{(1)} &= -\kappa_T^{(2)} \nabla T^{(2)} \cdot \mathbf{n}^{(2)}, && \text{heat flux continuity} \end{aligned}$$

where κ_T is the thermal conductivity.

The above interface conditions are set by defining the numerical fluxes of the residual interface terms in the DG discretization appropriately.

4.1.1 Interface State and Discretization

The inviscid and viscous interface discretization definition relies on the construction of a coupled interface state, $\mathbf{u}_\Sigma$, which is used to compute each domain's numerical flux. As proposed by Sun [71], the choice of the constructed interface state is:

$$\mathbf{u}_\Sigma^{(1)} = \begin{bmatrix} \rho^{(1)} \\ \mathbf{0} \\ \rho^{(1)} \{T\} \frac{R}{\gamma - 1} \end{bmatrix}, \quad \text{and} \quad \mathbf{u}_\Sigma^{(2)} = \{T\} \tag{4.1}$$

where $\{T\} = 0.5(T^{(1)} + T^{(2)})$ is the average temperature across the interface. FIGURE 4-1 shows an illustration of the interface states used in the interface residual evaluation.

With this interface state definition, the inviscid interface discretization term for $i = 1, 2$ on each domain Ω_i, is set to:

$$\mathcal{R}_\Sigma^{inv}(\mathbf{w}, \mathbf{v}) = \sum_{f \in \Sigma} \int_f \mathbf{v}^{+^T} \mathcal{H}_\Sigma^{(i)}(\mathbf{w}^+, \mathbf{u}_\Sigma^{(i)}; \mathbf{n}^{(i)}) \tag{4.2}$$

where the fluid inviscid flux is the inviscid boundary flux calculated with the interface state: $\mathcal{H}_\Sigma^{(1)} = \mathcal{H}^b(\mathbf{u}_\Sigma^{(1)})$, and the solid inviscid flux is set to zero: $\mathcal{H}_\Sigma^{(2)} = 0$.

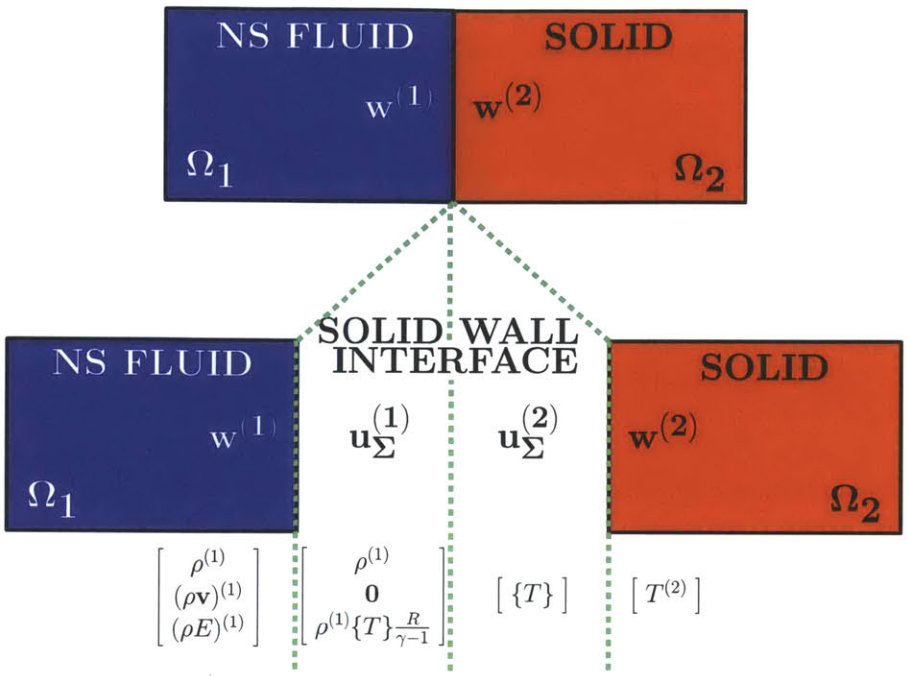


FIGURE 4-1: Sketch of NS solid wall interface states used to compute numerical fluxes

For ease of comparison between interface and non-interface residual terms, the viscous discretization presented in SECTION 3.2 is repeated here:

$$\begin{aligned}\mathcal{R}^v_{h,p}(\mathbf{w},\mathbf{v}) \equiv & \sum_{\kappa\in\mathcal{T}}\int_\kappa \nabla\mathbf{v}^T\cdot(\mathcal{A}(\mathbf{w})\nabla\mathbf{w}) \\ & -\sum_{f\in\Gamma_i}\int_f \left[[[\mathbf{w}]]^T\cdot\{\mathcal{A}^T(\mathbf{w})\nabla\mathbf{v}\}+[[\mathbf{v}]]^T\cdot(\{\mathcal{A}(\mathbf{w})(\nabla\mathbf{w}-\eta_f\mathbf{r}_f([[\mathbf{w}]]))\})\right] \\ & -\sum_{f\in\Gamma_b}\int_f \left[(\mathbf{w}^+-\mathbf{u}^b)^T\mathcal{A}_b^T\nabla\mathbf{v}^+)\cdot\mathbf{n}^+ + \mathbf{v}^{+^T}\left(\mathcal{A}_b\left(\nabla\mathbf{u}^b-\eta_f\mathbf{r}_f^b(\mathbf{w}^+-\mathbf{u}^b)\right)\right)\cdot\mathbf{n}^+\right] \\ & -\mathcal{R}^{visc}_\Sigma(\mathbf{w},\mathbf{v})\end{aligned} \tag{4.3}$$

The viscous interface residual term for $i=1,2$ on each domain Ω_i, is chosen as:

$$\begin{aligned}\mathcal{R}^{visc}_\Sigma(\mathbf{w},\mathbf{v}) = & \sum_{f\in\Sigma^{(1)}}\int_f\left((\mathbf{w}^{(1)}-\mathbf{u}^{(1)}_\Sigma)^T(\mathcal{A}^{(1)^T}_\Sigma\nabla\mathbf{v}^{(1)})+\mathbf{v}^{(1)^T}\widehat{\mathcal{A}\sigma}^{(1)}\right)\cdot\mathbf{n}^{(1)} \\ & +\sum_{f\in\Sigma^{(2)}}\int_f\left((\mathbf{w}^{(2)}-\mathbf{u}^{(2)}_\Sigma)^T(\kappa^{(2)}_T\nabla\mathbf{v}^{(2)})+\mathbf{v}^{(2)^T}\widehat{\mathcal{A}\sigma}^{(2)}\right)\cdot\mathbf{n}^{(2)}\end{aligned} \tag{4.4}$$

where the coupled viscous fluxes, $\widehat{\mathcal{A}\sigma}^{(\mathrm{i})} \cdot \mathbf{n}^{(\mathrm{i})}$, are defined by:

$$\widehat{\mathcal{A}\sigma}^{(1)} \cdot \mathbf{n}^{(1)} = \begin{cases} (\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{mass}} \\ (\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{momentum}} \\ 0.5((\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{energy}} + \kappa_T^{(2)} \sigma_h^{(2)} \cdot \mathbf{n}^{(1)}) \end{cases} \tag{4.5}$$

$$\widehat{\mathcal{A}\sigma}^{(2)} \cdot \mathbf{n}^{(2)} = 0.5((\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(2)})_{\text{energy}} + \kappa_T^{(2)} \sigma_h^{(2)} \cdot \mathbf{n}^{(2)}). \tag{4.6}$$

The viscosity matrix is computed using the interface state: $\mathcal{A}_\Sigma^{(1)} = \mathcal{A}^{(1)}(\mathbf{u}_\Sigma^{(1)})$, and the lifted gradient, $\sigma_h^{(\mathrm{i})}$, is given by:

$$\sigma_h^{(\mathrm{i})} = \nabla \mathbf{w}^{(\mathrm{i})} - \eta_f \mathbf{r}_{f^{(\mathrm{i})}}^{\Sigma}(\mathbf{w}^{(\mathrm{i})} - \mathbf{u}_\Sigma^{(\mathrm{i})}) \tag{4.7}$$

where η_f is a stabilizing coefficient, and $r_{f^{(\mathrm{i})}}^{\Sigma}$ is the local lifting operator defined by:

$$\int_{\Omega_h^{f^{(\mathrm{i})}}} \tau^T \cdot \mathbf{r}_{f^{(\mathrm{i})}}^{\Sigma}(\phi) = \int_f \phi^T \tau^+ \cdot \mathbf{n}^{(\mathrm{i})} \quad \forall \tau \in [\mathcal{V}_{h,p}^{(\mathrm{i})}]^d \tag{4.8}$$

Note that $(\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{energy}}$ is the same as the heat flux, $\kappa_T^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)}$, since the interface state, $\mathbf{u}_\Sigma^{(1)}$, satisfies the no-slip condition. Finally, additional analysis on the current interface boundary condition is recommended to determine adjoint consistency and other discretization properties.

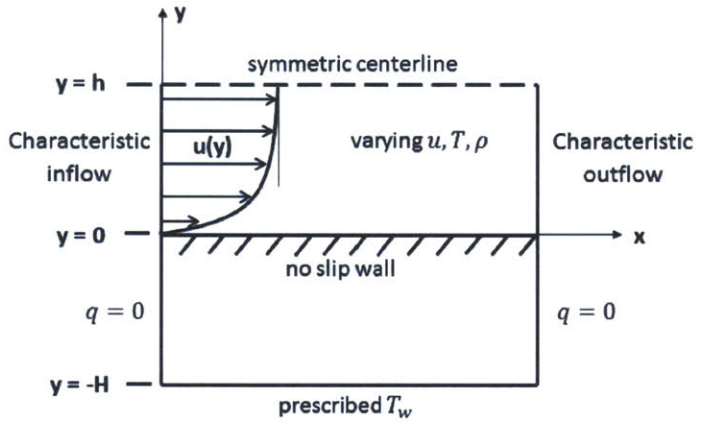
4.2 Compressible Poiseuille Flow over a Cooled Slab

In this section, the optimal order of convergence of the DG scheme applied to a compressible laminar flow over a cooled slab using cut-cell methods is demonstrated. First, a uniform refinement study is performed to demonstrate the optimal L^2-error convergence rates for the CHT framework. Then, adaptation tests are performed

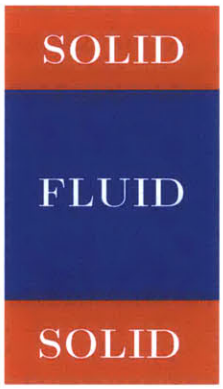
to show super-convergent rates in both drag and heat flux error. To compute the L^2-error of the conservative states, an analytical manufactured solution for a varying viscosity compressible flow is developed.

4.2.1 Conjugate Model

FIGURE 4-2(a) shows a sketch of half of the computational domain used to model the conjugate heat transfer channel flow while FIGURE 4-2(b) shows the separate regions of the full model.



(a) Sketch of half of the computational domain



(b) Region definition

FIGURE 4-2: Compressible Poiseuille flow model

The fluid domain is bounded by an inflow, outflow, and solid wall interfaces. The solid domain of thickness H is bounded by left and right adiabatic walls, a Dirichlet temperature condition on the outer wall, and a fluid interface on the inner wall. In this work, the channel height was chosen to be $(2h) = 2$ and the slab thickness, $H = 0.8$. Solutions are obtained for an inlet Reynolds number of $Re_h = 10{,}000$, set by the non-dimensionalized viscosity with reference to the inlet conditions: $\overline{\mu} = \mu/(\rho_{ref} V_{ref} h) = 1/Re_{ref} = 10^{-4}$. Additionally, a Prandtl number of $Pr = 1$, a specific heat ratio of $\gamma = 1.4$, and a conductivity ratio of $\kappa_s/\kappa_f = 10$ is set. The inflow and outflow conditions are weakly enforced by calculating a boundary flux based on a boundary state constructed from the analytical solution and interior information. With a fully

developed flow assumption, the analytical solution comprises of a constant pressure field with source terms used to drive the compressible flow.

4.2.2 Conjugate Manufactured Solution

An analytical solution is derived to verify optimal convergence rates for a variable viscosity fluid region's solution coupled with a solid region's solution (see APPENDIX B for a detailed derivation). In the fluid region Ω_f, this manufactured solution assumes a variable viscosity and thermal conductivity that follows Maxwell's molecular theory. In the solid region Ω_s, a sinusoidal source term is used to manufacture a solution that is outside the polynomial space. Together, these analytical solutions are given by:

Fluid

$$u_f^* = -\frac{T_i \epsilon}{2 T_a \delta} \left(s^{*2} - 2 s^* \right) \tag{4.9}$$

$$v_f = 0 \tag{4.10}$$

$$T_f^* = \frac{T_i}{T_a} - \frac{\beta}{12} \left(s^{*4} - 4 s^{*3} + 6 s^{*2} - 4 s^* \right) \tag{4.11}$$

Solid

$$T_s = \tau \mathsf{sin}(y + H) + \left(\frac{\kappa_f T_a^2 \beta}{\kappa_s 3 h T_i} - \tau \mathsf{cos}(H) \right) y + T_i - \tau \mathsf{sin}(H) \tag{4.12}$$

where s^* is a non-dimensionalized coordinate scaled by the non-dimensional temperature ($T^* ds^* = dy^*$), T_a is a reference temperature, ϵ, β, and δ are non-dimensional numbers, τ is a source coefficient, and T_i is the constant non-dimensional interface temperature set to $T_i = 20.0$. The density and energy profile is then determined from the state equation and internal energy definition:

$$\rho = \frac{P}{RT} \qquad E = c_v T + \frac{1}{2} \mathbf{v}^2 \tag{4.13}$$

In order to achieve the correct interface temperature, the outer wall Dirichlet condition T_w is:

$$T_w = \left(\frac{\kappa_f T_a^2 \beta}{\kappa_s 3 h T_i} - \tau \cos(H) \right) (-H) + T_i - \tau \sin(H) \tag{4.14}$$

Furthermore, the fluid and solid domain source terms required in manufacturing this analytical solution are:

$$\mathcal{S}_f = \begin{bmatrix} 0 \\ \rho f \\ 0 \\ \rho u f \end{bmatrix} \qquad \mathcal{S}_s = \begin{bmatrix} \tau \kappa_s \sin(y + H) \end{bmatrix}$$

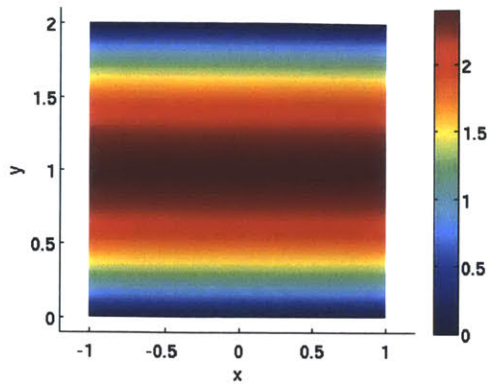
where f is a force coefficient set to six times the reference viscosity: $f = 6\mu_0$.

The addition of these source terms along with appropriate boundary conditions (defined by the analytical solution) is sufficient to solve the problem. Given the analytical solution, the total drag and heat flux across the interface is determined exactly by:

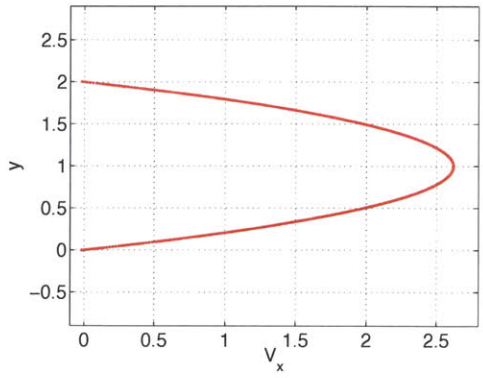
$$D = (2L)\mu_0 \left. \frac{\partial u}{\partial y} \right|_{y=0} = 2L\mu_0 \frac{\epsilon}{\delta} = 2.234668913653411 \times 10^{-3} \tag{4.15}$$

$$Q = (2L)\kappa_f \frac{\partial T}{\partial y}_f = 2L\kappa_f \frac{T_a^2 \beta}{3hT_i} = 3.905778470148311 \times 10^{-3} \tag{4.16}$$

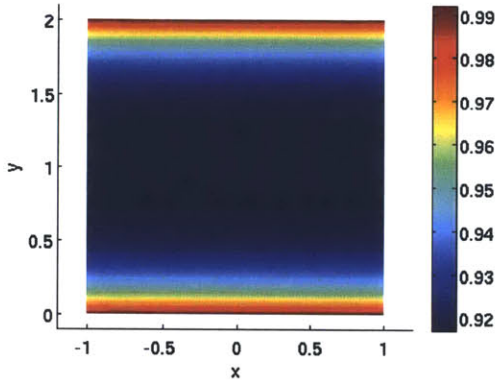
where $L = 2$ is the interface length. FIGURE 4-3 shows the resulting fluid's streamwise velocity, density, and Mach from solving on a 40x80 uniform triangular mesh with $p = 3$. Vertical slices of the solution remain constant in the streamwise direction, and are also illustrated as a function of the vertical coordinate y. Furthermore, FIGURE 4-4 shows the same plots for temperature throughout the entire computational domain, as well as the viscosity assuming Maxwell's molecular theory.



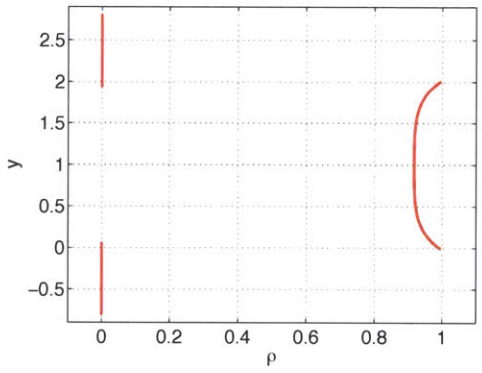
(a) Fluid streamwise velocity



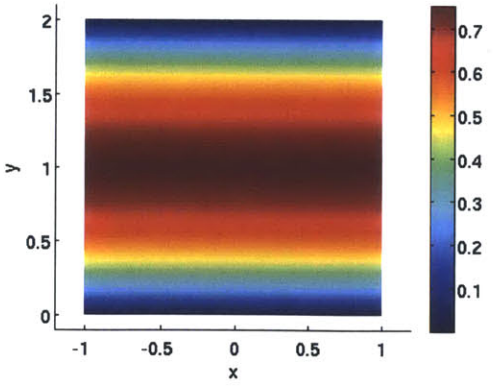
(b) Streamwise velocity vertical slice



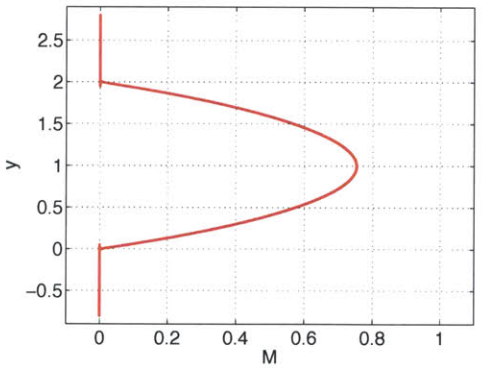
(c) Fluid density



(d) Density vertical slice

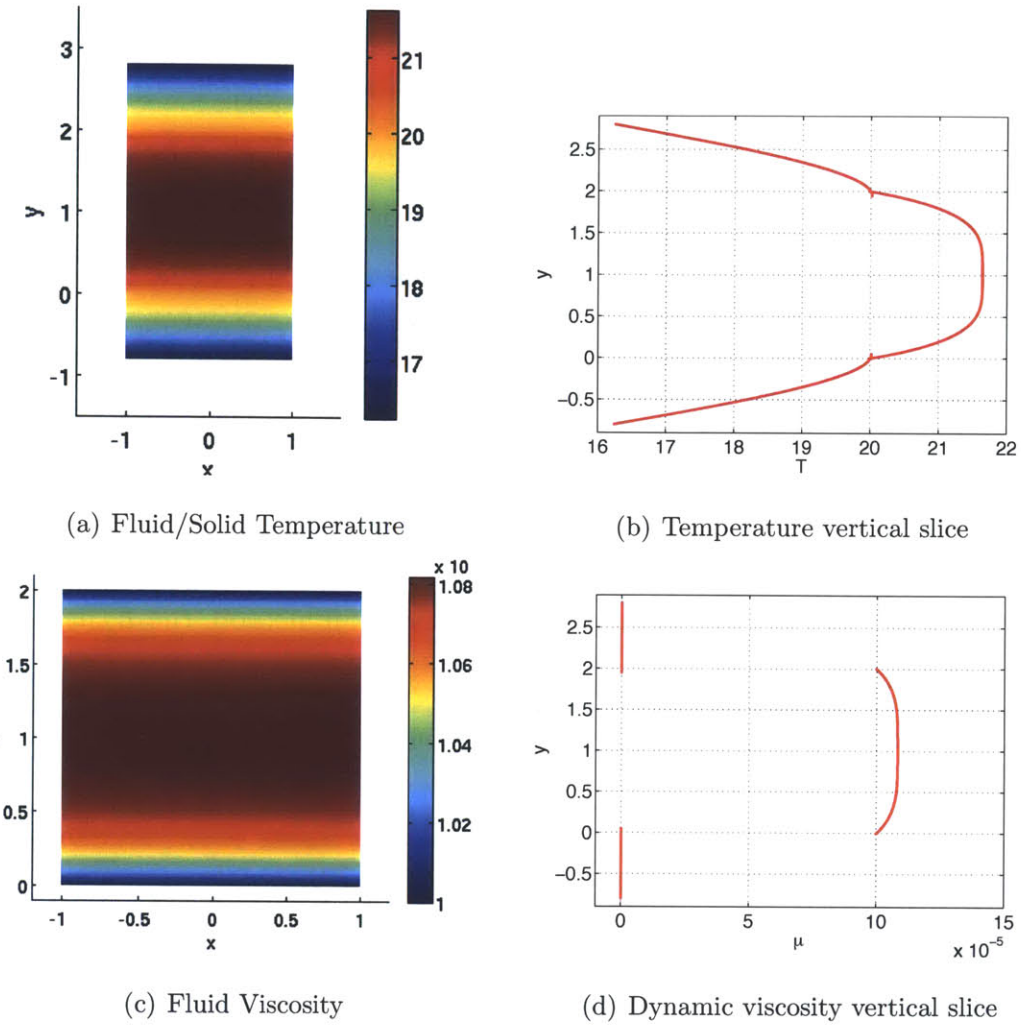


(e) Fluid Mach



(f) Mach vertical slice

FIGURE 4-3: Numerical solution to the compressible Poiseuille flow with variable μ and κ



(a) Fluid/Solid Temperature

(b) Temperature vertical slice

(c) Fluid Viscosity

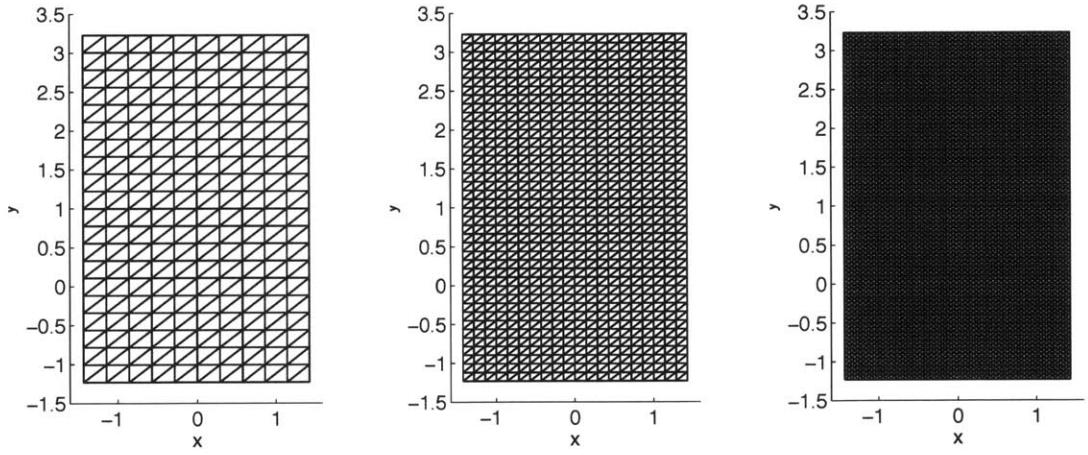
(d) Dynamic viscosity vertical slice

FIGURE 4-4: Temperature and corresponding viscosity variation for compressible Poiseuille flow

As expected, all states preserve their 1-D profiles. The streamwise velocity and temperature plots in the fluid regime reveal a similar variation in y as the incompressible Poiseuille case, though are inherently different under the same conditions. The sinusoidal source variation of temperature in the solid is also observed, as well as the discontinuity in the temperature derivative at the interface in order to enforce a constant heat flux between two mediums of different thermal conductivities. Lastly, the viscosity variation due to the temperature dependence, as well as a maximum value corresponding to the maximum temperature at the channel centerline is verified.

4.2.3 Uniform Refinement Convergence Study

In order to verify proper convergence of the DG scheme with a CHT problem, a convergence study on the conservative states' L_2 error is performed. Uniformly refined triangular meshes (FIGURE 4-5) are used as background meshes to determine the L_2 error rate of convergence. For each refinement, the Poiseuille flow geometry is intersected with the background mesh to create a new cut-cell mesh. Solutions are then obtained by using a direct solver (UMFPACK[25]). The L_2 error for each conservative variable is calculated by integrating the numerical error (based on the analytical solution) using a quadrature rule over. For the fluid conservative states, the integration is computed over the fluid domain, while the temperature L_2 is calculated only in the solid domain. For the discontinuous Galerkin method, the L_2 error of the conservative variables is expected to converge at a rate of $p + 1$ where p is the order of the polynomial bases used to represent the solution.



(a) 10 × 20 (b) 20 × 40 (c) 40 × 80

FIGURE 4-5: Uniformly refined meshes

FIGURE 4-6 and TABLE 4.1 show the respective convergence rates for the uniform refinement study.

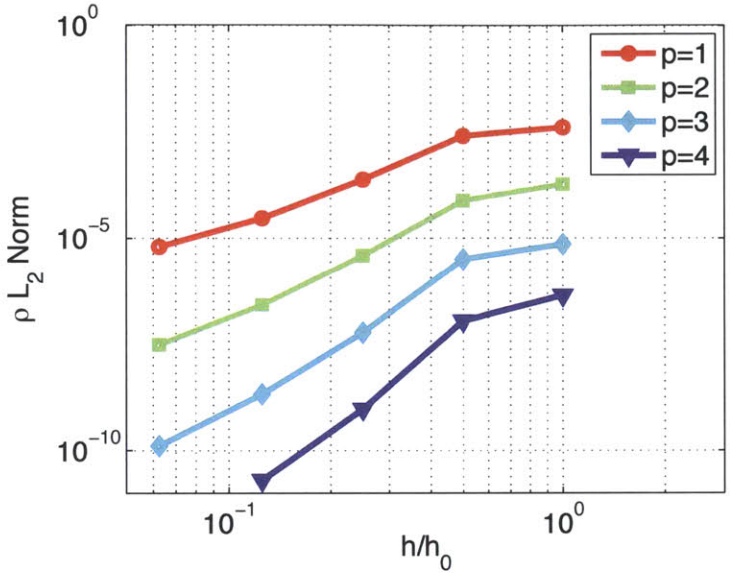


FIGURE 4-6: Convergence of the density L_2 error.

	ρ	ρu	ρv	ρe	T_s
$p=1$	0.65	1.26	1.04	1.31	0.70
	3.43	2.64	2.48	2.46	3.02
	3.02	2.15	2.18	2.07	3.02
	2.24	1.97	2.30	2.00	2.55
$p=2$	1.27	1.69	1.44	1.60	0.86
	4.29	4.00	3.72	4.00	5.89
	3.87	4.05	3.72	3.47	4.61
	3.11	3.08	3.41	3.04	3.16
$p=3$	1.19	1.64	1.41	1.78	0.66
	5.74	5.67	5.34	5.68	5.86
	4.80	4.68	4.67	4.16	7.63
	4.03	3.99	4.52	3.99	4.12
$p=4$	2.02	2.67	2.47	3.55	0.69
	6.86	6.90	5.89	5.88	8.84
	5.55	5.55	5.64	5.09	7.79
	mprec				

TABLE 4.1: Convergence rates of L_2 error.

The convergence rates of the fluid conservative state's L_2 error asymptotically approach the $p+1$ optimum as the mesh is refined. The solid's temperature L_2 error rates for $p = 1, 2, 3$ also approach $p+1$ with an applied sinusoidal source function. For $p = 4$, the solid's temperature rate appears to have a similar trend as in the other p-order cases, and would approach the $p+1$ optimum, though the error reaches machine precision (mprec) at the finest mesh level beforehand.

4.2.4 Adapted Solutions and Output Super-convergence

In this section, mesh adaptation is performed on the compressible Poiseuille conjugate problem to demonstrate super-convergent rates of interface output errors. For the case at hand, both the drag and heat flux outputs on the interface are calculated and

used to solve the dual problem in order to determine a dual weighted residual for error estimation and mesh adaptation. Lu [50] has shown that the implementation of dual consistent boundary conditions is important for achieving super-convergent functional output results. Likewise, interface conditions for CHT problems using a DG scheme need to be dual consistent in order to ensure super-convergent rates.

Consider a general DG discretization of the primal problem:

$$R_{h,p}(u_{h,p}, v_{h,p}) = 0, \quad \forall v_h \in \mathcal{V}_{h,p} \tag{4.17}$$

where $R_{h,p}$ is a semi-linear form derived from the weak formulation. Interest is given to interface outputs defined by:

$$J = \mathcal{J}(u) \equiv -\int_{\Sigma} g_{\Sigma} \mathcal{A}^{(-)} \nabla u^{(-)} \cdot \mathbf{n}^{(-)} ds \tag{4.18}$$

where g_{Σ} is a function defined on an interface Σ. In order for the discrete output, $\mathcal{J}_{h,p}(u_{h,p})$, to yield dual consistency for a nonlinear system, the exact dual solution, ψ, must satisfy:

$$R'_{h,p}[u](v_{h,p}, \psi) = \mathcal{J}'_{h,p}[u](v_{h,p}) \quad \forall v_{h,p} \in \mathcal{W}_{h,p} \tag{4.19}$$

where $\mathcal{W}_{h,p} \equiv \mathcal{V}_{h,p} + \mathcal{V}$ represents all functions existing in both the discrete solution and adjoint functional space, and $(\cdot)'[u]$ denotes the Fréchet derivatives with respect to u. Hartmann [36] and Oliver [56] provide detailed derivations of dual consistent DG discretizations for nonlinear systems.

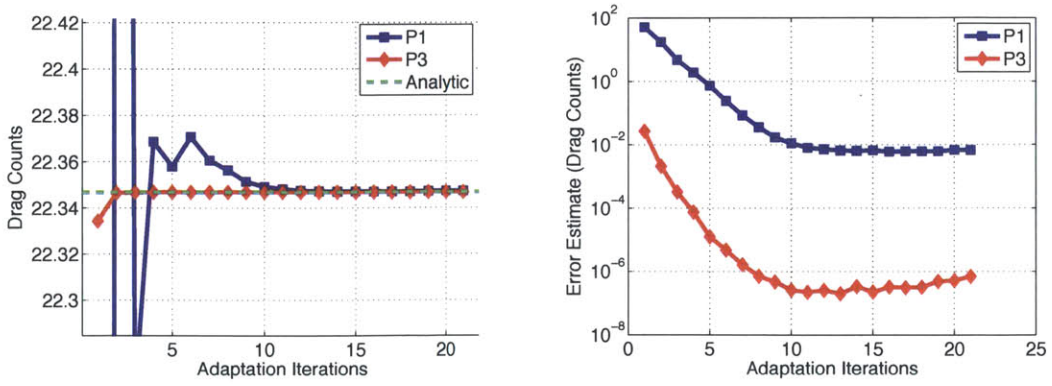
When super-convergence of the output error is achieved using the BR2 scheme, the rate of convergence approaches h^{2p} where h is a characteristic length scale of the mesh:

$$|\mathcal{J}(u) - \mathcal{J}_{h,p}(u_{h,p})| \leq Ch^{2p} \tag{4.20}$$

Since h scales with the number of degrees of freedom ($h \sim 1/\sqrt{\text{DOF}}$), the super-convergent rate of h^{2p} can also be expressed as $\text{DOF}^{-\text{p}}$. A formal proof of dual consistency of the functional output $\mathcal{J}_{h,p}(u_{h,p})$ for *scalar* interface problems is given by Sun [71], though additional analysis to prove dual consistency for the current nonlinear interface discretization is left for future research.

Drag Adapted Compressible Poiseuille Flow

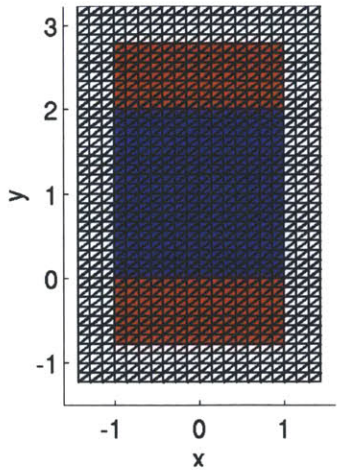
Here, the same analytical compressible Poiseuille flow solution shown in SECTION 4.2.2 is selected to perform a super-convergence study by adapting to drag on both of the no-slip wall interfaces. The error in the output is computed over a range of p-order's and degrees of freedom (DOF), and plotted to show the convergence. For each case, 20 adaptation iterations are computed to ensure adaptation convergence as illustrated in FIGURE 4-7, and the corresponding $p = 1$ and $p = 3$ meshes are shown in FIGURE 4-8. The computed output from the last 8 adaptation iterations is then averaged to represent the optimal discrete solution at a given p-order and DOF. A true output error is computed by taking the difference between the computed average and the exact analytical output solution given in EQUATION 4.15. The resulting errors and convergence rates are shown in FIGURE 4-9.



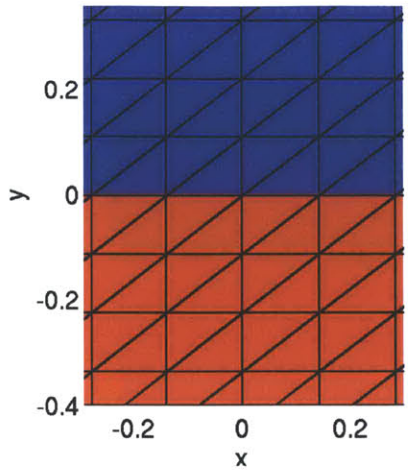
(a) Drag vs. adaptation iteration

(b) Drag error estimate convergence

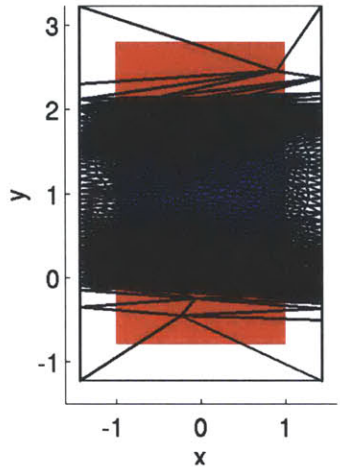
FIGURE 4-7: Poiseuille flow drag adaptation history for 16k DOF



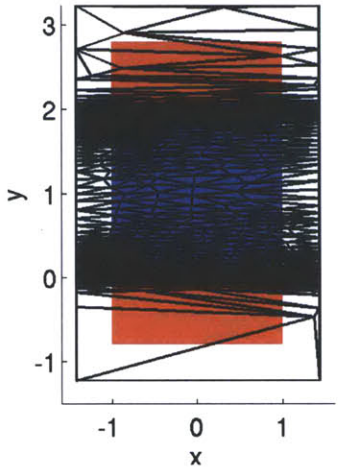
(a) Initial background mesh



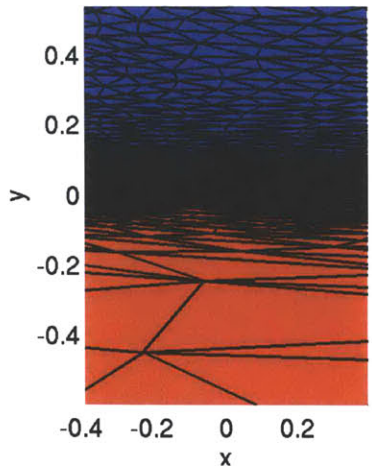
(b) Initial background mesh zoom



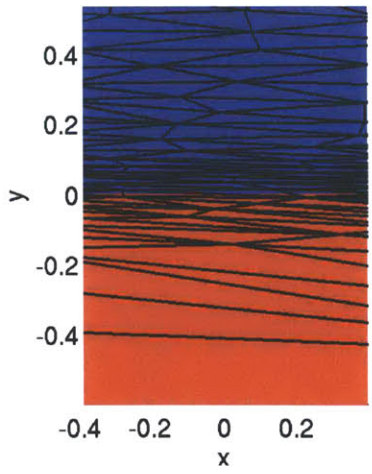
(c) Drag adapted p=1 16k DOF



(d) Drag adapted p=3 16k DOF

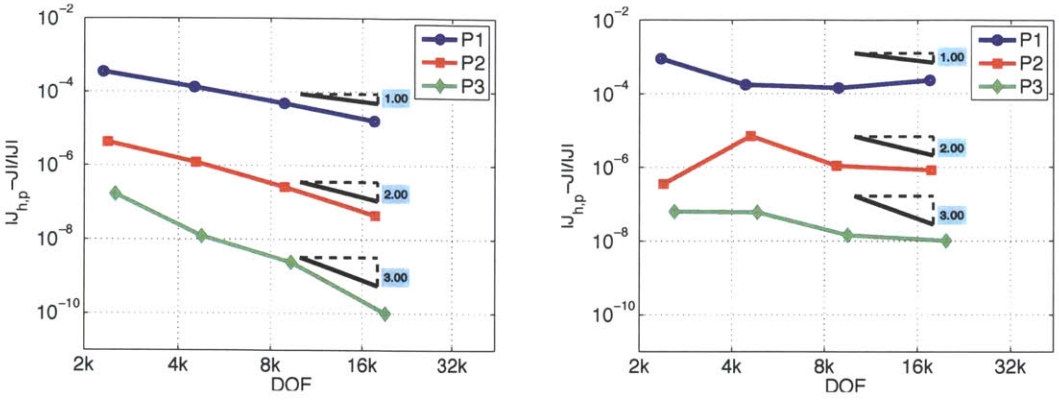


(e) Drag adapted p=1 16k DOF zoom



(f) Drag adapted p=3 16k DOF zoom

FIGURE 4-8: Poiseuille flow drag adapted meshes



(a) Drag convergence

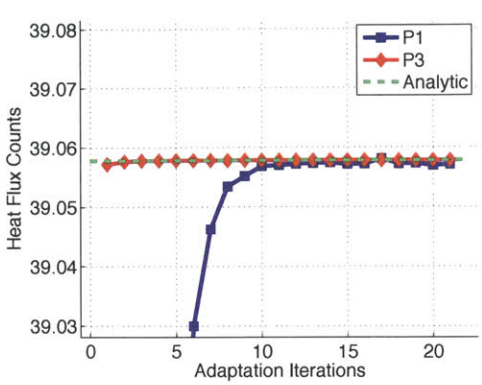
(b) Heat flux convergence

FIGURE 4-9: Poiseuille flow drag adapted error convergence

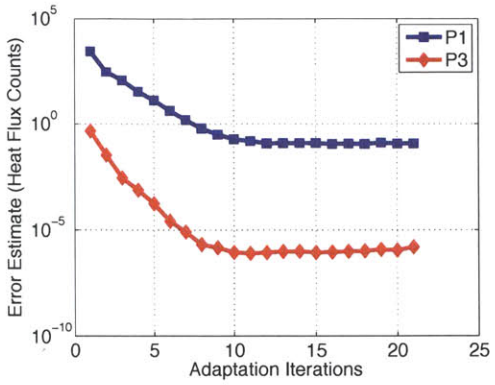
As shown in the drag adaptation error history, both $p = 1$ and $p = 3$ 16k DOF solutions converge in ~ 12 adaptation iterations from a coarse, uniform grid. In the final adapted meshes, highly anisotropic refinement is observed in the fluid near the interface, which grows rapidly into very coarse elements in the solid and null regions. This is because the calculated drag on the interface is insensitive to perturbations in the solid region's solution compared to perturbations in the fluid region's solution. As such, super-convergent rates of $\text{DOF}^{-\text{p}}$ (h^{2p}) for $p = 1, 2$, and 3 are achieved for the drag error, though the rates are sub-optimal for the calculated error in heat flux across the interface. This is due to the lack of refinement in the solid region, which is required to properly resolve the temperature distribution for heat flux calculations.

Heat Flux Adapted Compressible Poiseuille Flow

Performing the same procedure as above, but adapting to heat flux instead yields improved error convergence results for the heat flux output. Again, a true output error is computed by taking the difference between the computed average from the last 8 adaptation iterations and the exact analytical output solution given in EQUATION 4.15. FIGURE 4-10 shows the convergence history for a $p = 1$ and $p = 3$ solution relative to the exact solution, while FIGURE 4-11 shows the corresponding meshes. Both the drag and heat flux are again computed over meshes of increasing degrees of freedom, and the resulting output error convergence is shown in FIGURE 4-12.

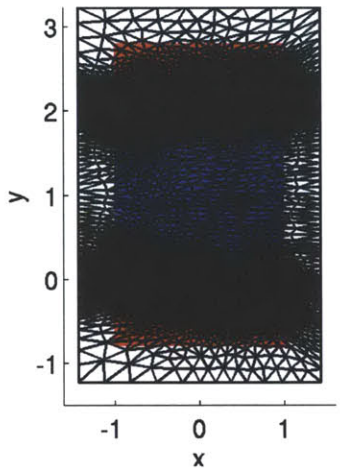


(a) Heat flux vs. adaptation iteration

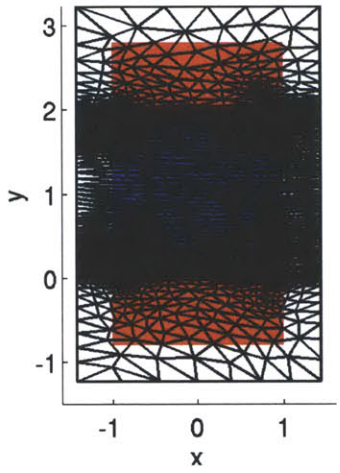


(b) Heat flux error estimate convergence

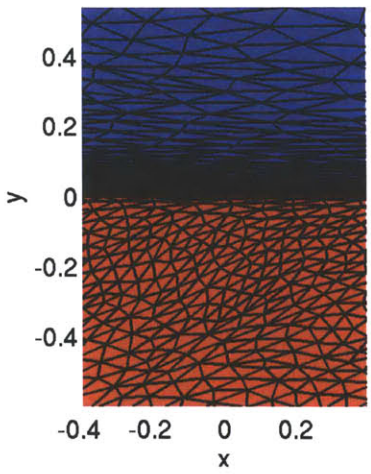
FIGURE 4-10: Poiseuille flow heat flux adaptation history for 16k DOF



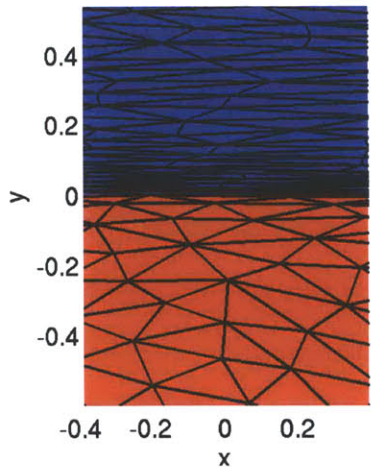
(a) Heat flux adapted p=1 16k DOF



(b) Heat flux adapted p=3 16k DOF

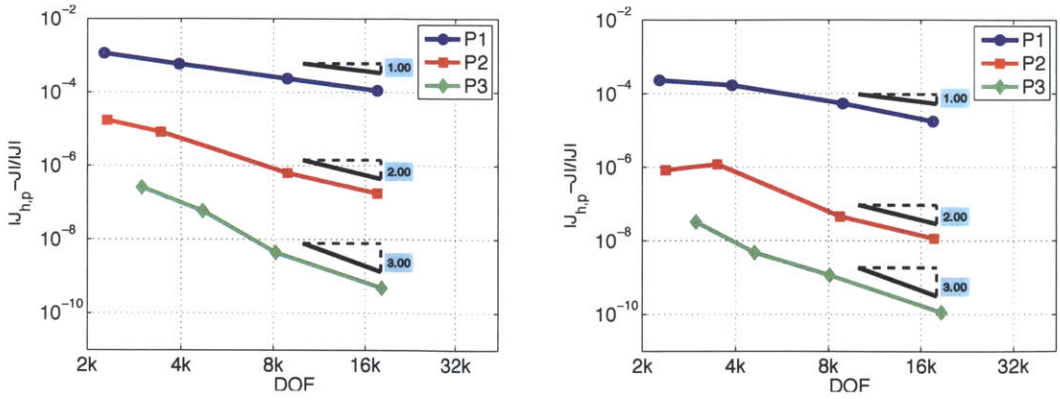


(c) Heat flux adapted p=1 16k DOF zoom



(d) Heat flux adapted p=3 16k DOF zoom

FIGURE 4-11: Poiseuille flow heat flux adapted meshes



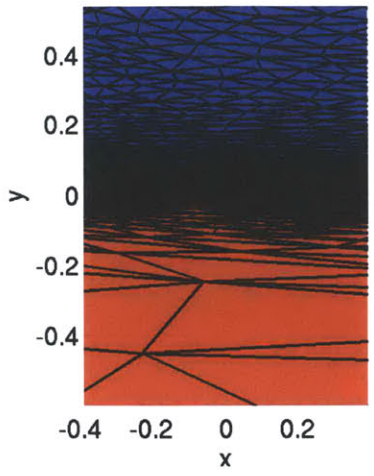
(a) Drag convergence

(b) Heat flux convergence

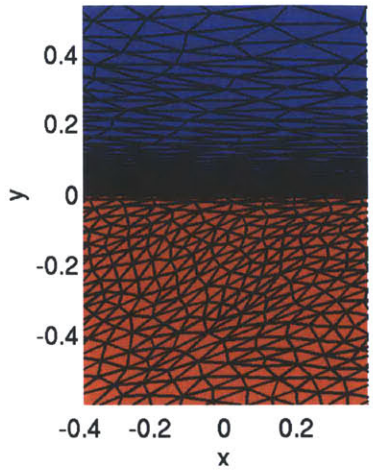
FIGURE 4-12: Poiseuille flow heat flux adapted error convergence

As illustrated in the heat flux adaptation error history, both $p = 1$ and $p = 3$ 16k DOF solutions converge in $\sim$ 12 adaptation iterations from a coarse, uniform grid. The $p = 3$ solution achieves over 4 orders of magnitude improvement in the heat flux error compared to that of the $p = 1$ solution. In the final adapted meshes, highly anisotropic refinement is still observed in the fluid near the interface, though many degrees of freedom are diffused throughout the rest of the fluid region and the solid regions. This grid distribution occurs due to the heat flux output's larger region of influence. As such, super-convergent rates of $\text{DOF}^{-\text{p}}$ (h^{2p}) for $p = 1, 2$, and 3 are achieved for the heat flux error, and several orders of magnitude improvement in the computed heat flux accuracy is realized.

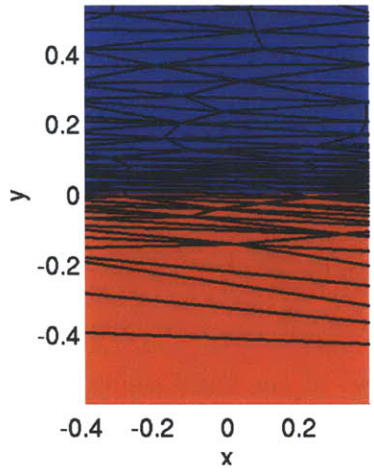
Super-convergent rates are also achieved for the drag error, though the error levels are higher than those found when adapting to drag. The preservation of drag's super-convergent rates is due to sufficient resolution of the boundary layer in the heat flux adapted meshes. Conversely, the increased error in drag is caused by the relocation of degrees of freedom away from the boundary layer into the rest of the fluid and solid regions. This shows that adapting to heat flux can yield accurate predictions in both drag and heat flux outputs, while adapting to drag may impede the accuracy in heat flux predictions. For comparison, a side by side view of the drag and heat flux adapted meshes for $p = 1$ and $p = 3$ is shown in FIGURE 4-13.



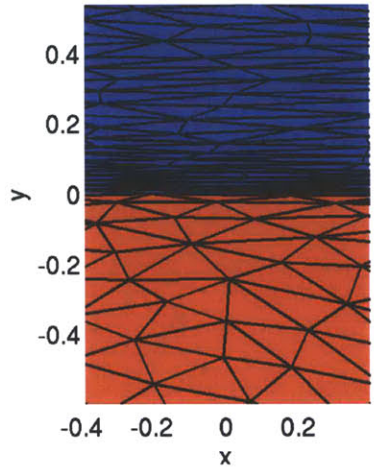
(a) Drag adapted p=1 16k DOF zoom



(b) Heat flux adapted p=1 16k DOF zoom



(c) Drag adapted p=3 16k DOF zoom



(d) Heat flux adapted p=3 16k DOF zoom

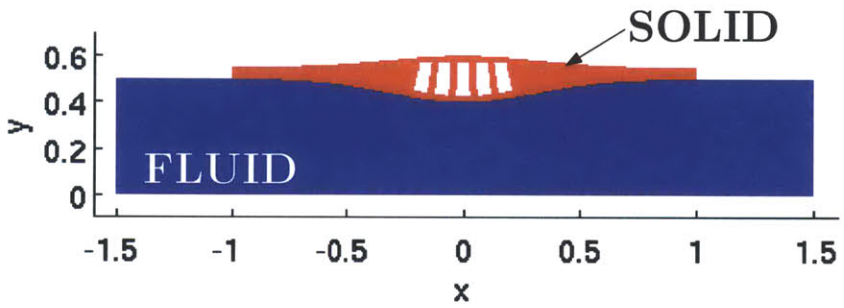
FIGURE 4-13: Poiseuille flow comparison of drag vs. heat flux adaptation

4.3 Navier-Stokes Cooled Nozzle

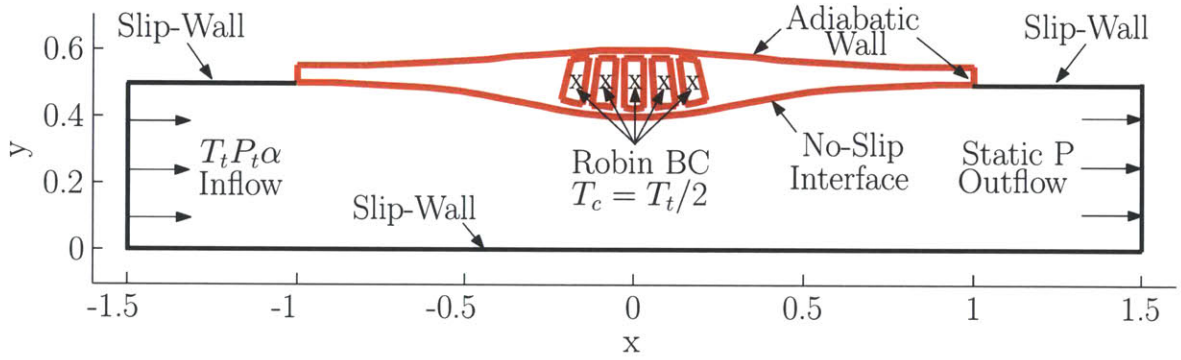
The coupled interaction between a heated fluid and a cooled solid is found in many aerospace applications, including heat exchangers, secondary flows in turbomachinery, and manifolds in rocket engines. Though these applications typically consist of highly turbulent and/or reacting flows, the case presented here demonstrates a benchmark capability of the CHT framework in computing solutions for laminar dominated flows.

4.3.1 Conjugate Model

FIGURE 4-14(a) and FIGURE 4-14(b) show the separate regions and sketch of the computational domain used to model the conjugate heat transfer laminar nozzle flow.



(a) Region definition



(b) Sketch of computational domain

FIGURE 4-14: Cooled nozzle flow model

For the case presented here, the inflow condition is set by a uniform flow with a non-dimensionalized total pressure and temperature calculated from a reference Mach $M_{ref} = 0.4$. In particular, the total pressure and temperature are non-dimensionalized by inlet reference conditions, and are calculated by:

$$\overline{P_t} = \frac{P_t}{\rho_{ref} V_{ref}^2} = \frac{P(1+\frac{\gamma-1}{2}M_{ref}^2)^{\frac{\gamma}{\gamma-1}}}{\rho_{ref} V_{ref}^2} = \frac{(1+\frac{\gamma-1}{2}M_{ref}^2)^{\frac{\gamma}{\gamma-1}}}{\gamma M_{ref}^2} \tag{4.21}$$
$$\overline{T_t} = \frac{T_t}{V_{ref}^2/c_v} = \frac{T(1+\frac{\gamma-1}{2}M_{ref}^2)}{V_{ref}^2/c_v} = \frac{(1+\frac{\gamma-1}{2}M_{ref}^2)}{\gamma(\gamma-1)M_{ref}^2}$$

Solutions were obtained with an inlet Reynolds number of $Re_H = 10{,}000$, set by the non-dimensionalized viscosity with reference to the inlet conditions: $\overline{\mu} = \mu/(\rho_{ref} V_{ref} H) = 1/Re_{ref} = 10^{-4}$ where $H = 1$ represents the full channel height. The thermal conductivity ratio κ_r is set to $\kappa_r = \kappa_s/\kappa_f = 700$ with a Prandtl number $Pr = 1$, and the fluid viscosity varies with temperature according to Sutherland's law. The solid domain of length $L = 2$ is bounded by adiabatic walls, a fluid interface on the bottom wall, and rectangular "cooling channels" that represent a cooling manifold. The cooling channel boundary condition is imposed through a Robin boundary condition:

$$-\kappa_s \nabla T = h(T - T_c) \tag{4.22}$$

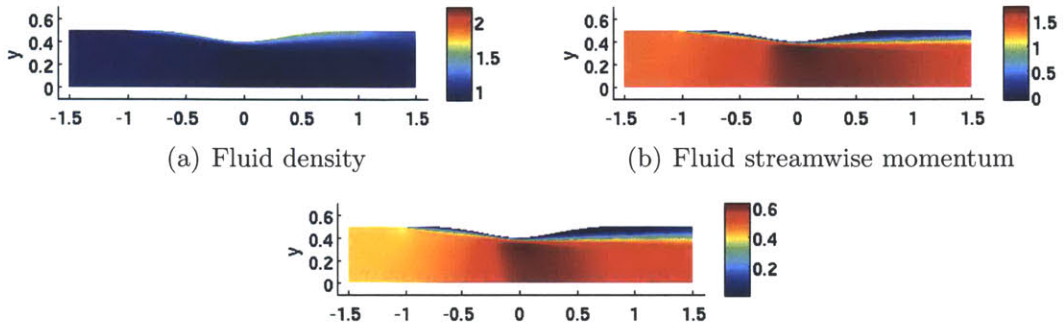
where T_c is the cooling temperature, and h is the convective heat transfer coefficient set to $h = 5\kappa_s$. For this problem, the primary outputs of interest are drag and heat flux along the interface wall.

4.3.2 Adapted Solutions and Output Super-convergence

In this section, mesh adaptation is performed on the cooled nozzle conjugate problem to demonstrate the super-convergence of interface output errors on a curved geometry using cut-cell methods. For this case, the primary outputs of interest are total drag and heat flux along the interface wall, and mesh adaptation is performed with respect to each output.

Numerical Solution to Cooled Nozzle Flow

FIGURE 4-15 shows the resulting fluid's density, streamwise momentum, and Mach from solving on a well refined heat flux adapted 50k DOF $p = 3$ mesh, while FIGURE 4-16 shows the coupled fluid/solid temperature and resulting thermal distribution within the solid.

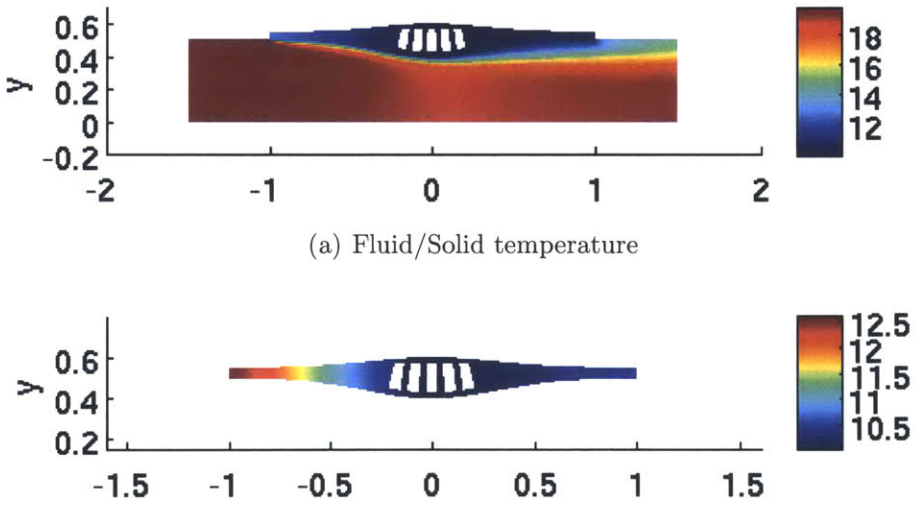


(a) Fluid density

(b) Fluid streamwise momentum

(c) Fluid Mach

FIGURE 4-15: Numerical solution to the cooled nozzle flow



(a) Fluid/Solid temperature

(b) Thermal variation in solid

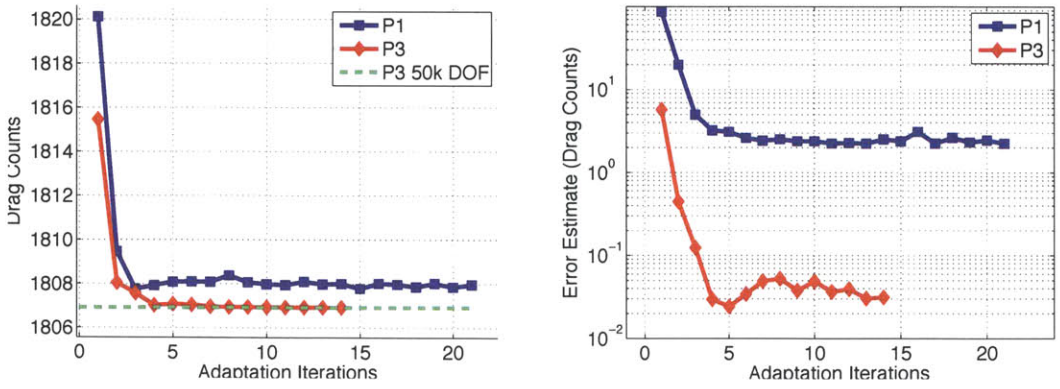
FIGURE 4-16: Temperature solution to the cooled nozzle flow

As expected in a fluid solution with a Prandtl number $Pr = 1$, a growing thermal and viscous boundary layer with similar growth rates is observed. A maximum Mach

number of $M_{max} = 0.627$ at $x = 0$ is detected, and, due to blockage effects, the exit free-stream Mach remains well above the inlet Mach. The leading edge of the solid reaches a maximum temperature of $T_{max} = 12.57$, and the cooling channels help maintain a cooler temperature distribution throughout the solid.

Drag Adapted Cooled Nozzle Flow

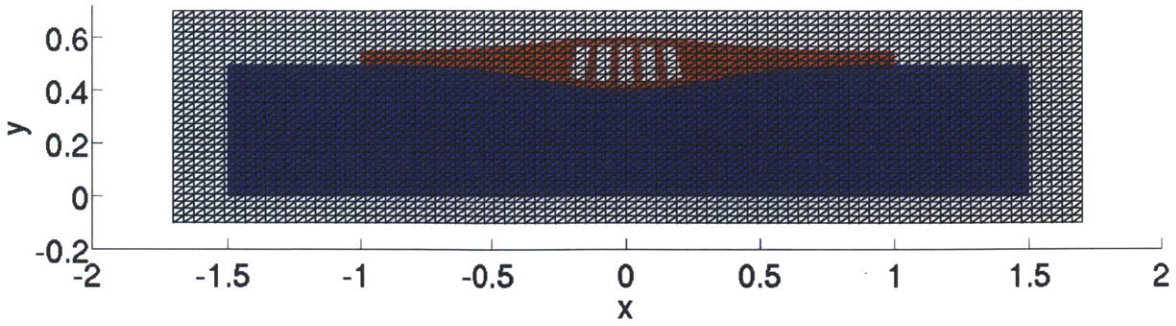
Here, a super-convergence study is performed by adapting to drag on the no-slip wall interface. The error in the output is computed over a range of p-order's and degrees of freedom (DOF), and plotted to show the convergence. For each case, a number of adaptation iterations are computed to ensure adaptation convergence as illustrated in FIGURE 4-17, and the corresponding $p = 1$ and $p = 3$ meshes are shown in FIGURES 4-18 and 4-18. Note that the $p = 3$ case has a limited number of adaptation iterations due to convergence issues originating from a cut element on the interface. It is suspected that the issue is caused by the interface discretization formulation. The computed output from the last 8 adaptation iterations is then averaged to represent the optimal discrete solution at a given p-order and DOF. A "true" output error is computed by taking the difference between the computed average output and the output from an adapted grid of 50k DOF (deemed "exact"). The resulting errors and convergence rates are shown in FIGURE 4-19.



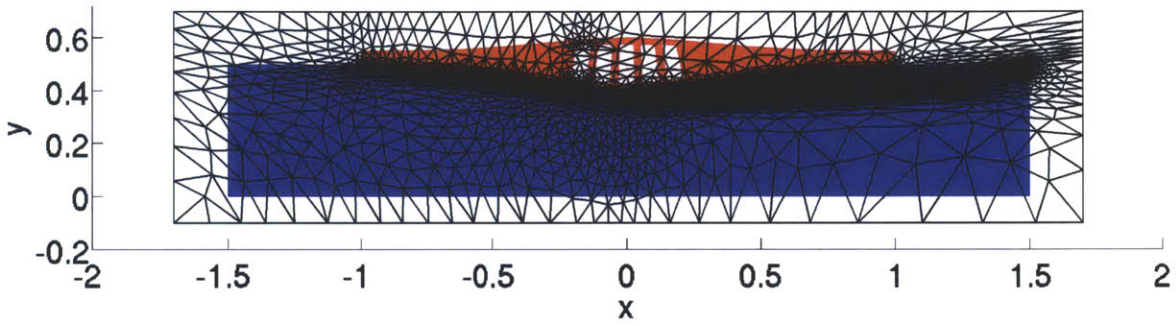
(a) Drag vs. adaptation iteration

(b) Drag error estimate convergence

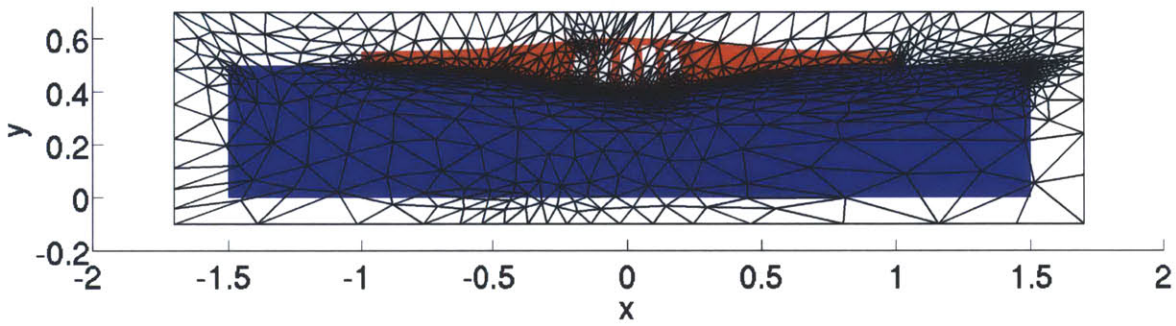
FIGURE 4-17: Cooled nozzle drag adaptation history for 16k DOF



(a) Initial background mesh

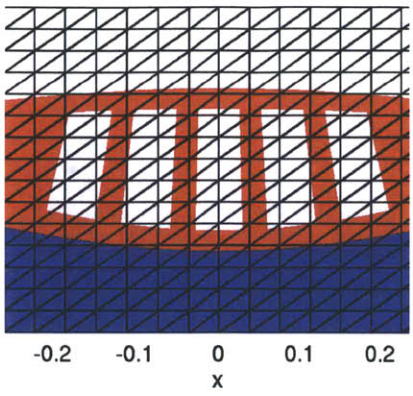


(b) Drag adapted p=1 16k DOF

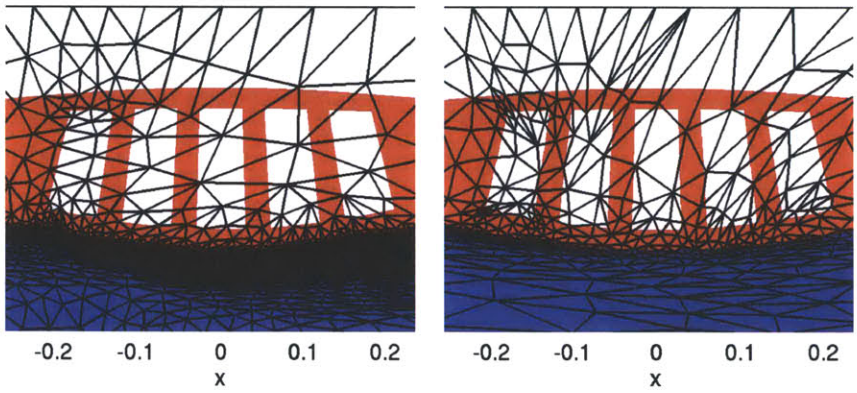


(c) Drag adapted p=3 16k DOF

FIGURE 4-18: Cooled nozzle drag adapted meshes



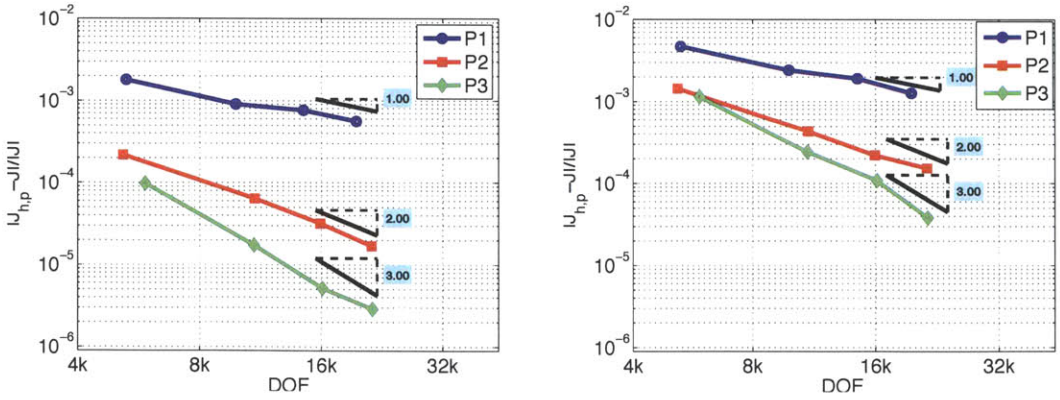
(a) Initial background mesh zoom



(b) Drag adapted p=1 16k DOF zoom

(c) Drag adapted p=3 16k DOF zoom

FIGURE 4-18: Cooled nozzle drag adapted meshes zoom



(a) Drag convergence

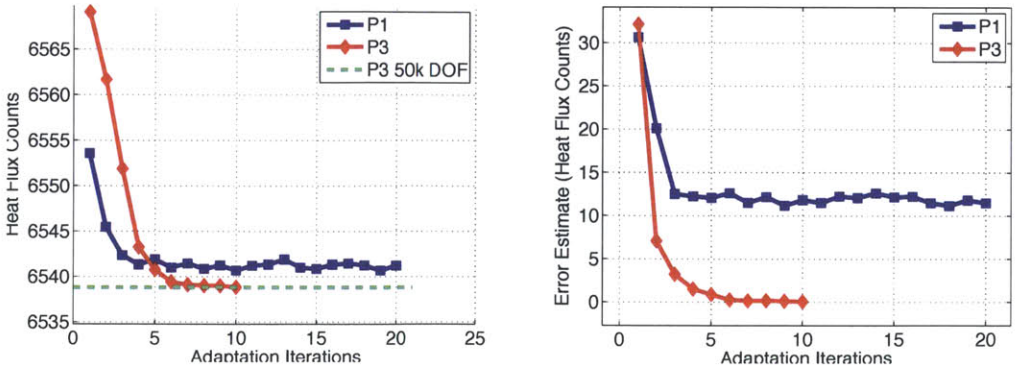
(b) Heat flux convergence

FIGURE 4-19: Cooled nozzle drag adapted error convergence

As depicted in the drag adaptation error history, both $p = 1$ and $p = 3$ 16k DOF solutions converge in ~ 10 adaptation iterations from a moderately refined uniform grid. In the final adapted meshes, anisotropic refinement resolves the boundary layer near the interface, and coarsens elements throughout the solid and null regions. With high resolution of the boundary layer, super-convergent rates of $\text{DOF}^{-\text{p}}$ (h^{2p}) for $p = 1, 2$, and nearly 3 are achieved for the drag error. Though near optimal rates are also observed for the calculated heat flux error across the interface, the level of accuracy is much lower than in the drag prediction. This degradation is due to the lack of refinement in the solid region, which is necessary for resolving the temperature for heat flux calculations, especially near the cooling channels.

Heat Flux Adapted Cooled Nozzle Flow

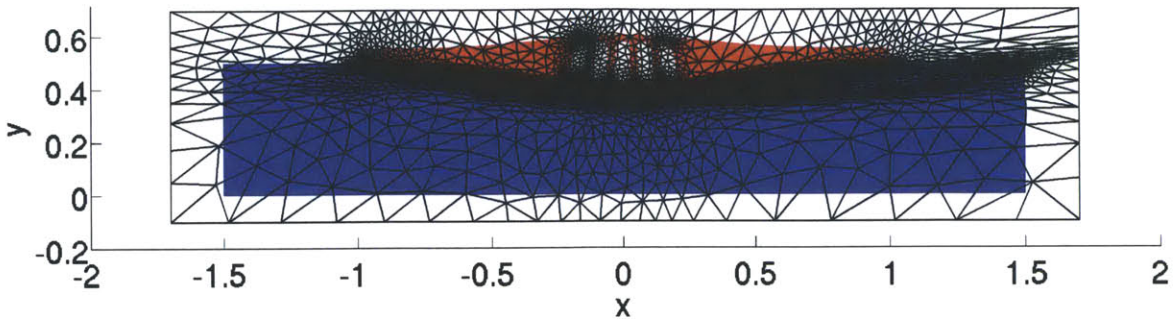
Performing the same procedure as above, but adapting to heat flux instead yields near super-convergent rates for the heat flux and drag error, though degrades the accuracy in computing drag. FIGURE 4-20 shows the convergence history for a $p = 1$ and $p = 3$ solution, while FIGURES 4-21 and 4-22 show the corresponding meshes. Note again that the interface discretization is suspected for causing convergence issues, which ultimately limits the number of converged adaptation iterations achieved for the $p = 3$ case to 5. As such, both the drag and heat flux outputs are computed by averaging over the last 5 adaptation iterations, and the resulting output error convergence is shown in FIGURE 4-23.



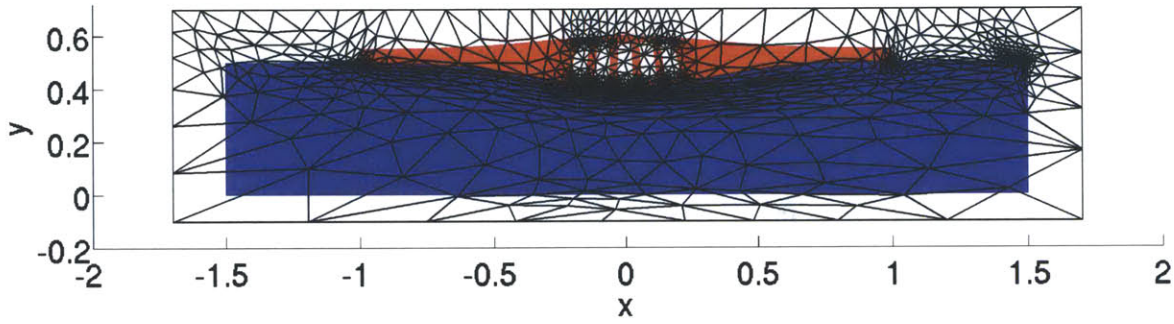
(a) Heat flux vs. adaptation iteration

(b) Heat flux error estimate convergence

FIGURE 4-20: Cooled nozzle heat flux adaptation history for 16k DOF

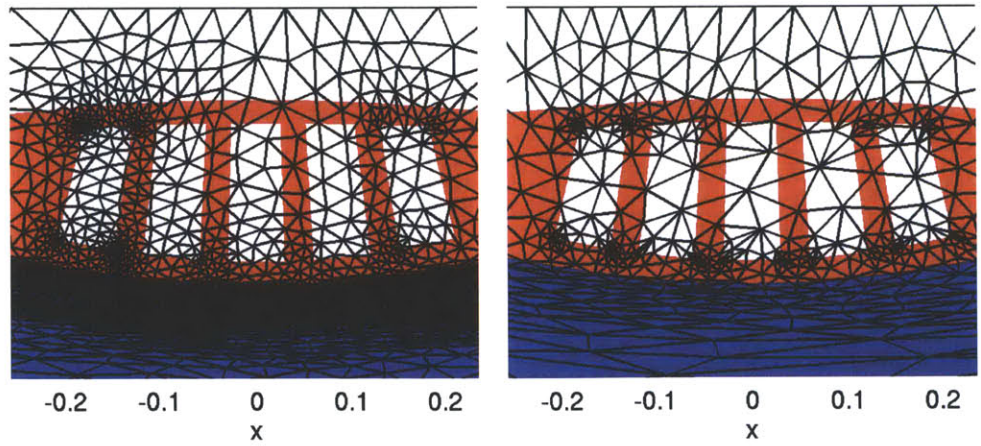


(a) Heat flux adapted p=1 16k DOF



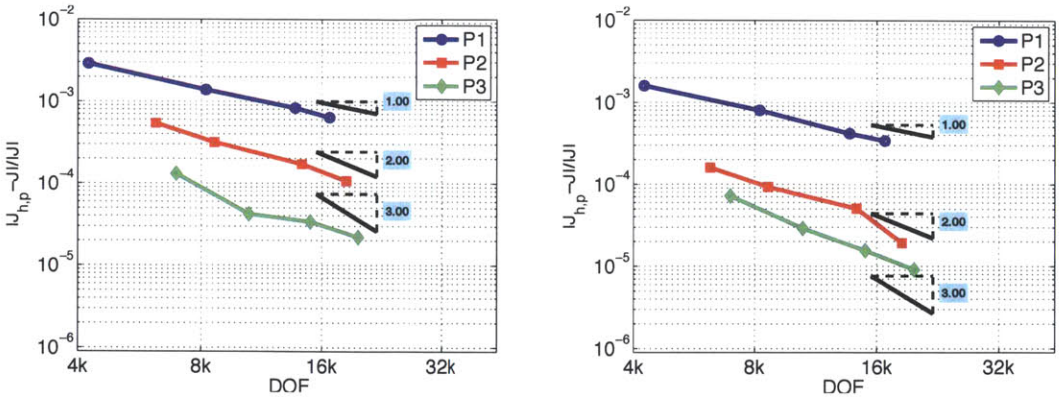
(b) Heat flux adapted p=3 16k DOF

FIGURE 4-21: Cooled nozzle heat flux adapted meshes



(a) Heat flux adapted p=1 16k DOF zoom (b) Heat flux adapted p=3 16k DOF zoom

FIGURE 4-22: Cooled nozzle heat flux adapted meshes zoom



(a) Drag convergence

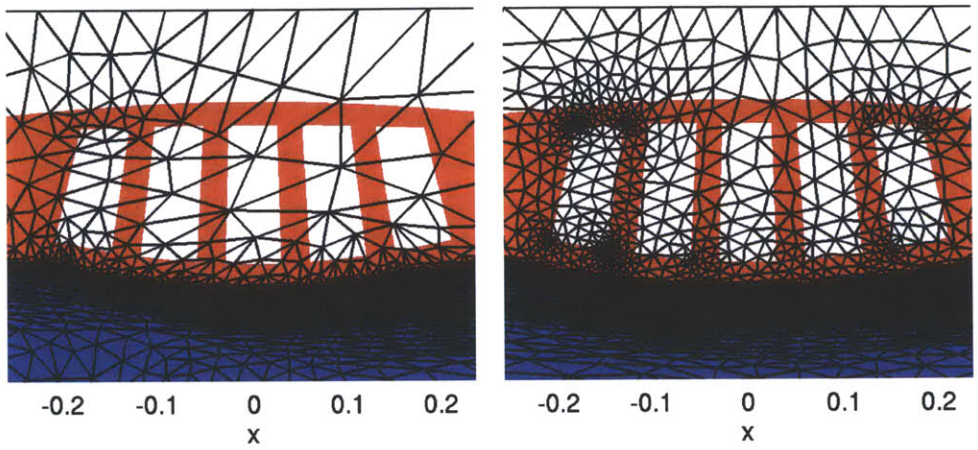
(b) Heat flux convergence

FIGURE 4-23: Cooled nozzle heat flux adapted error convergence

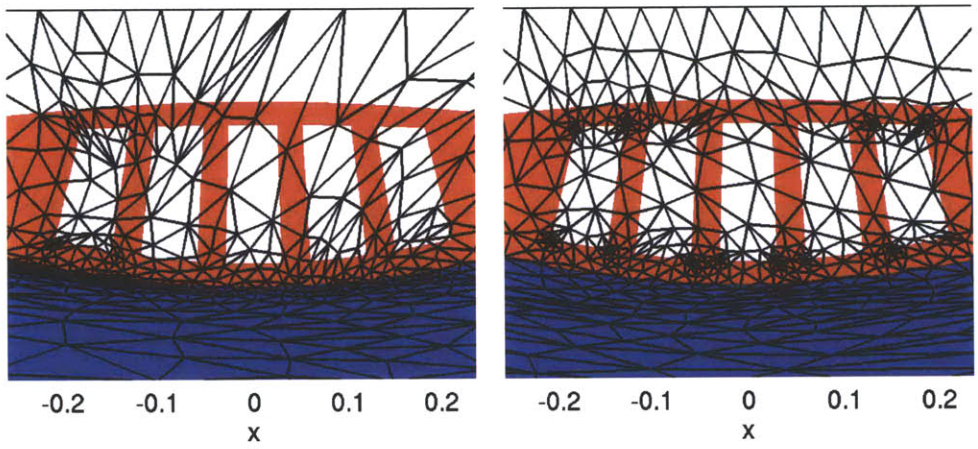
As shown in the heat flux adaptation error history, both $p = 1$ and $p = 3$ 16k DOF solutions converge in $\sim$ 10 adaptation iterations from a moderately refined uniform grid. In the final adapted meshes, anisotropic refinement is still observed in the fluid near the interface, which is still important for heat flux calculations, though several degrees of freedom are dispersed throughout the solid with strong mesh grading on the corners of the cooling channels. As such, the heat flux error converges with near super-convergent rates of $\text{DOF}^{-\text{p}}$ (h^{2p}) for $p = 1$ and 2, and the magnitude in error is greatly reduced compared to the drag adapted case. For $p = 3$, a partial degradation in the error convergence is observed presumably due to the limited number of converged solutions.

The drag error convergence also maintains near-optimal rates when adapting to heat flux for $p = 1$ and 2, similar to the trend found in the compressible Poiseuille flow case in SECTION 4.2.4. However, the magnitude in drag error is about an order of magnitude higher than the case where the mesh is adapted to drag. This loss in accuracy is caused by the strong mesh grading at the corners of the cooling channels, which leaves fewer degrees of freedom in the boundary layer. Nonetheless, the boundary layer is still well resolved when adapting to heat flux, whereas the solid region receives far less refinement when adapting to drag. These results show that adapting to heat flux performs moderately better in achieving accurate predictions in both the heat flux and drag outputs, whereas adapting to drag is best if the output of interest

is drag alone. A side by side view of the drag and heat flux adapted meshes for $p = 1$ and $p = 3$ is illustrated in FIGURE 4-24 to clearly show the effects on the mesh by the choice of output used for mesh adaptation.



(a) Drag adapted p=1 16k DOF zoom (b) Heat flux adapted p=1 16k DOF zoom



(c) Drag adapted p=3 16k DOF zoom (d) Heat flux adapted p=3 16k DOF zoom

FIGURE 4-24: Cooled nozzle comparison of drag vs. heat flux adaptation

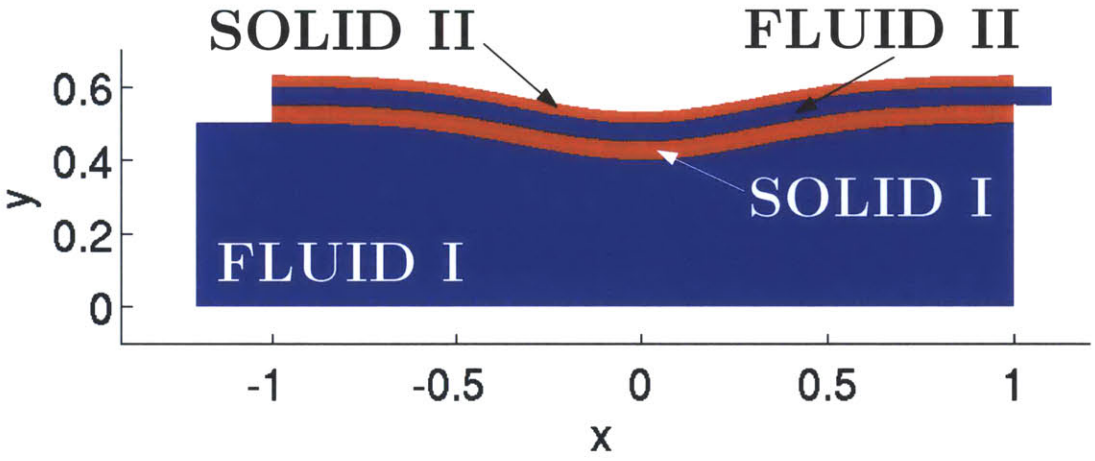
4.4 Navier-Stokes Multi-Flow Simulation

This case demonstrates the capability of the DG CHT framework in computing accurate solutions for models consisting of multiple fluid and solid domains. The ability to efficiently and accurately predict cooling flow rates for passages of complex geometry and coupled high heat transfer rates is pivotal in early design phases, and

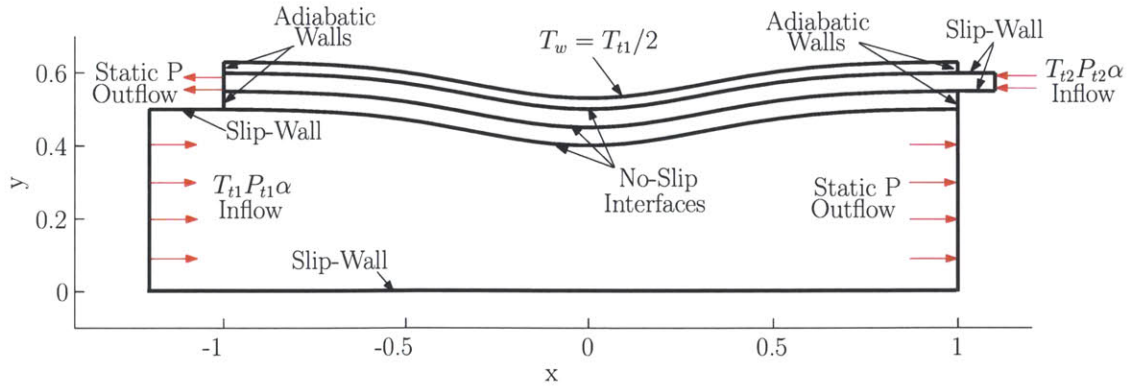
is of high interest for turbomachinery and rocket engine applications. Though these fields typically observe turbulent and/or reacting flows in design, the case shown here focuses on the capability of the CHT framework in computing high-order solutions for multiple laminar flows, which can be extended to turbulent or reacting flows.

4.4.1 Conjugate Model

FIGURE 4-25(a) and FIGURE 4-25(b) show the separate regions and sketch of the computational domain used to model the conjugate heat transfer between multiple flows.



(a) Region definition



(b) Sketch of computational domain

FIGURE 4-25: Multi-regioned flow model

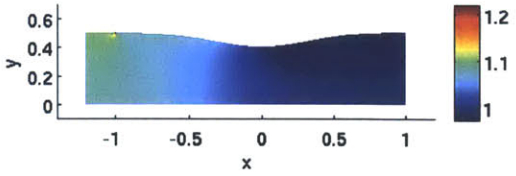
For this case, both inflow conditions are set by uniform counter-flows with a non-dimensionalized total pressure and temperature calculated from a reference Mach $M^1_{ref} = M^2_{ref} = 0.5$. In particular, the total pressure and temperature are calculated by EQUATION 4.21 where the reference conditions are defined by the core flow (fluid I) inlet conditions. Solutions to the two flows are also obtained with Reynolds numbers of $Re^1_H = 100$ and $Re^2_H = 10,000$, set by the non-dimensionalized viscosity with reference to the core flow (fluid I) inlet conditions: $\overline{\mu^i} = \mu^i/(\rho_{ref} V_{ref} H) = 1/Re^i_H = 10^{-2}|_{i=1}, 10^{-4}|_{i=2}$ where $H = 1$ represents the entire core flow channel height. The thermal conductivity ratios are $\kappa^1_r = \kappa^2_s/\kappa^1_f = 1$, $\kappa^2_r = \kappa^2_s/\kappa^2_f = 71$, and $\kappa^3_r = \kappa^2_s/\kappa^1_s = 100$. The Prandtl numbers are $Pr_1 = 0.71$ and $Pr_2 = 0.1$, and both fluid's viscosity vary with temperature according to Sutherland's law. The solid I domain is bounded by adiabatic walls to the left and right, and fluid interfaces. The boundary condition on the top wall of solid II is a Dirichlet temperature condition that sets the wall temperature to $T_w = T_{t1}/2$.

4.4.2 Adapted Solutions and Output Super-convergence

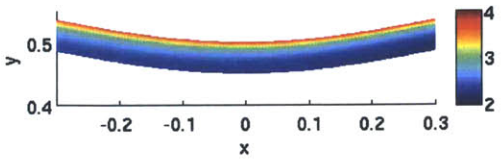
Here, mesh adaptation is performed on the multi-flow problem to demonstrate super-convergent rates of output errors on a more complex multi-regioned cut mesh. For this problem, the primary outputs of interest are the secondary flow's (fluid II) mass flux and the heat flux along the secondary flow's interface walls.

Numerical Solution to Multi-Flow Model

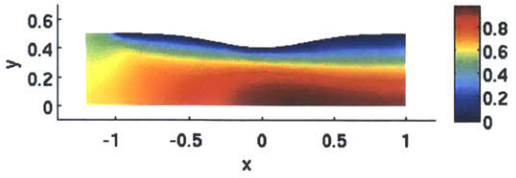
FIGURE 4-26 shows the resulting fluid's density and streamwise momentum from solving on a 24k DOF $p = 2$ mesh while FIGURE 4-27 shows the multi-regioned coupled temperature.



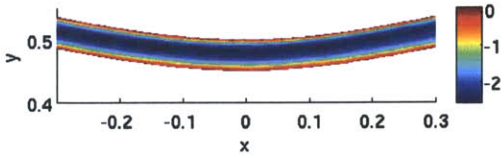
(a) Fluid I density



(b) Fluid II density

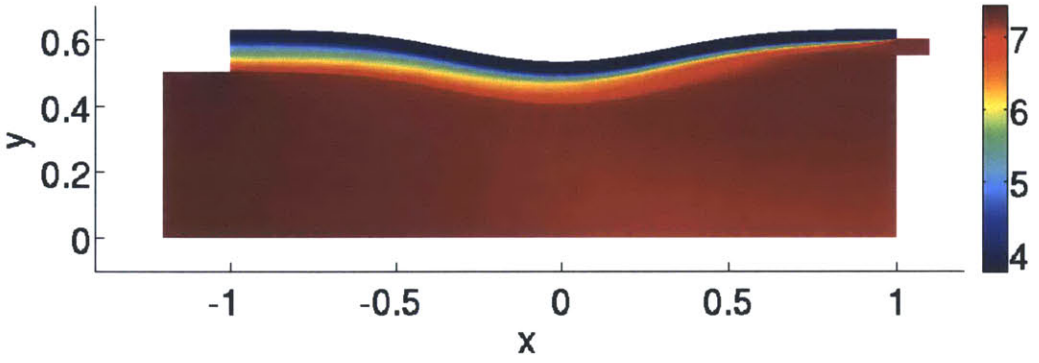


(c) Fluid I x-momentum

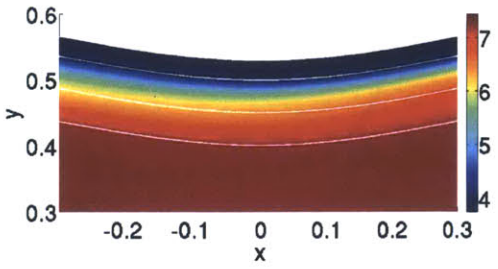


(d) Fluid II x-momentum

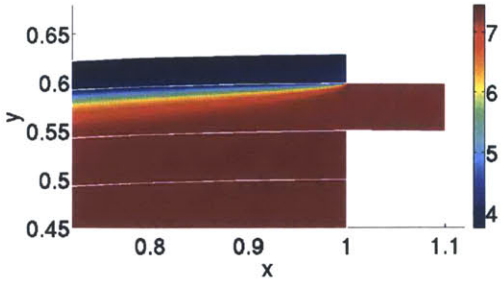
FIGURE 4-26: Numerical solution to the multi-flow problem



(a) Fluid/Solid temperature



(b) Fluid/Solid temperature channel



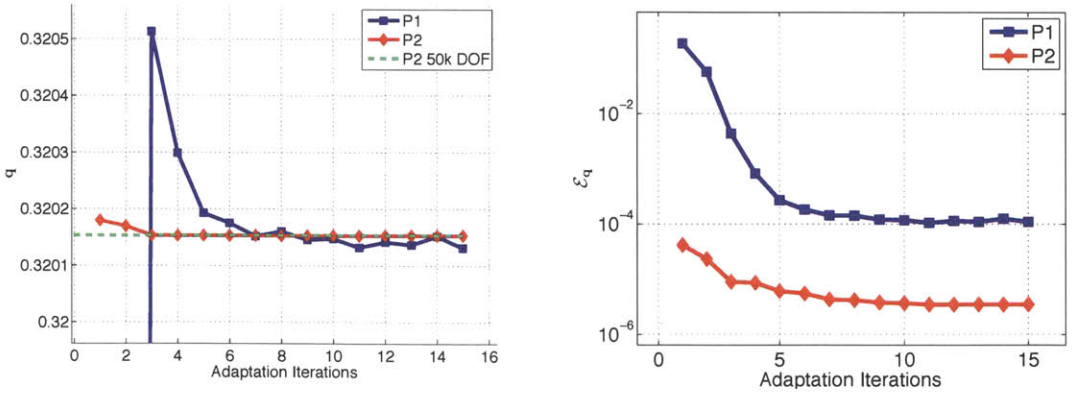
(c) Fluid/Solid temperature inlet

FIGURE 4-27: Temperature solution to the multi-flow problem

In both fluid solutions, a growing viscous boundary layer is observed, while a pronounced thermal boundary layer is apparent only in the secondary flow. The secondary flow forms a 'dense' film layer along the colder outer surface, and the thermal gradient penetrates the hot inner wall closer to the secondary flow outlet.

Heat Flux Adaptation

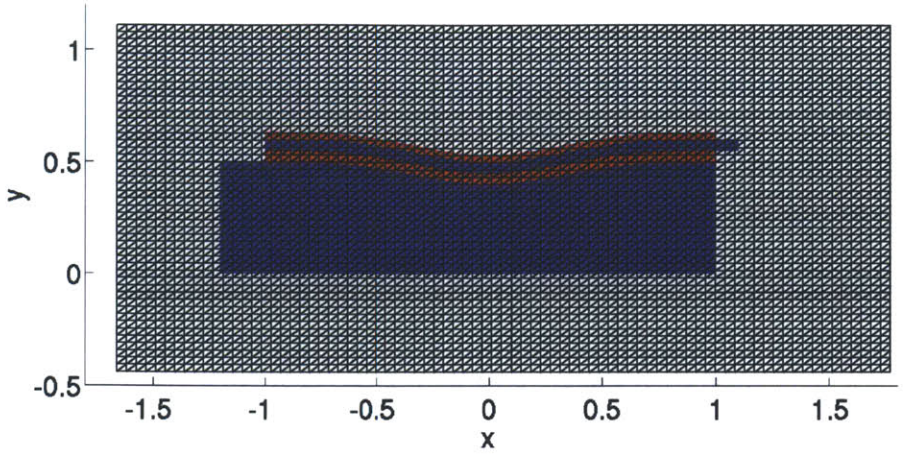
A super-convergence study is performed by adapting to heat flux on the secondary flow's no-slip wall interfaces. The error in the output is computed over increasing p-order's and degrees of freedom (DOF). For each case, a series of adaptation iterations are computed to ensure adaptation convergence as illustrated in FIGURE 4-28. The corresponding $p = 1$ and $p = 2$ meshes are shown in FIGURE 4-29 and FIGURE 4-30. The computed output from the last 8 adaptation iterations is then averaged to represent the optimal discrete solution at a given p-order and DOF. A "true" output error is computed by taking the difference between the computed average output and the output from a $p = 2$ adapted grid of 50k DOF. The resulting errors and convergence rates are shown in FIGURE 4-31.



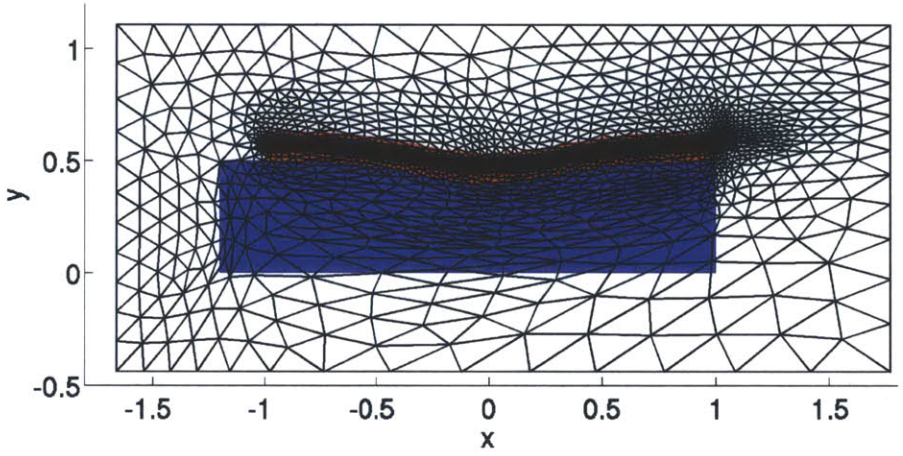
(a) Heat flux vs. adaptation iteration

(b) Heat flux error estimate convergence

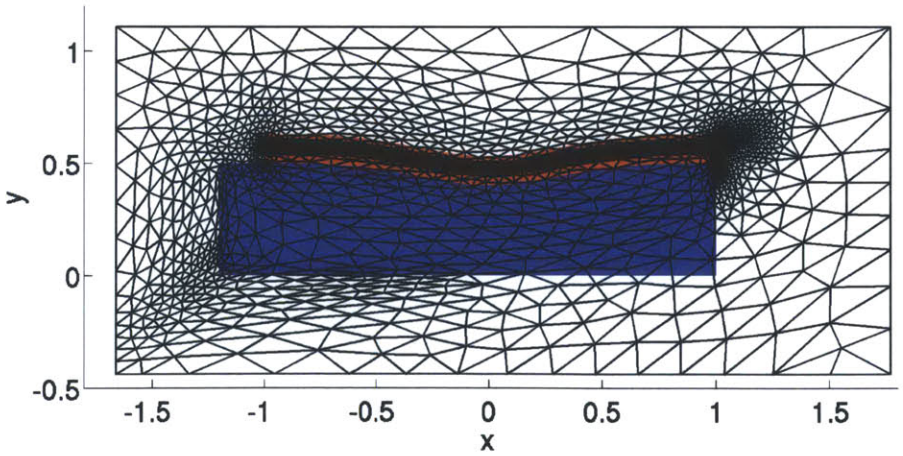
FIGURE 4-28: Multi-flow 24k DOF heat flux adaptation history



(a) Initial background mesh

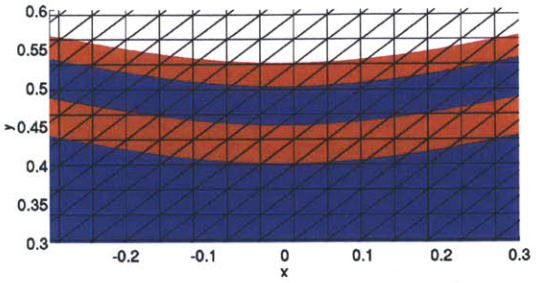


(b) Heat flux adapted p=1 24k DOF

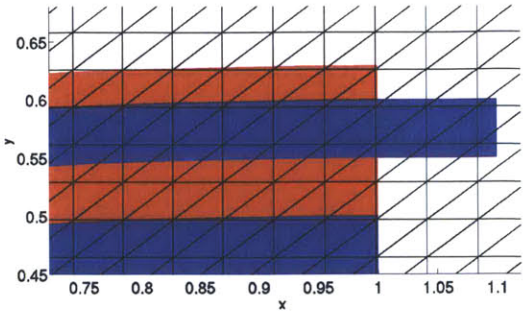


(c) Heat flux adapted p=2 24k DOF

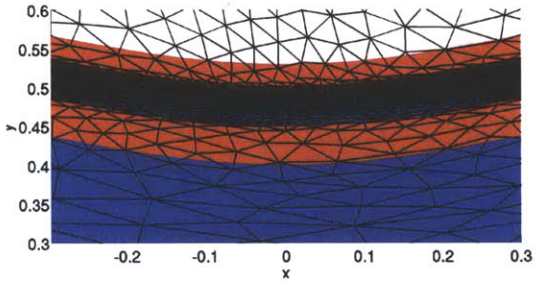
FIGURE 4-29: Multi-flow heat flux adapted meshes



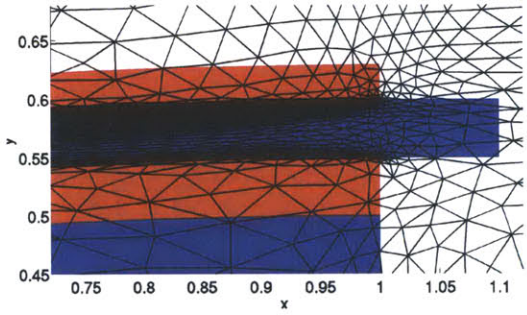
(a) Initial background mesh zoom



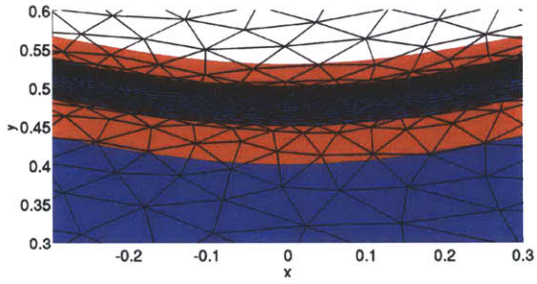
(b) Initial background mesh inlet



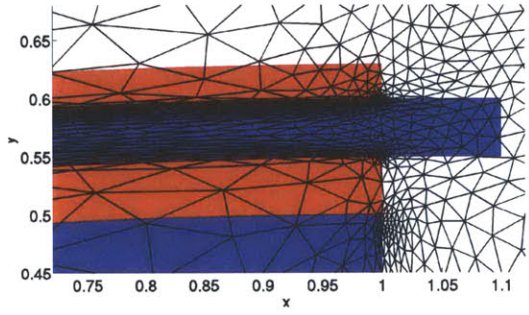
(c) Heat flux adapted p=1 24k DOF zoom



(d) Heat flux adapted p=1 24k DOF inlet

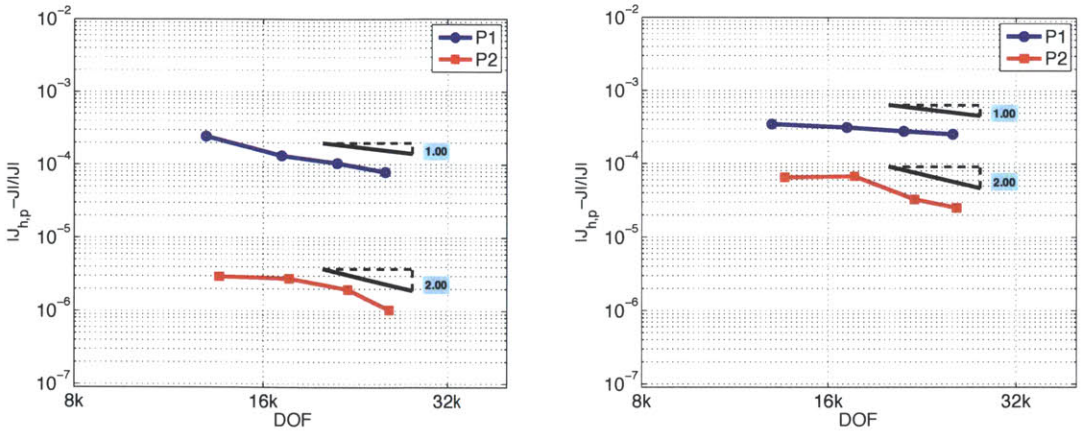


(e) Heat flux adapted p=2 24k DOF zoom



(f) Heat flux adapted p=2 24k DOF inlet

FIGURE 4-30: Multi-flow heat flux adapted meshes zoom



(a) Heat flux convergence

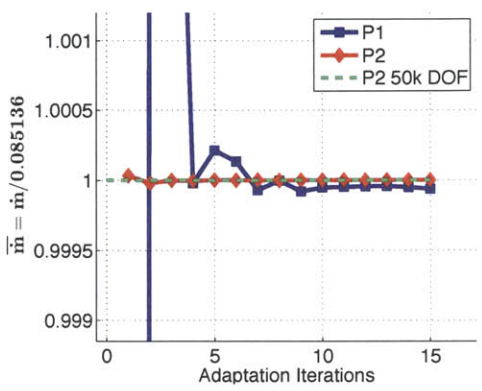
(b) Mass flux convergence

FIGURE 4-31: Multi-flow heat flux adapted error convergence

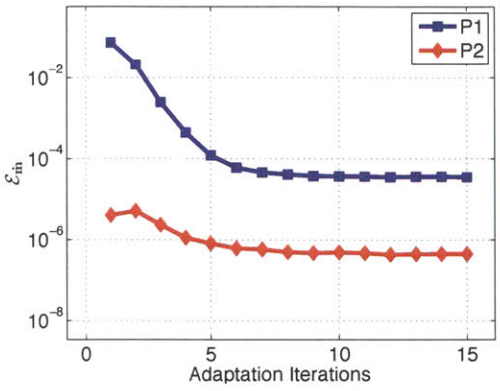
As illustrated in the heat flux error history in FIGURE 4-28, both $p = 1$ and $p = 2$ 24k DOF solutions converge in ~ 12 adaptation iterations from a moderately refined uniform grid. The adapted meshes show a strong mesh grading within the secondary flow path in order to resolve the boundary layer for accurate convective heat flux calculations. Additionally, the computed heat flux error converges near the super-convergent rate of $\text{DOF}^{-\text{p}}$ (h^{2p}) for $p = 1$, and 2, as well as the secondary flow mass flux. Nevertheless, adapting to the interface heat flux yields greater accuracy in the predicted heat flux over the predicted mass flux.

Mass Flux Adaptation

Performing the same procedure as above, but adapting to the secondary flow mass flux instead yields improved mass flux predictions, though the accuracy in the heat flux output degrades. FIGURE 4-32 shows the convergence history for a $p = 1$ and $p = 2$ solution, while FIGURES 4-33 and 4-34 show the corresponding meshes. Again, both the mass and heat flux outputs are computed by averaging the last 8 adaptation iterations, and the resulting output error convergence is shown in FIGURE 4-35.

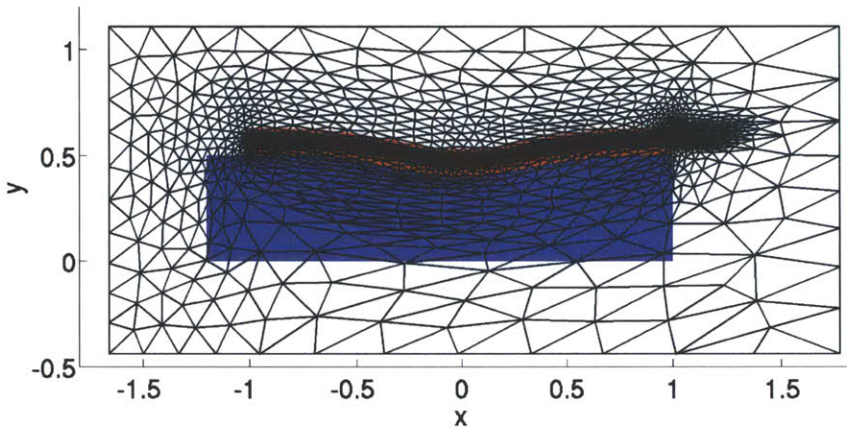


(a) Mass flux vs. adaptation iteration

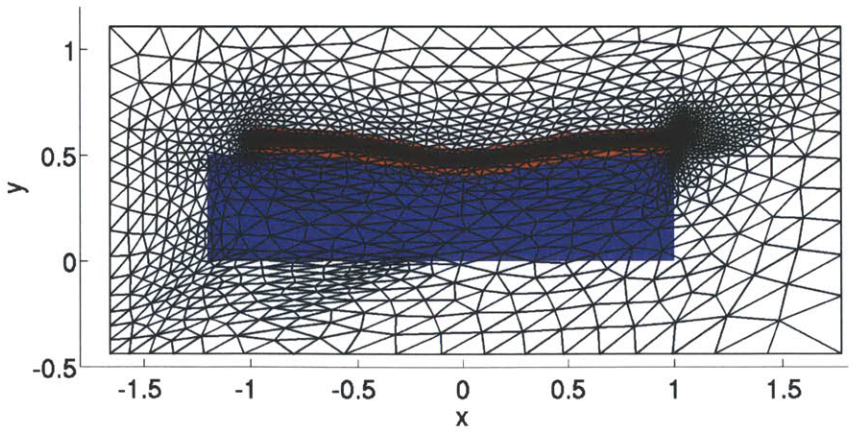


(b) Mass flux error estimate convergence

FIGURE 4-32: Multi-flow 24k DOF normalized mass flux adaptation history

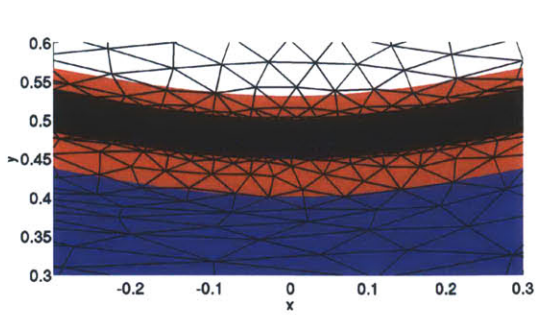


(a) Mass flux adapted p=1 24k DOF

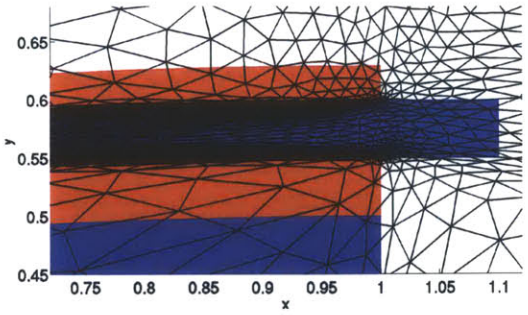


(b) Mass flux adapted p=2 24k DOF

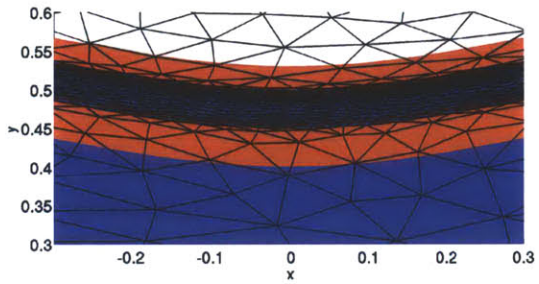
FIGURE 4-33: Multi-flow mass flux adapted meshes



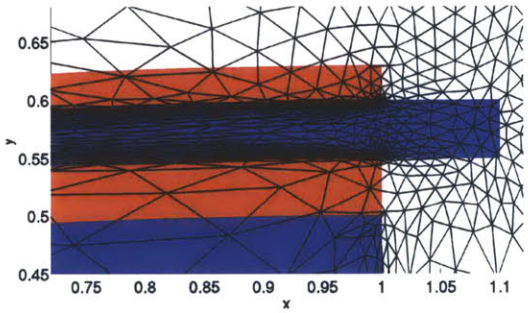
(a) Mass flux adapted p=1 24k DOF zoom



(b) Mass flux adapted p=1 24k DOF inlet

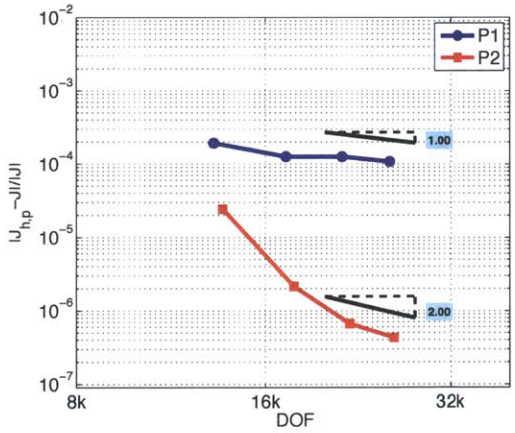


(c) Mass flux adapted p=2 24k DOF zoom

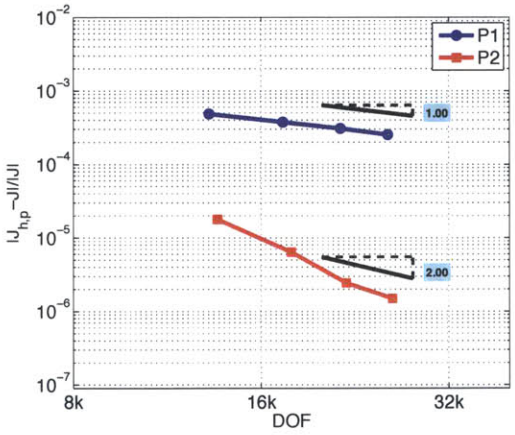


(d) Mass flux adapted p=2 24k DOF inlet

FIGURE 4-34: Multi-flow mass flux adapted meshes zoom



(a) Mass flux convergence



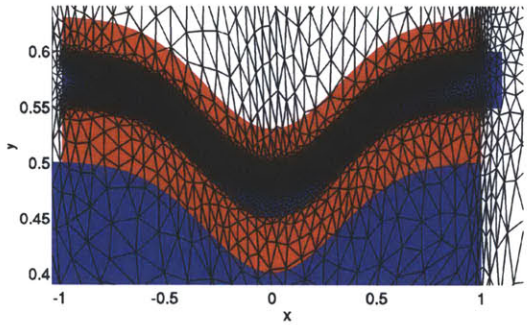
(b) Heat flux convergence

FIGURE 4-35: Multi-flow mass flux adapted error convergence

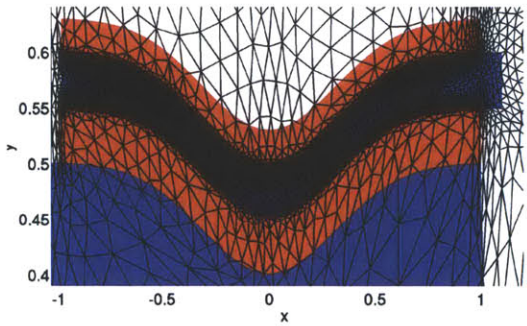
As depicted in the mass flux error history in FIGURE 4-32, both $p = 1$ and $p = 2$ 24k DOF solutions converge in ~ 12 adaptation iterations from a moderately refined uniform grid. The adapted meshes again show a strong mesh grading within the

secondary flow path in order achieve sufficient resolution for accurate mass flux calculations. Additionally, the mass flux error converges near the super-convergent rate of $\text{DOF}^{-\text{p}}$ (h^{2p}) for $p = 1$, and 2, as well as the interface heat flux output. Since the prediction in the mass flux output greatly improves with mass flux adaptation, and the heat flux prediction is relatively insensitive to the adapted output choice, mass flux adaptation performs better in achieving accurate predictions in *both* heat and mass flux outputs.

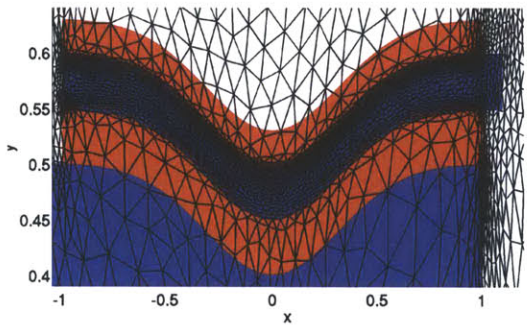
Though the differences are subtle, the heat and mass flux adapted meshes have distinguishable features. To better see the differences, a scaled side by side view of the heat and mass flux adapted meshes for $p = 1$ and $p = 2$ is shown in FIGURE 4-36. Observed is a greater resolution of the viscous boundary layer on both secondary flow interface walls for the mass flux adapted mesh. Since this resolution encompasses the same refined areas in the heat flux adapted case, it is expected that adapting to mass flux would achieve better predictions for a combined interest.



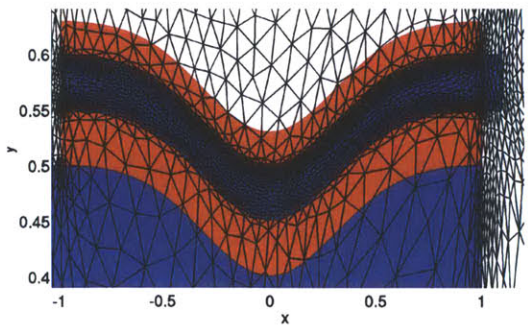
(a) Heat flux adapted p=1 24k DOF



(b) Mass flux adapted p=1 24k DOF



(c) Heat flux adapted p=2 24k DOF



(d) Mass flux adapted p=2 24k DOF

FIGURE 4-36: Multi-flow adapted mesh comparison (NOT TO SCALE)

Chapter 5

Conjugate RANS Heat Transfer

In this chapter, the same conjugate heat transfer framework developed for laminar CHT is applied to RANS calculations. Since many aerospace applications involve high-Reynolds turbulent flows, the ability to solve the conjugate problem in these environments becomes of high engineering interest. For the models presented here, the Spalart-Allmaras (SA) turbulent model [70] is used to close the Reynolds Averaged Navier-Stokes equations on the fluid domain, while the steady state heat conduction equation is simultaneously solved on the solid domain. SECTION 5.1 details the application of the DG formulation to a RANS-heat interface, SECTION 5.2 shows the convergence of a baseline flat slab case, and SECTION 5.3 demonstrates the capability and importance of high-order, mesh-adaptive, cut-cell methods for multi-regioned conjugate RANS problems.

5.1 Interface Conditions for RANS CHT

For 2-D conjugate RANS heat transfer, the fluid is modeled with the RANS-SA equations with a conservative state vector $\mathbf{u}^{(1)} = [\rho, \rho\mathbf{u}, \rho E, \rho\tilde{\nu}]^T$, while the solid is modeled with the heat conduction equation with a state vector $\mathbf{u}^{(2)} = T$. See APPENDIX A.1 and APPENDIX A.3 for a detailed formulation of the heat and RANS-SA equations.

Just as in the Navier-Stokes CHT interface, the same three constraints at the interface are imposed:

$$\begin{aligned} u = v &= 0, && \text{no} - \text{slip condition} \\ T^{(1)} &= T^{(2)}, && \text{temperature continuity} \\ \kappa_T^{(1)} \nabla T^{(1)} \cdot \mathbf{n}^{(1)} &= -\kappa_T^{(2)} \nabla T^{(2)} \cdot \mathbf{n}^{(2)}, && \text{heat flux continuity} \end{aligned}$$

where κ_T is the thermal conductivity.

5.1.1 Interface State and Discretization

The inviscid and viscous interface discretization definition again relies on the construction of a coupled interface state, $\mathbf{u}_\Sigma$, which is used to compute each domain's numerical flux. The choice of the constructed interface state is:

$$\mathbf{u}_\Sigma^{(1)} = \begin{bmatrix} \rho^{(1)} \\ \mathbf{0} \\ \rho^{(1)} \{T\} \frac{R}{\gamma - 1} \\ 0 \end{bmatrix}, \quad \text{and} \quad \mathbf{u}_\Sigma^{(2)} = \{T\} \tag{5.1}$$

where $\{T\} = 0.5(T^{(1)} + T^{(2)})$ is the average temperature across the interface. FIGURE 5-1 shows an illustration of the interface states used in the interface residual evaluation.

The inviscid and viscous discretization for the RANS solid wall interface directly follows the definition for the NS solid wall interface given in SECTION 4.1.1 with the addition of the SA equation. The resulting coupled viscous fluxes, $\widehat{\mathcal{A}\sigma}^{(\mathrm{i})} \cdot \mathbf{n}^{(\mathrm{i})}$, for $i = 1, 2$ on each domain Ω_i, are defined by:

$$\widehat{\mathcal{A}\sigma}^{(1)} \cdot \mathbf{n}^{(1)} = \begin{cases} (\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{mass}} \\ (\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{momentum}} \\ 0.5((\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{energy}} + \kappa_T^{(2)} \sigma_h^{(2)} \cdot \mathbf{n}^{(1)}) \\ (\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(1)})_{\text{SA}} \end{cases} \tag{5.2}$$

$$\widehat{\mathcal{A}\sigma}^{(2)} \cdot \mathbf{n}^{(2)} = 0.5((\mathcal{A}_\Sigma^{(1)} \sigma_h^{(1)} \cdot \mathbf{n}^{(2)})_{\text{energy}} + \kappa_T^{(2)} \sigma_h^{(2)} \cdot \mathbf{n}^{(2)}). \tag{5.3}$$

where the viscosity matrix is computed using the interface state: $\mathcal{A}_\Sigma^{(1)} = \mathcal{A}^{(1)}(\mathbf{u}_\Sigma^{(1)})$, and the lifted gradient, $\sigma_h^{(\text{i})}$, is given by EQUATION 4.7. Again, additional analysis on the current interface boundary condition is recommended to determine adjoint consistency and other discretization properties.

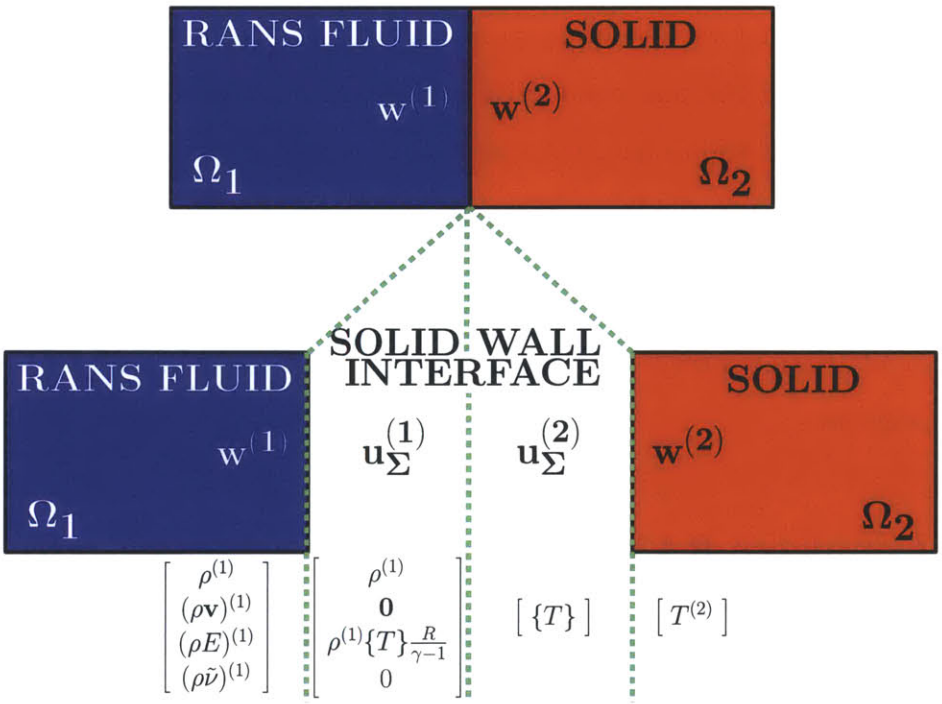


FIGURE 5-1: Sketch of RANS solid wall interface states used to compute numerical fluxes

5.2 Compressible Flow over a Cooled Slab

The compressible flow over a cooled slab problem serves as a benchmark capability of the CHT framework in computing solutions for conjugate turbulent flows. Numerous computational studies have been performed on the single fluid discipline for RANS flow along a flat plate. Nguyen *et al.* [55] presents solutions using a high-order discontinuous DG method, while Freeman and Roy [33], and NASA's turbulence modeling resource[1] provide solutions obtained by finite volume codes. Furthermore, Sun [71] has demonstrated the application of an adaptive cut-cell framework to subsonic RANS-SA RAE2822 airfoil cases.

Beyond the single discipline case, several investigations on the laminar conjugate flat slab problem have been performed. Mosaad [52] obtained a closed-form solution to the conjugate flat plate problem assuming an approximate boundary layer profile, and studied the effects of the local Brun number, while Kanna *et al.* [42] and Vynnycky *et al.* [74] used an analytical solution to investigate the effects of Reynolds number, Prandtl number, and thermal conductivity ratio on the interface Nusselt number. Similar analyses with additional heat sources are provided by Cole [23] and Rizk *et al.* [65].

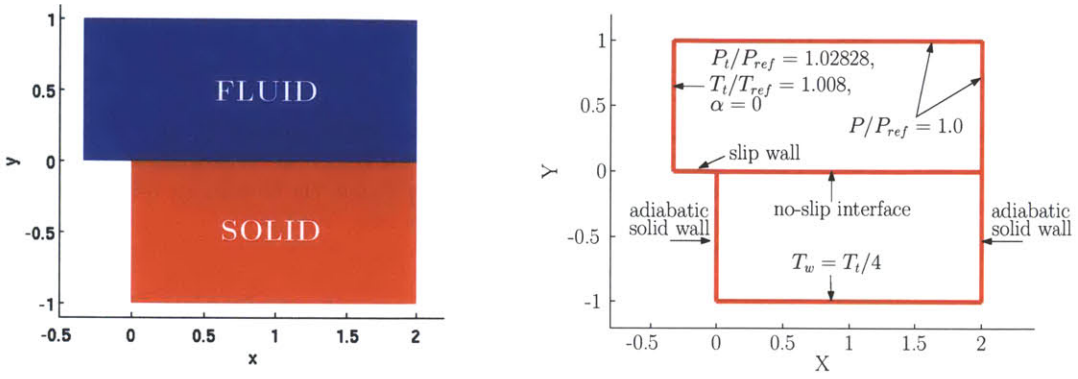
Here, a conjugate RANS flat slab problem is developed to investigate the robustness of the mesh adaptation algorithm when applied to a coupled RANS and heat equation interface problem.

5.2.1 Conjugate RANS Model

The model for the conjugate RANS case is based on the flat plate model used in a NASA turbulence modeling resource[2], which consists of a Mach $M = 0.2$ flow with a Reynolds number $Re_L = 5 \times 10^6$ for a $L = 1$ solid wall boundary. Under similar conditions, a flat slab model is developed with the addition of a solid region of length

[1] http://turbmodels.larc.nasa.gov/flatplate_sa.html
[2] http://turbmodels.larc.nasa.gov/flatplate.html

$2L$ and thickness L. FIGURE 5-2(a) and FIGURE 5-2(b) show the separate regions and sketch of the computational domain used to model the conjugate heat transfer over a cooled slab.



(a) Region definition

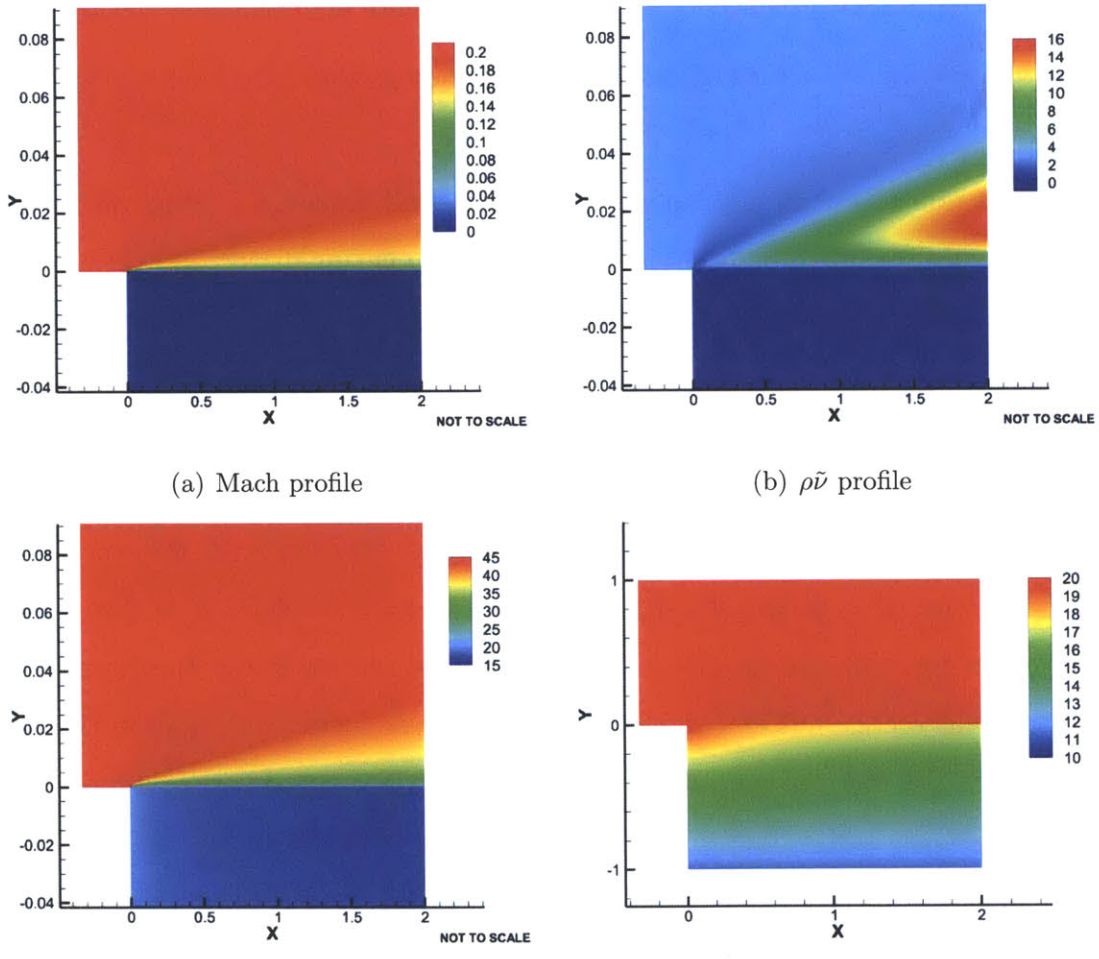
(b) Sketch of computational domain

FIGURE 5-2: RANS Slab flow model

The fluid region has a uniform inflow with a slip-wall entrance leading into a solid wall interface. The inlet condition is set by a non-dimensionalized total pressure and temperature calculated from a reference Mach $M_{ref} = 0.2$, and a free-stream SA working variable state is set to $\rho\tilde{\nu} = 3$. In particular, the total pressure and temperature are calculated by EQUATION 4.21 where the reference conditions are defined by the inlet conditions. Solutions to this case are obtained with a Reynolds number of $Re_L = 50,000$, set by the non-dimensionalized viscosity with reference to the inlet conditions: $\overline{\mu} = \mu/(\rho_{ref} V_{ref} L) = 1/Re_L = 2 \times 10^{-5}$ where $L = 2$ represents the length of the flat slab interface. The solid domain of thickness L is bounded by adiabatic walls on the left and right, a Dirichlet temperature condition on the lower wall that sets the wall temperature to a fourth of the free-stream stagnation temperature: $T_w = T_t/4$, and a fluid interface on the upper wall. This case uses a Prandtl number of $Pr = 0.72$, a specific heat ratio of $\gamma = 1.4$, and a conductivity ratio of $\kappa_r = \kappa_s/\kappa_f = 10^3$ to model the interaction between air and metal. Additionally, the fluid viscosity varies with temperature according to Sutherland's law. The primary output of interest studied for this case is the total heat flux on the slab interface.

5.2.2 Numerical Solution

FIGURE 5-3 illustrates the Mach, SA conservative state $\rho\tilde{\nu}$, and temperature profiles of the conjugate solution to a $p = 2$ 26k DOF case adapted to the total interface heat flux, while FIGURE 5-4 shows a series of solutions at vertical slices of constant x along the interface. A similar growth in both the momentum and thermal boundary layer is observed, and the thermal distribution within the solid peaks at the leading edge of the slab. Also detected is an undershoot of $\tilde{\nu}$ just prior to reaching the free-stream value near the edge of the boundary layer.



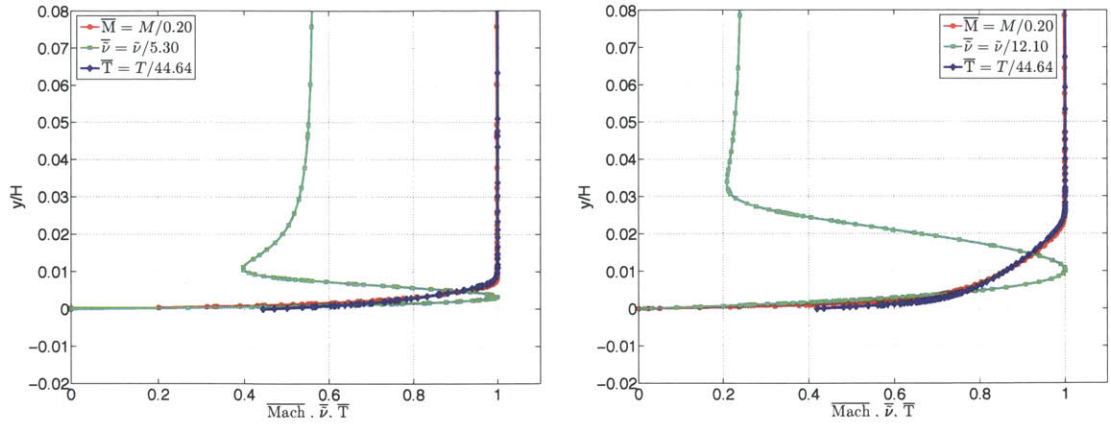
(a) Mach profile

(b) $\rho\tilde{\nu}$ profile

(c) Fluid temperature profile

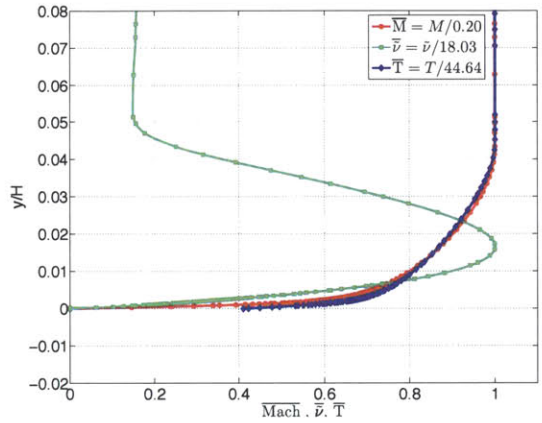
(d) Solid temperature profile

FIGURE 5-3: Numerical solution to RANS Flat Slab p=2 26k DOF



(a) $x = 0.25$

(b) $x = 1.0$



(c) $x = 1.75$

FIGURE 5-4: Vertical slices of normalized solutions to RANS Flat Slab p=2 26k DOF

In order to characterize the amount of heat flux through the interface, the Stanton number, which is typically used to describe the heat transfer in forced convection flows, is calculated. As reference, the Stanton number can be approximated through the Reynolds analogy as one half the skin friction: $St = C_f/2$, assuming $Pr = 1$. FIGURE 5-5 shows interface profiles of the non-dimensional temperature (θ), skin friction (C_f), and Stanton number (St) along the interface. The local temperature and Stanton number are functions of x, and are given by:

$$\theta(x) = \frac{T(x) - T_\infty}{T_w - T_\infty} \qquad St(x) = -\frac{\kappa_f \left.\frac{\partial T}{\partial y}\right|_{y=0}}{(\rho u)_e c_p (T(x) - T_e)} \tag{5.4}$$

where T_w and T_∞ are the slab lower wall and fluid free stream temperature, respectively, κ_f is the fluid's thermal conductivity, and all states $(\cdot)_e$ are evaluated at the edge of the boundary layer as a function of x. Note that for an open flat plate geometry, $(\cdot)_e$ equates to the free stream value $(\cdot)_\infty$. In calculating the Stanton number, the heat flux at the interface was determined by extracting the total energy viscous flux on the interface. Since the state gradients evaluated at arbitrary locations are 'noisy', a moving-average smoother was used to generate the heat flux trend along the interface. From this, the Stanton number appears to follow the same trend as the skin friction, and is slightly greater than $1/2C_f$.

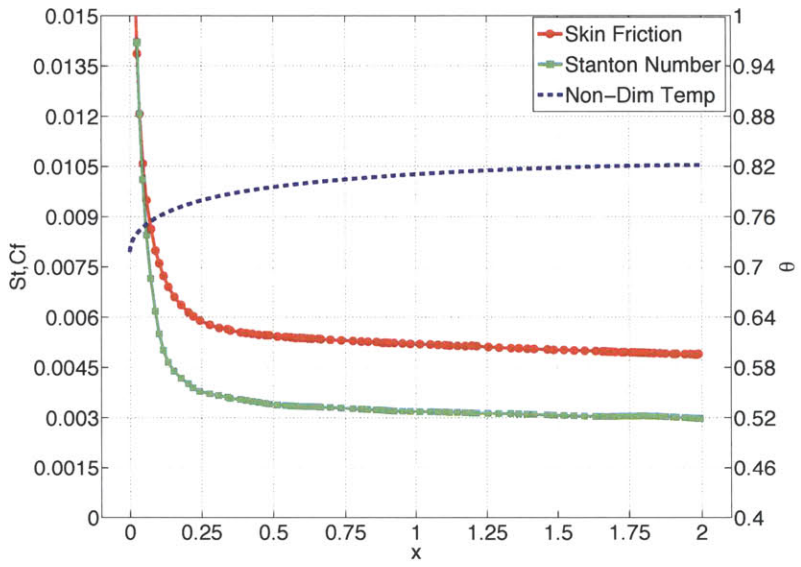
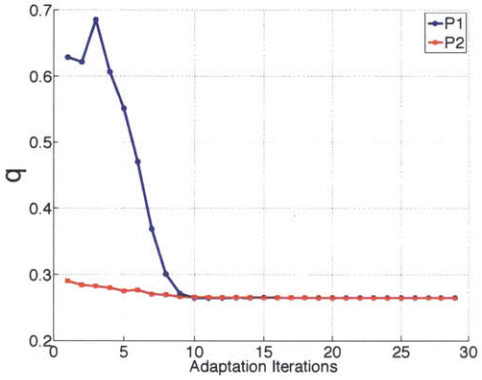


FIGURE 5-5: Interface thermal profile for RANS Flat Slab p=2 26k

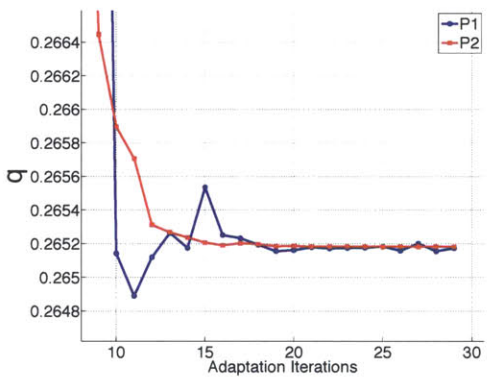
5.2.3 Optimized Meshes

A series of cases are constructed to observe the behavior of the cut-cell mesh adaptation algorithm when adapting to the interface heat flux. For each case, several adaptation iterations are computed until the error estimate of the output of interest converges. At each adaptation iteration, both the primal and dual adjoint solutions

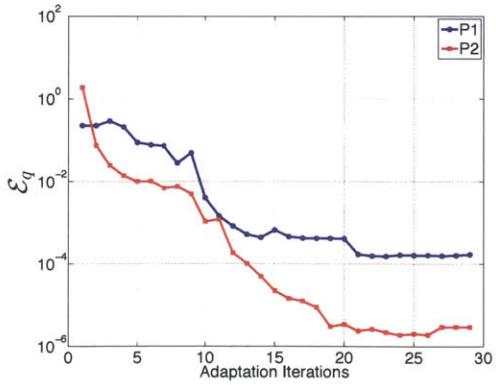
are converged in order to accurately calculate the localized dual weighted residual error used in optimizing the mesh. FIGURE 5-6 shows an example of the total heat flux and heat flux error estimate converging from a coarse uniform grid in approximately 25 adaptation iterations, while FIGURE 5-7 illustrates the domain geometry overlaid by the initial and optimized background meshes for $p = 1$ and $p = 2$ with 26k DOF. Additionally, FIGURE 5-8 shows mesh refinement near the edge of the boundary layer where a jump in the heat flux adjoint is observed. Note that a positive heat flux q is defined as heat transfer from the fluid to the solid domain.



(a) Heat flux convergence

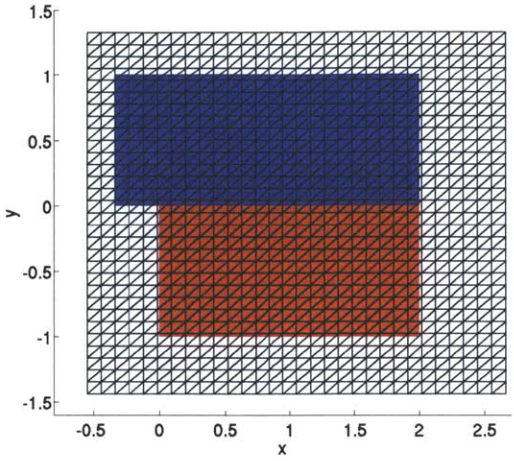


(b) Heat flux convergence zoom

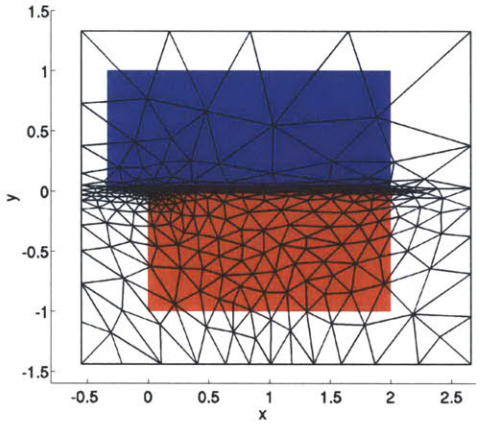


(c) Heat flux error convergence

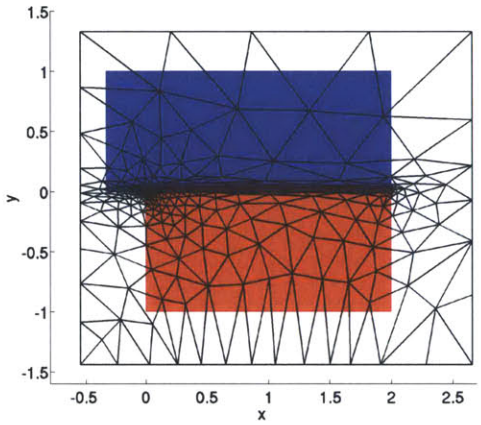
FIGURE 5-6: RANS Flat Slab heat flux adaptation history



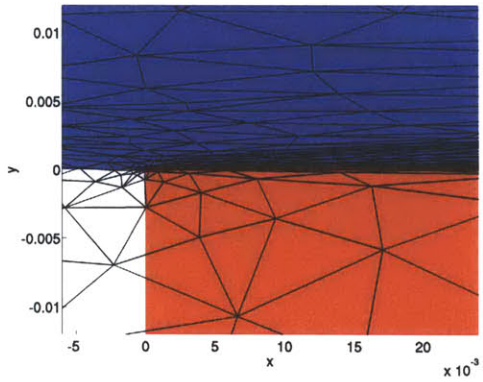
(a) Initial Background Mesh



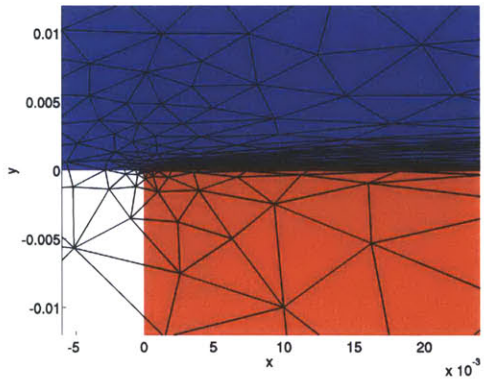
(b) p=1 26k DOF



(c) p=2 26k DOF

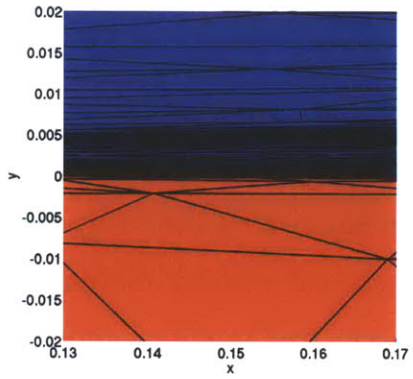


(d) p=1 26k DOF leading edge zoom

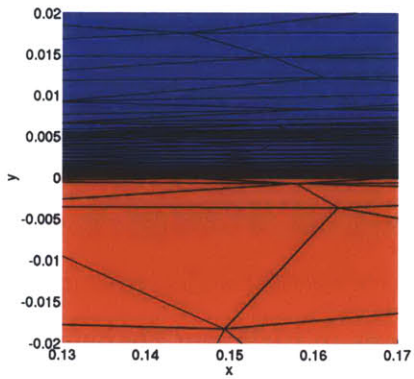


(e) p=2 26k DOF leading edge zoom

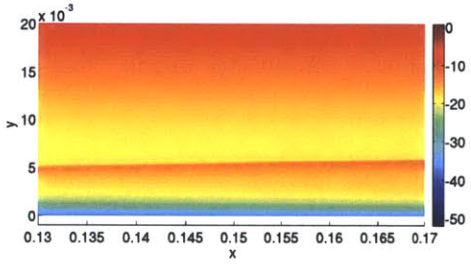
FIGURE 5-7: RANS Flat Slab heat flux adapted mesh



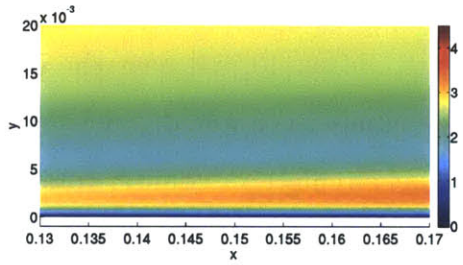
(a) p=1 26k DOF $x = 0.15$ zoom



(b) p=2 26k DOF $x = 0.15$ zoom



(c) Heat flux adjoint (mass) p=2 26k DOF



(d) $\rho\tilde{\nu}$ p=2 26k DOF

FIGURE 5-8: RANS Flat Slab heat flux adapted mesh correlation

The mesh adaptation process noticeably transforms a coarse uniform background mesh into an optimized mesh that strongly resolves the boundary layer and coarsens all other regions. Since the meshes are optimized for heat flux calculations, the strongest mesh grading in the solid occurs near the leading edge where the heat transfer is greatest, and more degrees of freedom are devoted to the solid region than in the fluid free-stream. Additionally, excess refinement is observed near the edge of the boundary layer due to a noticeable jump in the heat flux adjoint from the mass equation. Interestingly enough, the adjoint jump appears to correspond with an undershoot of the SA working variable near the edge of the boundary layer. Further details are given in APPENDIX C.

5.3 Backward-Facing Step

In this section, a conjugate backward-facing step (BFS) problem is developed to demonstrate the capability of the CHT framework in computing accurate solutions for models consisting of turbulent flows with separation and recirculation regions. These solutions are useful for determining viscous forces, thermal stresses, and temperature distributions in turbulent channels with complex geometry. Many experimental and numerical studies have been performed on the single fluid discipline BFS problem for laminar and turbulent flow, and can be found in Barton [29], Armaly *et al.* [2], Thangam [67], among many more. Biswas *et al.* [13] have demonstrated the formation of Moffatt eddies [40, 54] near the inner corner of the backward facing step as the Reynolds number is lowered. Such flows can be approximated as Stokes flow, yielding eddies that circulate in counter-rotating directions and have a strength that decays exponentially as the inner corner is approached. Furthermore, NASA has published a turbulence modeling resource for the BFS problem[3], which is used as a baseline model for the conjugate model.

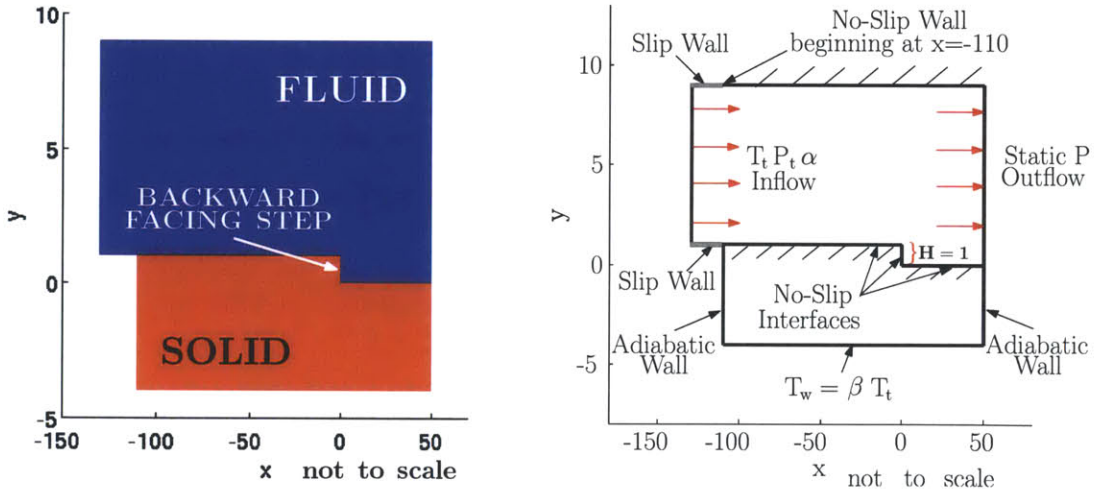
In addition to the single discipline studies, a few investigations have been carried out for the laminar conjugate BFS problem. Kanna and Das [42] studied the conjugate heat transfer for a 2D laminar wall jet over a BFS, where the effects of Reynolds number, Prandtl number, slab thickness ratio, and thermal conductivity ratio were found to be more significant near the recirculation region. Additionally, Kanna and Das [41] developed a benchmark conjugate model for the laminar BFS problem, and found that high thermal gradients were observed near the reattachment location, though the peak Nusselt number occurred further downstream. It was also found that the average Nusselt number increases with either an increased Reynolds number, Prandtl number, or thermal conductivity ratio, and that the temperature is reduced to a minimum value near the recirculation region for a cold wall solid.

[3]http://turbmodels.larc.nasa.gov/backstep_val.html

Here, a conjugate RANS BFS problem is constructed to determine the effects of varying solid wall temperatures on the overall interface temperature and heat flux distribution, especially near the recirculation region. In addition, meshes are adapted to the total heat flux on the interface in order to both achieve accurate solutions and determine areas of greatest influence and sensitivity.

5.3.1 Conjugate RANS Model

The conjugate RANS backward-facing step problem comprises of both a fluid and a solid region (FIGURE 5-9(a)) whose coupled solutions are solved simultaneously. The fluid region and conditions of the conjugate model is based on the NASA turbulence modeling resource for the BFS problem[4]. FIGURE 5-9(b) shows the sketch of the computational domain and conditions used to model the conjugate RANS backward-facing step problem.



(a) Region definition

(b) Sketch of computational domain

FIGURE 5-9: Backward-facing step conjugate flow model

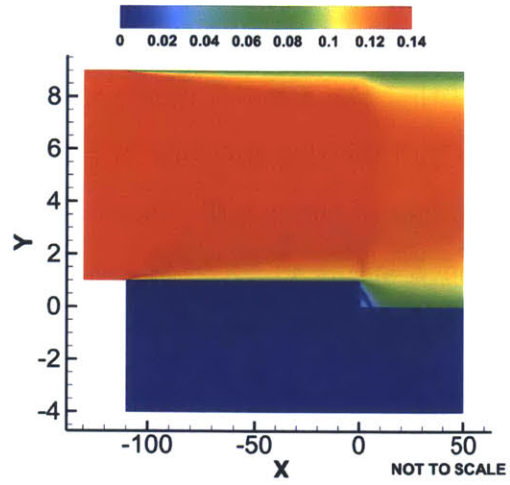
The fluid domain consists of a uniform inflow condition with a slip wall entrance on the top and bottom, a static pressure outflow, an adiabatic no-slip wall on the top,

[4]http://turbmodels.larc.nasa.gov/backstep_val.html

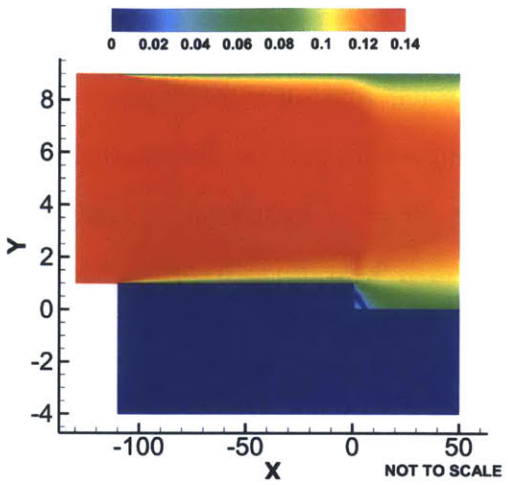
and a solid wall interface on the bottom. The inlet condition is set by a uniform flow with a non-dimensionalized total pressure and temperature calculated from a reference Mach $M_{ref} = 0.128$, and a free-stream SA working variable state is set to $\rho\tilde{\nu} = 3$. The total pressure and temperature are calculated by EQUATION 4.21 where the reference conditions are defined by the inlet conditions. Solutions to this case are obtained with a Reynolds number of $Re_H = 36,000$, set by the non-dimensionalized viscosity with reference to the inlet conditions: $\overline{\mu} = \mu/(\rho_{ref} V_{ref} H) = 1/Re_H = 2.\overline{7}\times 10^{-5}$ where $H = 1$ represents the step height. The respective fluid wall length and solid slab thickness are $110H$ and $5H$ before the step, and $50H$ and $4H$ after the step. The left and right walls are set to be adiabatic, while the bottom wall uses a Dirichlet temperature condition that sets the wall temperature T_w to a fraction β of the incoming fluid's total temperature. Additionally, the fluid viscosity varies with temperature according to Sutherland's law. This model uses a Prandtl number of $Pr = 0.72$, a specific heat ratio of $\gamma = 1.4$, and a conductivity ratio $\kappa_r = \kappa_s/\kappa_f = 1000$, which corresponds to a conductive wall. As reference, the conductivity ratio between titanium and air is $k_r \approx 1000$.

5.3.2 Numerical Solution

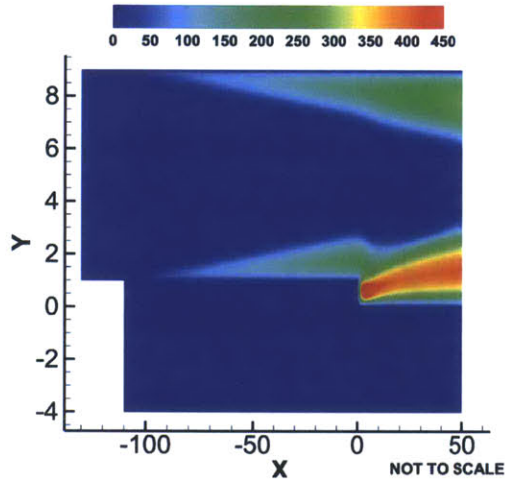
For this problem, the varying wall temperature's effects on the fluid solution is of interest. FIGURE 5-10 illustrates the Mach, SA conservative state $\rho\tilde{\nu}$, and temperature profiles of the steady conjugate solution solved on a $p = 2$ 50k DOF heat flux adapted mesh with $\beta = 0.7, 1.3$. The Mach and $\rho\tilde{\nu}$ profiles appear fairly insensitive to the range of changes in β, though differences in magnitude still occur. For $\beta > 1$, heat is transferred from the solid to the fluid, yielding higher temperatures and therefore lower densities (affecting $\rho\tilde{\nu}$) and Mach numbers in the boundary layer. Additionally, the largest changes in the fluid temperature are observed near the leading edge of the interface, as well as near the backward-facing step.



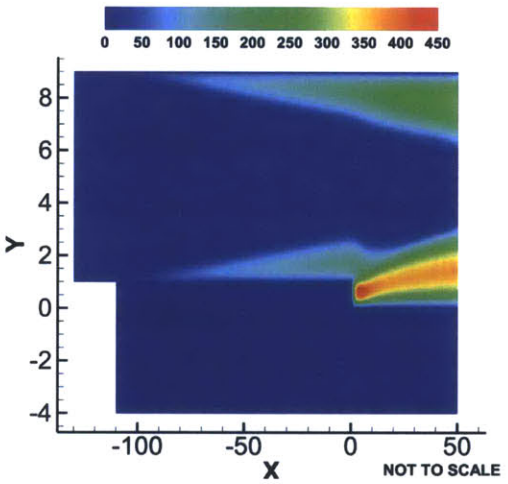
(a) Mach profile $\beta = 0.7$



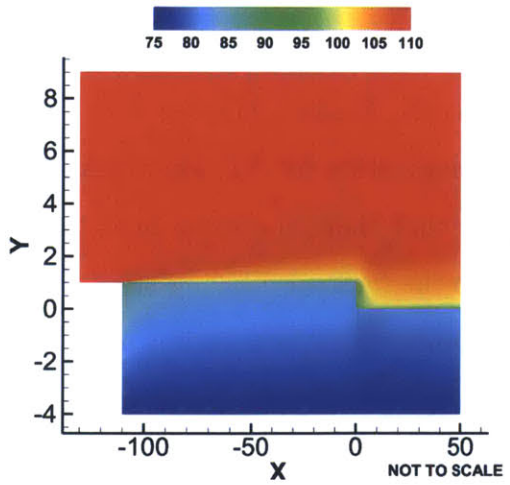
(b) Mach profile $\beta = 1.3$



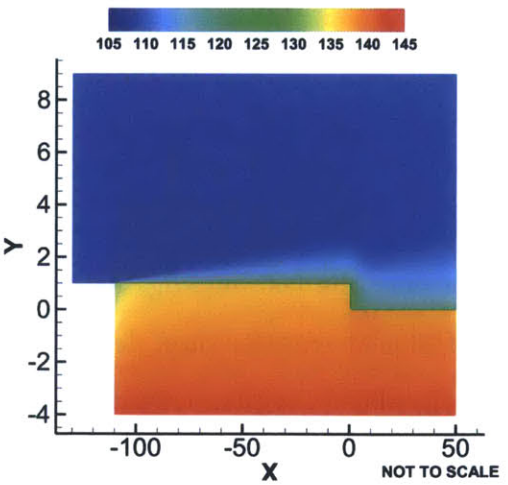
(c) $\rho\tilde{\nu}$ profile $\beta = 0.7$



(d) $\rho\tilde{\nu}$ profile $\beta = 1.3$



(e) Fluid temperature profile $\beta = 0.7$



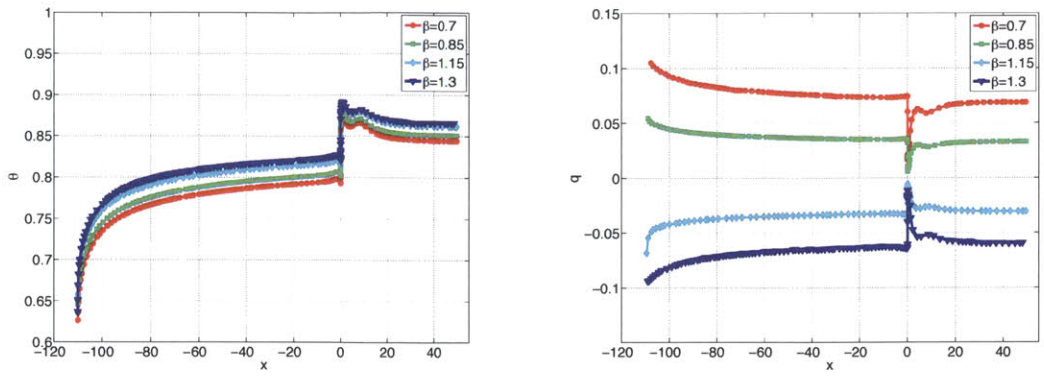
(f) Fluid temperature profile $\beta = 1.3$

FIGURE 5-10: Numerical solution to BFS p=2 50k DOF

Similar to the flat slab case, the amount of heat flux through the interface is characterized by calculating the Stanton number. Recall that the Stanton number can be approximated though the Reynolds analogy as one half the skin friction: $St = C_f/2$, assuming $Pr = 1$. FIGURE 5-11 shows interface profiles of the non-dimensional temperature (θ), heat flux (q), skin friction (C_f), and Stanton number (St) along the interface. A positive heat flux q is defined as heat transfer from the fluid to the solid domain. The local temperature and Stanton number are functions of x, and are given by:

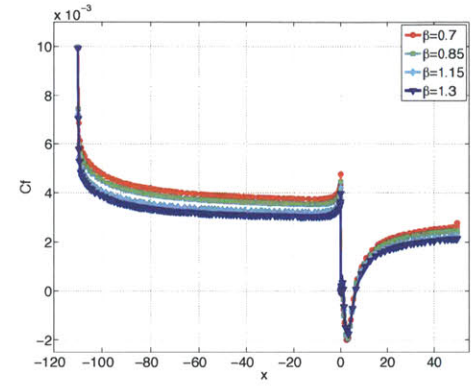
$$\theta(x) = \frac{T(x) - T_\infty}{T_w - T_\infty} \qquad St(x) = -\frac{\kappa_f \left.\frac{\partial T}{\partial y}\right|_{y=0}}{(\rho u)_e c_p (T(x) - T_e)} \tag{5.5}$$

where T_w and T_∞ are the solid lower wall and fluid free stream temperature, respectively, κ_f is the fluid's thermal conductivity, and all states $(\cdot)_e$ are evaluated at the edge of the boundary layer as a function of x. In calculating the Stanton number, the heat flux at the interface was determined by extracting the total energy viscous flux on the interface, and the boundary layer edge profiles were extracted from data along a line just outside the boundary layer. Again, a moving-average method was used to smooth the heat flux profile. From this, the Stanton number appears to follow the same trend as the skin friction, only deviating within the recirculation zone where the skin friction reverses sign while the direction of heat transfer does not. Though the heat flux is greatest for $\beta = 0.7$ and $\beta = 1.3$, the Stanton number for $\beta = 1.3$ is the lowest (also reflected in C_f) due to the normalization by ΔT where $(\Delta T)_{\beta=1.3}$ is largest. Additionally, the Stanton profile shows that there is greater heat transfer occurring near the leading edge as oppose to near the backward-facing step. This is because the heat transfer rate, dependent on the convective flow speed, is much lower in the recirculation region, despite the greater change in temperature.

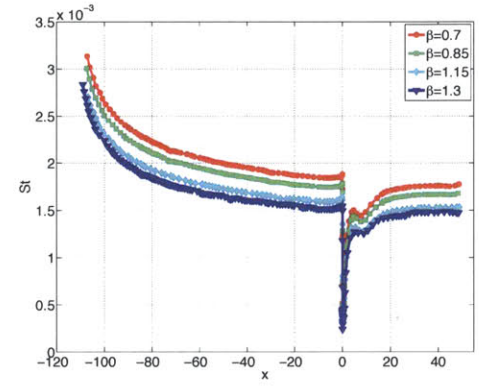


(a) Interface Temperature

(b) Interface Heat Flux (into solid)



(c) Interface Skin Friction

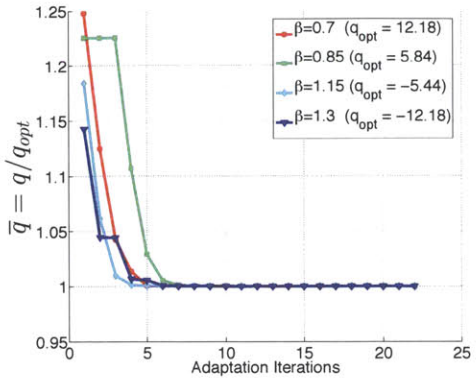


(d) Interface Stanton Number

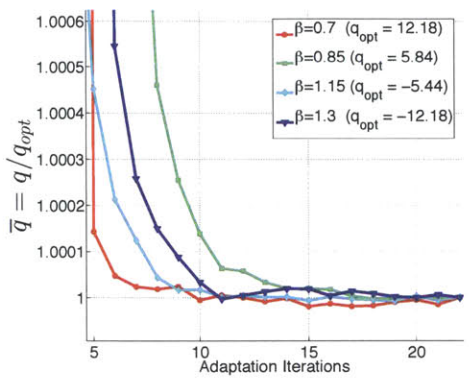
FIGURE 5-11: Interface profiles for BFS p=2 50k DOF

5.3.3 Optimized Meshes

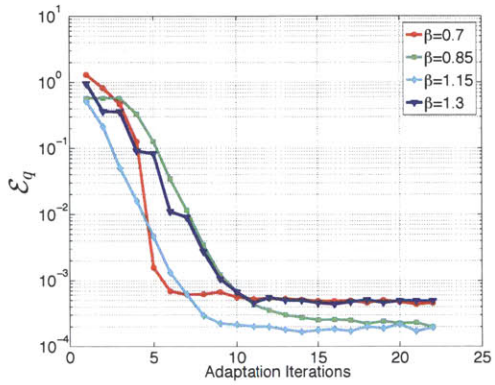
For each of the BFS slab wall temperature conditions, β, mesh adaptation was performed in order to refine the mesh with respect to the total interface heat flux. For each case, several adaptation iterations are computed until the error estimate of the output of interest converges. FIGURE 5-12 shows the heat flux ($\overline{q}$) and heat flux error estimate ($\mathcal{E}_q$) adaptation history for a range of β's. For convergence comparison, the heat flux is normalized by the optimized, or final mesh output. Each of the cases converge in approximately 15 adaptation iterations, and the converged error estimate for $\beta = 0.7$ and 1.3 is highest, indicating a greater uncertainty in cases with higher heat transfer rates.



(a) Normalized heat flux convergence



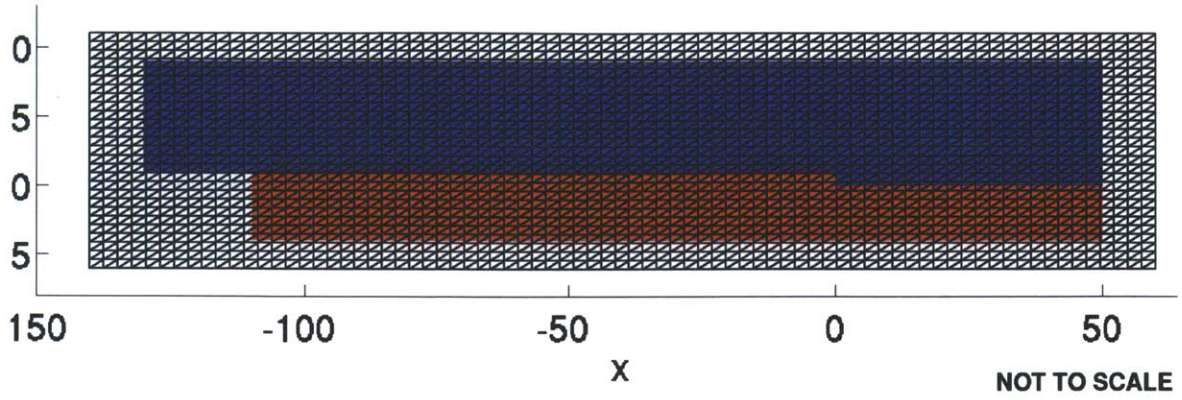
(b) Normalized heat flux convergence zoom



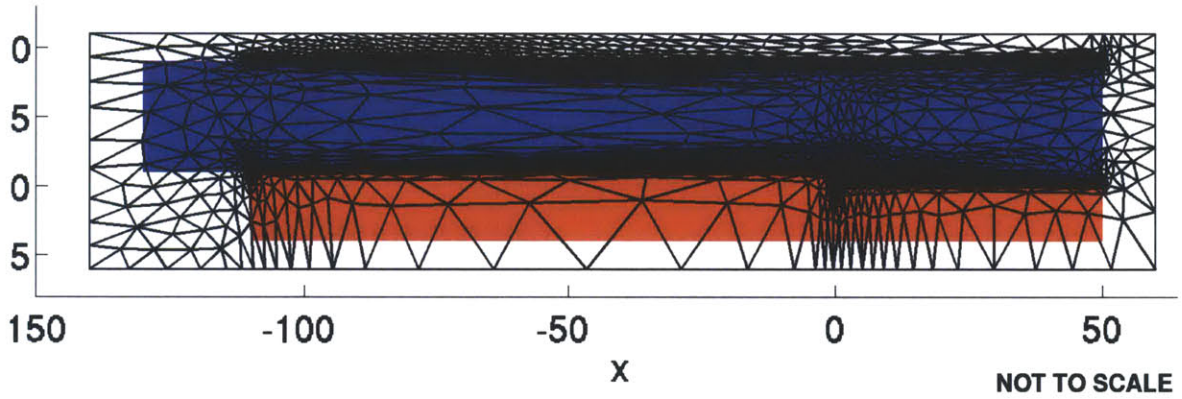
(c) Heat flux error convergence

FIGURE 5-12: BFS heat flux adaptation history over range of β's

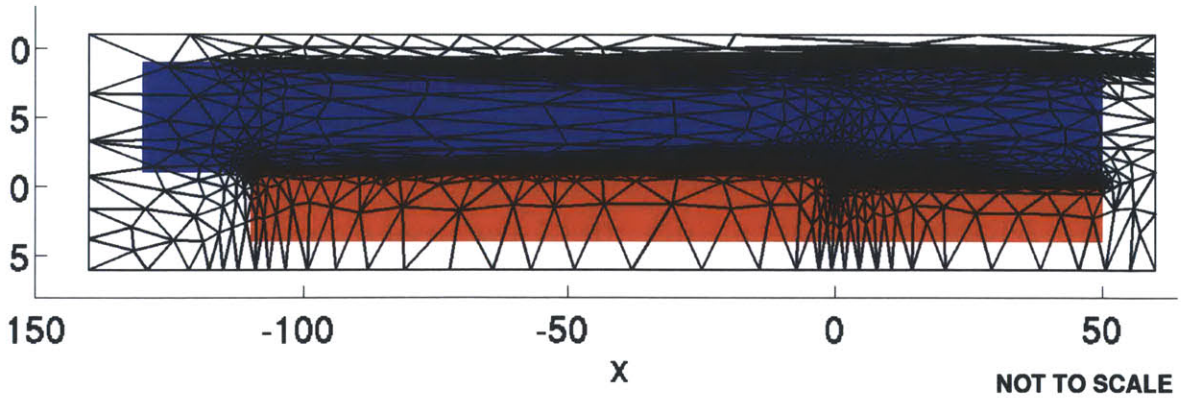
FIGURE 5-13 illustrates the scaled domain geometry overlaid by the interface heat flux adapted meshes for $p = 2$ solutions with 50k DOF and $\beta = 0.7, 1.3$. The optimized meshes are found to be generally insensitive to the changes made to the solid wall temperature, though small changes are observed near the backward-facing step due to the change in size of the recirculation zone for different β's (See SECTION 5.3.4). This effect is seen more clearly in FIGURE 5-14, where the mesh refinement along a curve within the recirculation zone visibly shifts downstream with increasing β.



(a) BFS initial background mesh

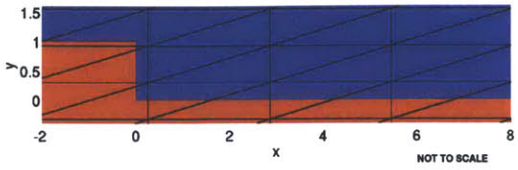


(b) Heat adapt p=2 50k ($\beta = 0.7$)

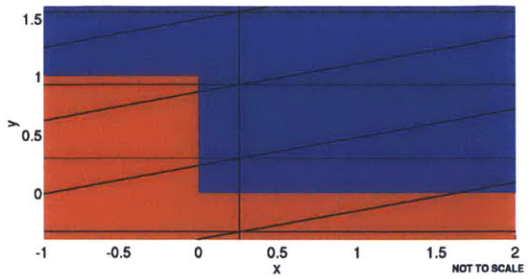


(c) Heat adapt p=2 50k ($\beta = 1.3$)

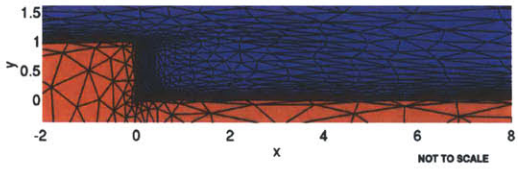
FIGURE 5-13: BFS heat flux adapted meshes



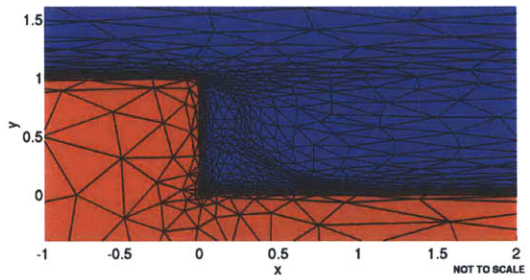
(a) BFS initial background mesh zoom



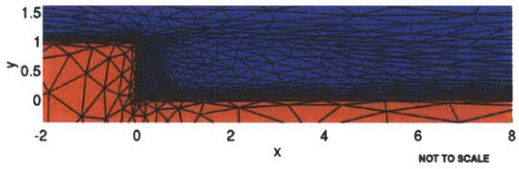
(b) BFS initial background mesh corner



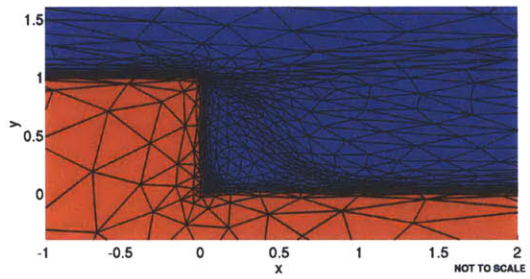
(c) Heat adapt p=2 50k zoom ($\beta = 0.7$)



(d) Heat adapt p=2 50k corner ($\beta = 0.7$)



(e) Heat adapt p=2 50k zoom ($\beta = 1.3$)



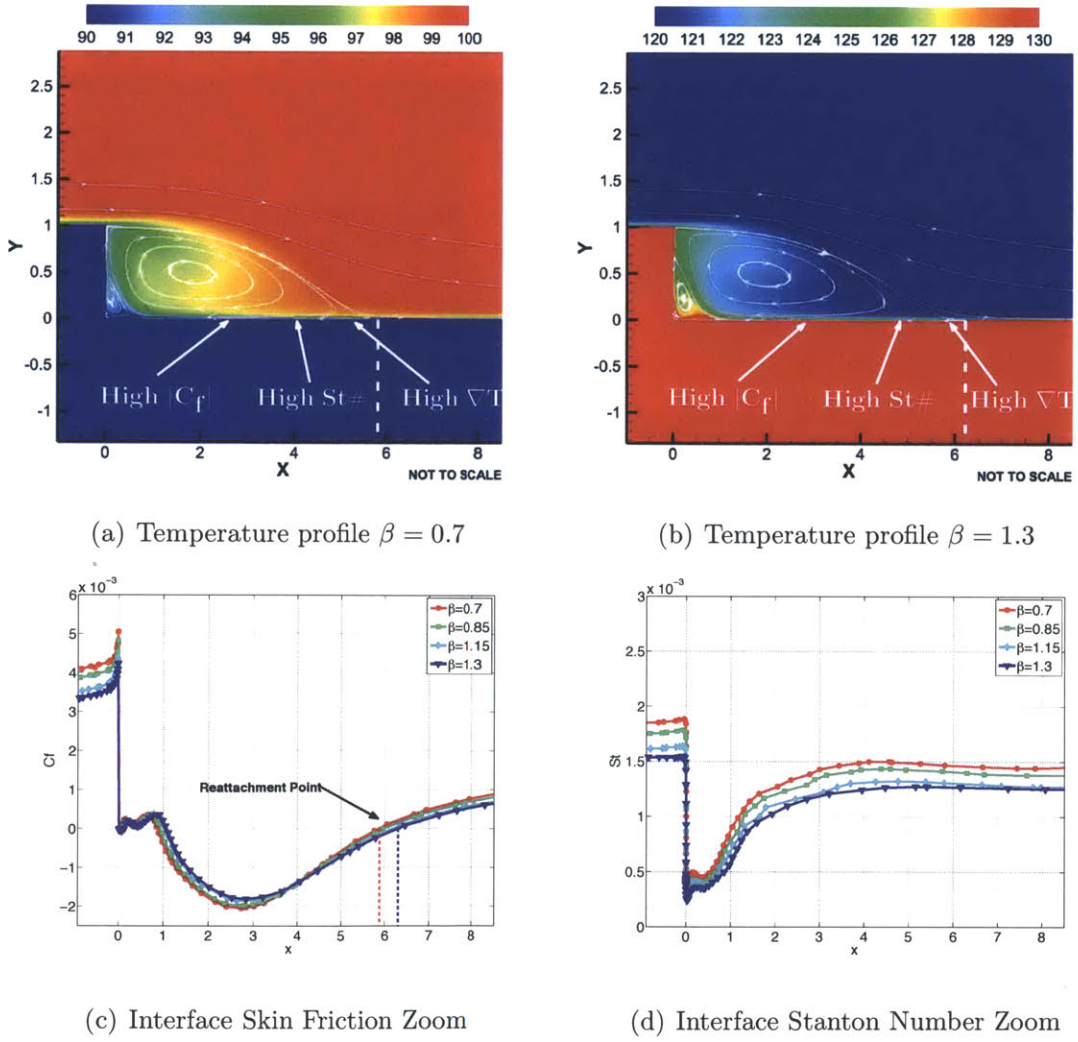
(f) Heat adapt p=2 50k corner ($\beta = 1.3$)

FIGURE 5-14: BFS heat flux adapted meshes zoom

5.3.4 Moffatt vortices and effect on heat transfer

Looking more closely at the backward-facing step, the presence of multiple counter-rotating vortices are discovered. These vortices, known as Moffatt vortices, have a strength and size that decays exponentially as the inner corner is approached. FIGURE 5-15 illustrates the vortices by overlaying Mach profiles near the backward-facing step with flow streamlines for $\beta = 0.7$ and 1.3. The corresponding skin friction and Stanton distribution are also provided in FIGURE 5-15. From these figures, it

appears that the size of the 'recirculation bubbles' grows with increasing β, and that the reattachment point moves farther downstream. The streamline paths are used to find the location of the core flow reattachment point, which is verified with a zero skin friction. Within the recirculation bubble between $x = 2$ and $x = 3$, a minimum skin friction is observed due to a large negative flow velocity; however, a local maximum in the Stanton number occurs farther downstream around $x = [4, 5]$. This is caused by the competing effects of a large convective heat transfer coefficient near the region of high skin friction and a high temperature gradient near the reattachment point as illustrated in FIGURE 5-15.



(a) Temperature profile $\beta = 0.7$

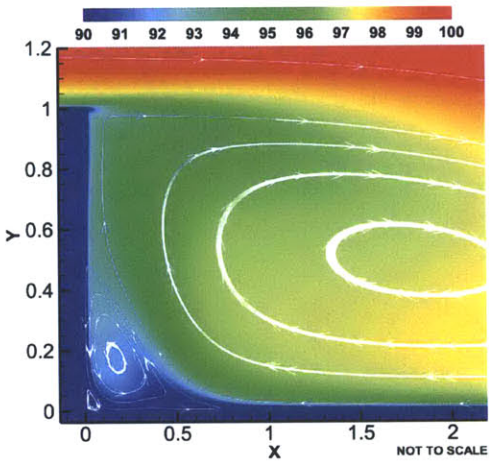
(b) Temperature profile $\beta = 1.3$

(c) Interface Skin Friction Zoom

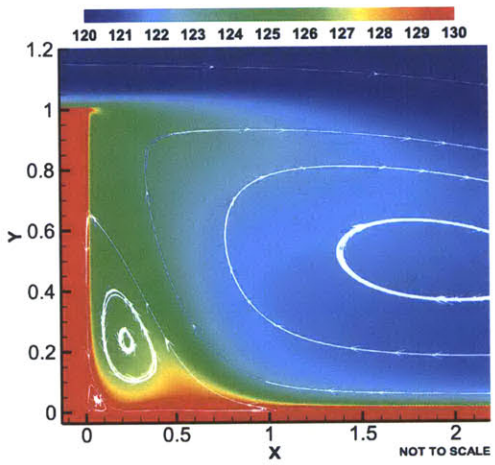
(d) Interface Stanton Number Zoom

FIGURE 5-15: Numerical solution to BFS p=2 50k DOF zoom

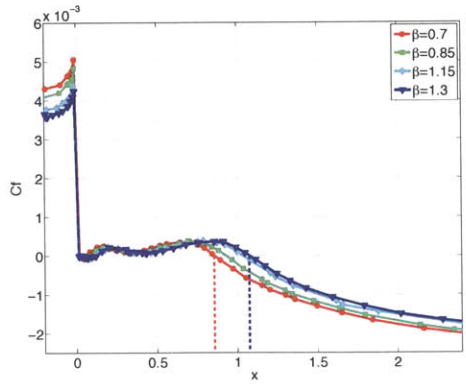
Looking even closer at the corner, the formation of 3 independent counter-rotating vortices is clearly visible. FIGURE 5-16 illustrates these vortices via streamlines on top of the temperature profile near the corner of the backward-facing step, as well as the corresponding skin friction and Stanton profiles. Again, the location of the stagnation point between the primary and secondary vortex is verified by observing where the skin friction is zero. A stagnation point also occurs between the secondary and tertiary vortex at $x = 0.1$, though only the location of the former stagnation point is pointed out on the skin friction plot. Just as before, the Stanton number exhibits another local maximum that occurs as a compromise between the local heat flux coefficient and temperature gradient.



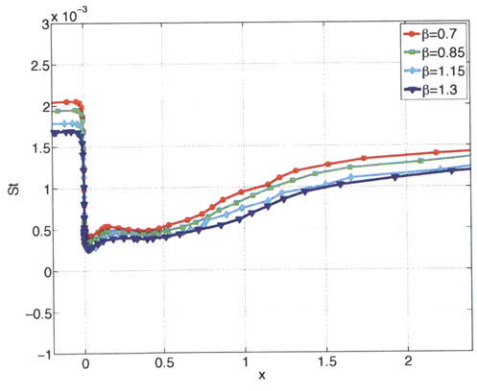
(a) Temperature profile $\beta = 0.7$



(b) Temperature profile $\beta = 1.3$



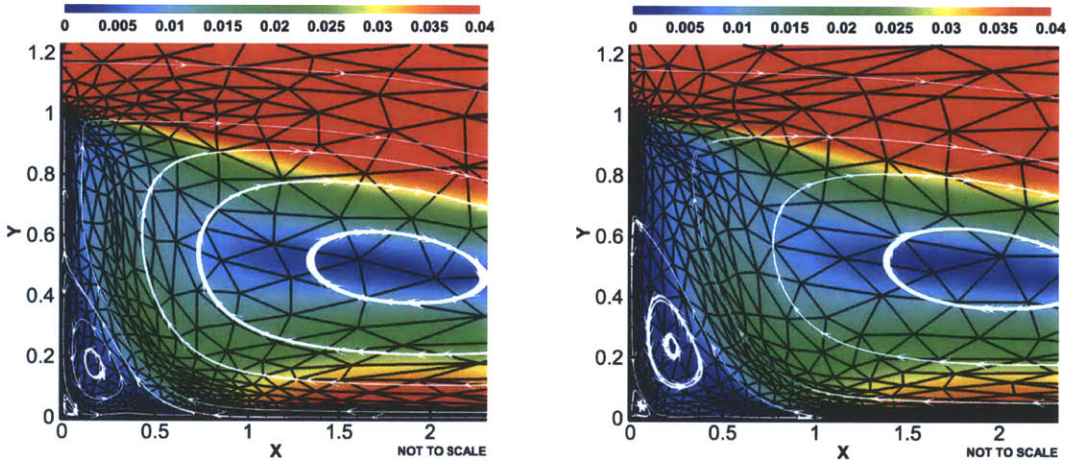
(c) Interface Skin Friction Corner Zoom



(d) Interface Stanton Number Corner Zoom

FIGURE 5-16: Moffatt Vortices p=2 50k DOF

Furthermore, the curved feature that has been refined by mesh adaptation is discovered to correspond to an 'updraft' from the primary recirculation bubble. FIGURE 5-17 shows the Mach distribution near the backward-facing step corner and displays the primary recirculation bubble's updraft matching with the refinement of the overlaid optimized mesh for $\beta = 0.7$ and 1.3. Though the heat transfer in the recirculation zone is not the highest in the overall flow, the local influence of the updraft is large enough to warrant mesh refinement. Moreover, a downstream shift in the refinement curve, corresponding to a 'growth' in the recirculation bubble, is apparent when increasing β. From the feature's refinement and movement, it is deduced that the updraft of the primary recirculation bubble has significance in determining both the downstream and local heat flux near the backward-facing step.



(a) Mach profile and mesh for $\beta = 0.7$

(b) Mach profile and mesh for $\beta = 1.3$

FIGURE 5-17: Mesh refinement of 'updraft' in BFS recirculation bubble (p=2 50k DOF)

As a final point, the refined features in the mesh have been utilized to identify regions where the physical flow has significant influence on the output of interest. Though experienced users are able to identify general regions that require refinement, distribution of degrees of freedom in an efficient manner is not trivial, especially when features within the optimal mesh transforms under different conditions. With the addition of geometric or physical complexity, mesh adaptation becomes an essential

tool to not only accurately predict outputs of interest, but also to gain insight into non-intuitive physical phenomena, as demonstrated.

Chapter 6

Conclusion

6.1 Summary and Conclusions

This thesis presents work toward developing a robust, PDE solution framework for CHT simulations that autonomously provides reliable output predictions. More specifically, the framework is comprised of a simplex cut-cell technique, a high-order DG discretization, and an output-based adaptation method. The cut-cell method is an extension of the method developed by Sun [71] to handle multi-regioned embedded domains, while the mesh adaptation algorithm, based on the work of Yano [77] with modifications by Kudo [39], was applied to conjugate heat transfer problems.

The cut-cell method was expanded to incorporate multi-regioned geometries, and was demonstrated for a variety of conjugate heat transfer models consisting of multiple regions and curved interfaces. The new cut-cell method was also applied to an r^{α}-type corner singularity problem to demonstrate the effective ability of cut cells with adaptation to achieve a mesh that conforms to a desired metric field. Additionally, the combination of the cut-cell method with mesh adaptation allowed for a more efficient and streamlined approach to solving multi-physics problems.

For Navier-Stokes CHT problems, the DG discretization applied to coupled regions of different physics yielded $p+1$ optimum convergence rates in each of the conservative

state's L_2 error. Mesh adaptation with respect to the interface heat flux was found to be important in achieving accurate predictions of the heat flux across the interface. For the cooled nozzle case, two noticeably different optimized meshes were generated by the mesh adaptation algorithm, depending on the output of interest. Near super-convergent rates for both drag and heat flux were observed when adapting to heat flux, though the accuracy in the heat flux output degraded when adapting to drag alone. The finding suggested that heat flux adaptation performs better than drag adaptation when computing a *combined* prediction in both drag and heat flux, whereas drag adaptation is necessary if drag is the only output of interest. Despite the output error convergence appearing to achieve near super-convergent rates, it is still recommended that additional analysis on the current interface boundary condition is performed to determine adjoint consistency and other discretization properties. Lastly, both the secondary flow mass flux and interface heat flux in the multi-flow problem converged with near super-convergent rates and resulted in similar optimized meshes. This implies that an accurate computation of the mass flux relies on resolving similar domain features that are equally important in computing the interface heat flux. Nevertheless, adapting to the secondary flow mass flux appeared to perform better in achieving accurate predictions in *both* the mass flux and heat flux, whereas heat flux adaptation is marginally favorable when interested in heat flux alone.

The CHT framework was applied to coupled RANS discretizations. The interface discretization, based on the work of Sun [71], was implemented for a flat slab and backward-facing step conjugate problem. The mesh adaptation process is capable of robustly and autonomously transforming a coarse uniform background mesh into an optimized mesh with highly resolved domain features pertinent to achieving accurate predictions in heat transfer. With the help of mesh adaptation, non-trivial features, such as Moffatt vortices in the backward-facing step problem, were identified. Finally, optimized meshes allowed for greater physical insight to the coupled interaction at interfaces.

6.2 Future Work

Listed here are several areas identified for future work.

Interface boundary analysis

The interface discretization and conditions defined in this work were based on the work of Sun [71], who had proved dual consistency for *scalar* interface problems. Though near super-convergent rates were observed in the Navier-Stokes CHT problems, a dual consistency analysis is still required. Additionally, further analysis is necessary to determine consistency in the properties of the interface discretization with respect to the boundary and interior discretizations.

Three-dimensional conjugate heat transfer problems

While only two-dimensional problems were developed in this work, the modifications made to the cut-cell intersection method to handle multi-regioned domains can be extended to three-dimensional cut-cell algorithms. Furthermore, along with additional analysis, the DG discretization for interface problems is already suitable for higher dimensions. Lastly, efficient quadrature rules, and merging techniques are already developed for three-dimensional cut-cell problems [71], and are also applicable to multi-regioned interface problems.

General multi-disciplinary simulation

The CHT framework presented here was developed to provide a modular approach to forming multi-disciplinary simulations. For multi-disciplinary applications where the number of conservative states between regions are different, the cost model constraint in the mesh adaptation method can be adjusted to reflect the differences in degrees of freedom. Additionally, extensions can be made to simulate coupled regions governed by different PDE's, such as aeroelastic (fluid-solid) or combustion (fluid-fluid) simulation. However, this would require the capability to handle moving boundaries or interfaces between domains, which can be achieved with cut-cell methods. Though

cut-cell methods can autonomously generate non-conforming meshes with respect to moving geometries, they would require modifications for increased efficiency. Unsteady adjoint analysis and output-based adaptation would also require additional development.

Appendix A

Governing Equations

In this work, three primary governing equations are considered: the heat equation, the Compressible Navier-Stokes equations, and the Reynolds-Averaged Navier-Stokes equations. These equations can all be written in the general conservation form:

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathcal{F}^i(\mathbf{u}, \mathbf{x}, t) - \nabla \cdot \mathcal{F}^v(\mathbf{u}, \nabla \mathbf{u}, \mathbf{x}, t) = \mathcal{S}(\mathbf{u}, \nabla \mathbf{u}, \mathbf{x}, t), \quad \forall \mathbf{x} \in \Omega, \ t \in I \quad \text{(A.1)}$$

where $\mathbf{u}(\mathbf{x}, t) : \mathbb{R}^{m_r}$ is the m_r-state solution vector in region r, $\mathcal{F}^i(\mathbf{u}, \mathbf{x}, t) : \mathbb{R}^{m_r \times d}$ is the inviscid flux, $\mathcal{F}^v(\mathbf{u}, \nabla \mathbf{u}, \mathbf{x}, t) : \mathbb{R}^{m_r \times d}$ is the viscous flux, and $\mathcal{S}(\mathbf{u}, \nabla \mathbf{u}, \mathbf{x}, t) : \mathbb{R}^{m_r}$ is the source term. Note that the state rank and residual term definitions are region dependent.

A.1 Heat Equation

The steady state heat equation is given by:

$$-\nabla \cdot (\kappa_s \nabla T) = 0 \quad \text{(A.2)}$$

where κ_s is the solid's thermal conductivity and T is the temperature. The solution to this equation has a state rank $m = 1$ with temperature being the conservative state. In terms of the general conservative form given by EQUATION A.1, the inviscid

flux, $\mathcal{F}^i$, is zero, and the viscous flux $\mathcal{F}^v = (\mathbf{F}_x^v, \mathbf{F}_y^v)$ is given by:

$$\mathbf{F}_x^v = \kappa_s \frac{\partial T}{\partial x} \qquad \mathbf{F}_y^v = \kappa_s \frac{\partial T}{\partial y} \tag{A.3}$$

Additionally, the source term, $\mathcal{S}$, is also set to zero, unless otherwise stated.

A.2 Compressible Navier-Stokes Equations

The conservative state vector used for the 2-D compressible Navier-Stokes equations is $u = [\rho, \rho u, \rho v, \rho E]^T$, where ρ is the density, u and v are the respective x and y-directional velocities, and E is the specific total internal energy. For a Newtonian fluid, the sheer stress τ is given by:

$$\tau_{ij} = \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) + \delta_{ij} \lambda \frac{\partial v_k}{\partial x_k} \tag{A.4}$$

where μ is the dynamic viscosity, and $\lambda = -2/3\mu$ is the bulk viscosity coefficient. Under this assumption, the inviscid flux vector $\mathcal{F}^i = (\mathbf{F}_x^i, \mathbf{F}_y^i)$ is:

$$\mathbf{F}_x^i = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{pmatrix} \quad , \quad \mathbf{F}_y^i = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \end{pmatrix} \tag{A.5}$$

and the viscous flux vector $\mathcal{F}^v = (\mathbf{F}_x^v, \mathbf{F}_y^v)$ is:

$$\mathbf{F}_x^v = \begin{pmatrix} 0 \\ \frac{2}{3}\mu(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}) \\ \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \\ \frac{2}{3}\mu(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y})u + \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})v + \kappa_f \frac{\partial T}{\partial x} \end{pmatrix} \tag{A.6}$$

$$\mathbf{F}_y^v = \begin{pmatrix} 0 \\ \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \\ \frac{2}{3}\mu(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}) \\ \frac{2}{3}\mu(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x})v + \mu(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})u + \kappa_f \frac{\partial T}{\partial y} \end{pmatrix} \tag{A.7}$$

where p is the static pressure, $H = E + p/\rho$ is the specific total enthalpy, T is the temperature, and κ_f is the thermal conductivity. The state equation in terms of the conservative variables is:

$$p = (\gamma - 1)\rho\left(E - \frac{1}{2}(u^2 + v^2)\right) \tag{A.8}$$

while the dynamic viscosity is modeled by Sutherland's law:

$$\mu = \mu_{ref}\left(\frac{T}{T_{ref}}\right)^{1.5}\frac{T_{ref} + T_s}{T + T_s} \tag{A.9}$$

unless otherwise stated. With a specific heat at constant pressure $c_p = \gamma R/(\gamma - 1)$ and a specified Prandtl number Pr, the thermal conductivity, κ_f, is determined by:

$$\kappa_f = c_p\frac{\mu}{Pr}. \tag{A.10}$$

A.3 Reynolds-Averaged Navier-Stokes Equations

The Reynolds-Averaged Navier-Stokes Equations (RANS) are derived by time-averaging the Navier-Stokes equations after applying Reynolds decomposition. The form of the RANS equations is given by:

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial}{\partial x_i}(\bar{\rho}\tilde{u}_i) = 0,$$
$$\frac{\partial}{\partial t}(\bar{\rho}\tilde{u}_i) + \frac{\partial}{\partial x_j}(\bar{\rho}\tilde{u}_j\tilde{u}_i + \bar{p}\delta_{ji}) = \frac{\partial}{\partial x_j}\left[2(\mu + \mu_t)\left(\tilde{s}_{ji} - \frac{1}{3}\frac{\partial \tilde{u}_k}{\partial x_k}\delta_{ji}\right)\right],$$

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

where ρ is the density, u_i is the i-directional velocity, p is the pressure, e is the 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 $\bar{(\cdot)}$ notation indicates Reynolds-averaging, while the $\tilde{(\cdot)}$ denotes Favre-averaging.

The RANS equations carry an unknown variable (eddy viscosity μ_t), which requires an additional equation for closure.

A.3.1 The SA Turbulence Model

The closure of the RANS equations can be achieved by introducing a turbulence model that governs the transport of the eddy viscosity μ_t. In this work, the SA turbulence model [70] is chosen to close the RANS equations since it has been shown to be accurate for most attached and partially separated aerodynamic flows [35, 27, 17]. Here, the implemented SA turbulence model along with the negative model variations [57, 1] is presented.

The conservative state vector for the 2-D RANS-SA equations is $u=[\rho,\rho u,\rho v,\rho E,\rho\tilde{\nu}]^T$ where $\tilde{\nu}$ is the working variable for the SA model. The working variable $\tilde{\nu}$ is related to the eddy viscosity μ_t by:

$$\mu_t=\begin{cases}\rho\tilde{\nu}f_{v1}, & \tilde{\nu}\geq 0\\ 0, & \tilde{\nu}<0\end{cases} \tag{A.11}$$

where

$$f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad \chi = \frac{\tilde{\nu}}{\nu} \tag{A.12}$$

where $\nu = \mu/\rho$ is the kinematic viscosity. The inviscid flux vector $\mathcal{F}_i = (\mathbf{F}_x^i, \mathbf{F}_y^i)$ for 2-D RANS-SA is given by:

$$\mathbf{F}_x^i = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \\ \rho \tilde{\nu} u \end{pmatrix} \quad , \quad \mathbf{F}_y^i = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vH \\ \rho \tilde{\nu} v \end{pmatrix} \tag{A.13}$$

and the viscous flux vector $\mathcal{F}^v = (\mathbf{F}_x^v, \mathbf{F}_y^v)$ is:

$$\mathbf{F}_x^v = \begin{pmatrix} 0 \\ \frac{2}{3}\mu'(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}) \\ \mu'(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \\ \frac{2}{3}\mu'(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y})u + \mu'(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})v + \kappa_f' \frac{\partial T}{\partial x} \\ \frac{1}{\sigma}\eta \frac{\partial \tilde{\nu}}{\partial x} \end{pmatrix} \tag{A.14}$$

$$\mathbf{F}_y^v = \begin{pmatrix} 0 \\ \mu'(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}) \\ \frac{2}{3}\mu'(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}) \\ \frac{2}{3}\mu'(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x})v + \mu'(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x})u + \kappa_f' \frac{\partial T}{\partial y} \\ \frac{1}{\sigma}\eta \frac{\partial \tilde{\nu}}{\partial y} \end{pmatrix} \tag{A.15}$$

where

$$\mu' = (\mu + \mu_t) \qquad \kappa_f' = c_p \left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t} \right) \tag{A.16}$$

and the SA diffusion coefficient η is:

$$\eta = \mu(1 + \chi f_n) \tag{A.17}$$

where

$$f_n = \begin{cases} 1.0, & \tilde{\nu} > 0 \\ \dfrac{c_{n1} + \chi^3}{c_{n1} - \chi^3}, & \tilde{\nu} \leq 0 \end{cases} \tag{A.18}$$

The RANS-SA system also incorporates a source term of the form:

$$\mathcal{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ P - D + \dfrac{c_{b2}\rho}{\sigma}\dfrac{\partial\tilde{\nu}}{\partial x_k}\dfrac{\partial\tilde{\nu}}{\partial x_k} + \dfrac{\nu + \tilde{\nu}}{\sigma}\dfrac{\partial\rho}{\partial x_k}\dfrac{\partial\tilde{\nu}}{\partial x_k} \end{bmatrix} \tag{A.19}$$

The production term, P, is given by:

$$P = \begin{cases} c_{b1}(1 - f_{t2})\rho\tilde{S}\tilde{\nu}, & \tilde{\nu} > 0 \\ c_{b1}(1 - c_{t3})\rho S\tilde{\nu}, & \tilde{\nu} \leq 0 \end{cases} \tag{A.20}$$

where the laminar suppression term f_{t2} is:

$$f_{t2} = c_{t3}\exp(-c_{t4}\chi^2) \tag{A.21}$$

and

$$\tilde{S} = \begin{cases} S + \overline{S}, & \overline{S} \geq -c_{v2}S \\ S + \dfrac{S(c_{v2}^2 S + c_{v3}\overline{S})}{(c_{v3} - 2c_{v2})S - \overline{S}}, & \overline{S} < -c_{v2}S \end{cases} \tag{A.22}$$

where $S = \sqrt{2\Omega_{ij}\Omega_{ij}}$ is the magnitude of the vorticity, $\Omega_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i}\right)$, and the near-wall correction term is:

$$\overline{S} = \frac{\tilde{\nu} f_{v2}}{\kappa^2 d^2} \quad \text{with} \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \tag{A.23}$$

where d is the wall-distance.

The destruction term, D, is given by:

$$D = \begin{cases} \left(c_{w1} f_w - \dfrac{c_{b1}}{\kappa^2} f_{t2} \right) \dfrac{\rho \tilde{\nu}^2}{d^2}, & \tilde{\nu} > 0 \\ -c_{w1} \dfrac{\rho \tilde{\nu}^2}{d^2}, & \tilde{\nu} \leq 0 \end{cases} \tag{A.24}$$

where

$$f_w = g \left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6} \right)^{1/6}, \quad g = r + c_{w2}(r^6 - r), \quad \text{and} \quad r = \frac{\tilde{\nu}}{\tilde{S} \kappa^2 d^2} \tag{A.25}$$

The turbulence model constants are set to: $c_{b1} = 0.1355$, $\sigma = 2/3$, $c_{b2} = 0.622$, $\kappa = 0.41$, $c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma$, $c_{w2} = 0.3$, $c_{w3} = 2$, $c_{v1} = 7.1$, $c_{v2} = 0.7$, $c_{v3} = 0.9$, $c_{t3} = 1.2$, $c_{t4} = 0.5$, $c_{n1} = 16$, and $Pr_t = 0.9$.

Note that the SA trip terms are not included in this work and therefore do not appear in the source term.

Appendix B

Derivation of Manufactured Solution to Compressible Poiseuille Flow

The derivation of a quasi-1D compressible Poiseuille flow solution starts with the assumption of a fully developed flow. For a variable viscosity and thermal conductivity, an additional coordinate transformation that is scaled by the solution temperature, as shown by Myong [53], is required. Here, the analytic solution is derived and used for convergence study purposes.

B.1 Fully Developed Flow Assumption

In a fully developed flow (both hydraulically and thermally), all conservative variables are steady ($\mathbf{u}_t = 0$) and the pressure gradient $\frac{\partial P}{\partial x}$ is non-zero. In order to find an analytic solution, a driving body force is used to balance the viscous force $-\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right)$ as oppose to the pressure gradient, and a constant pressure is assumed:

$$
\begin{aligned}
mass: \quad & 0 && = 0 \\
x-mom: \quad & -\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) && = \rho f \\
y-mom: \quad & 0 && = 0 \\
energy: \quad & -\frac{\partial}{\partial y}\left(\mu u\frac{\partial u}{\partial y}+\kappa_f\frac{\partial T}{\partial y}\right) && = \rho u f
\end{aligned}
$$

where f is a constant. This body force varies linearly with density and performs work on the flow, warranting a source term in the energy equation.

B.2 Variable Viscosity and Thermal Conductivity

With a temperature dependent viscosity, the x-momentum equation cannot be solved independent of the energy equation. However, the coupled system can be analytically solved if the viscosity and thermal conductivity is assumed to vary with temperature according to Maxwell's molecular theory with reference to the interface temperature T_i:

$$\mu(T) = \mu_0 T/T_i \tag{B.1}$$

$$\kappa_f(T) = \kappa_0 T/T_i, \tag{B.2}$$

B.3 Non-Dimensionalization

The first step in performing the derivation is to non-dimensionalize the equations, followed by performing a coordinate transformation as shown by Myong [53]. Here, the following dimensionless variables, as inspired by Myong [53], are used:

$$y^* = y/h, \qquad u^* = u/u_a, \qquad T^* = T/T_a$$

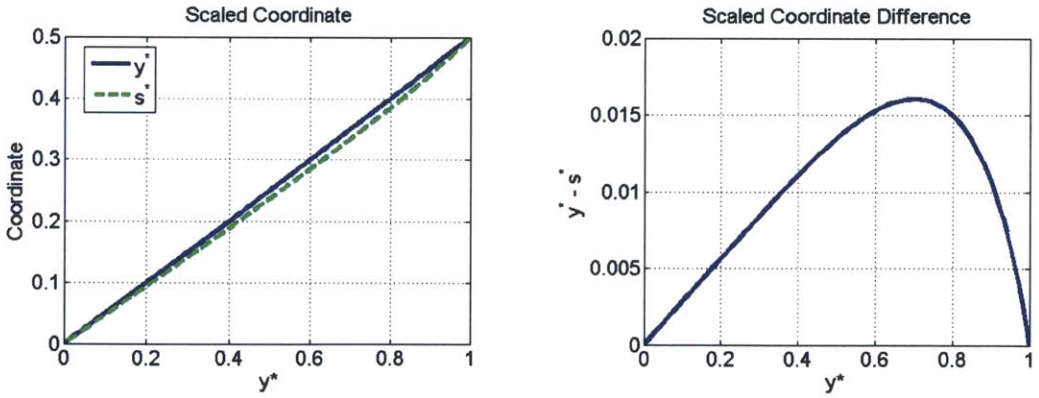
$$\epsilon = \frac{fh}{RT_a}, \qquad \delta = \frac{\mu_0 u_a}{P_c h} \qquad Pr = \frac{C_p \mu_0}{\kappa_0} \qquad M = \frac{u_a}{\sqrt{\gamma R T_i}}$$

where T_i is a prescribed interface temperature, u_a is a reference velocity, and T_a is a reference temperature.

Next, a new coordinate variable s is introduced by:

$$Tds = dy \quad , \quad T^* ds^* = dy^* \tag{B.3}$$

FIGURE B-1 shows the deviation between the newly transformed coordinate s^* and the physical coordinate y^*.



(a) Coordinate Comparison

(b) Coordinate Deviation

FIGURE B-1: Transformed Coordinates

Transforming the governing equations with this new non-dimensional variable allows for an analytical solution to be derived.

B.4 Fluid Solution

Velocity Profile

To start, The x-momentum equation is used to determine the fluid velocity profile.

$$
\begin{aligned}
f\rho &= -\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right) \\
f\rho &= -\frac{1}{h}\frac{\partial}{\partial y^*}\left(\frac{\mu_w u_a}{T_i h}T\frac{\partial u^*}{\partial y^*}\right) \\
f\rho &= -\frac{\mu_w u_a T_a^2}{h^2 T_i T}\frac{\partial^2 u^*}{\partial s^{*2}} \\
\frac{\partial^2 u^*}{\partial s^{*2}} &= -\frac{f\rho h^2 T_i}{\mu_w u_a T_a^2}\frac{P}{\rho R} \\
\frac{\partial^2 u^*}{\partial s^{*2}} &= -\frac{T_i\epsilon}{T_a\delta} \\
\frac{\partial u^*}{\partial s^*} &= -\frac{T_i\epsilon}{T_a\delta}s^* + C_1 \\
u^* &= -\frac{T_i\epsilon}{T_a\delta}s^{*2} + C_1 s^* + C_2
\end{aligned}
\tag{B.4}
$$

Applying the no-slip condition ($u^*(0) = 0$) and no shear stress at the centerline ($\frac{\partial u^*}{\partial s^*}(1) = 0$) gives the final velocity profile:

$$
u^* = -\frac{T_i\epsilon}{2T_a\delta}\left(s^{*2} - 2s^*\right) \tag{B.5}
$$

Temperature Profile

In order to determine the fluid temperature profile, EQUATION B.5 is used to solve the non-dimensionalized and transformed energy equation.

$$
\begin{aligned}
f\rho u &= -\frac{\partial}{\partial y}\left(\mu u\frac{\partial u}{\partial y}+\kappa_f\frac{\partial T}{\partial y}\right) \qquad \text{(B.6)}\\
f\rho u &= \underbrace{-u\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right)}_{f\rho u}-\mu\left(\frac{\partial u}{\partial y}\right)^2-\frac{\partial}{\partial y}\left(\kappa_f\frac{\partial T}{\partial y}\right)\\
0 &= -\mu\left(\frac{\partial u}{\partial y}\right)^2-\frac{\partial}{\partial y}\left(\kappa_f\frac{\partial T}{\partial y}\right)\\
0 &= -\frac{\mu_w T u_a^2}{T_i h^2}\left(\frac{\partial u^*}{\partial y^*}\right)^2-\frac{1}{h}\frac{\partial}{\partial y^*}\left(\frac{\kappa_{f_w} T T_a}{T_i h}\frac{\partial T^*}{\partial y^*}\right)\\
0 &= -\frac{\mu_w T_a^2 u_a^2}{T T_i h^2}\left(\frac{\partial u^*}{\partial s^*}\right)^2-\frac{T_a}{T h}\frac{\partial}{\partial s^*}\left(\frac{\kappa_{f_w} T_a^2}{T_i h}\frac{\partial T^*}{\partial s^*}\right)\\
\frac{\partial^2 T^*}{\partial s^{*2}} &= -\frac{\mu_w u_a^2\epsilon^2 T_i^3}{\delta^2 T_a^3}(s^*-1)^2\\
\frac{\partial^2 T^*}{\partial s^{*2}} &= -\frac{(\gamma-1)PrM^2\epsilon^2 T_i^3}{\delta^2 T_a^3}(s^*-1)^2
\end{aligned}
$$

Introducing the non-dimensional variable $\beta = \dfrac{(\gamma-1)PrM^2\epsilon^2 T_i^3}{\delta^2 T_a^3}$, the temperature profile becomes:

$$T^* = -\beta\frac{(s^*-1)^4}{12} + C1s^* + C2 \qquad \text{(B.7)}$$

Applying a prescribed interface temperature ($T^*(0) = T_i/T_a$) and zero heat flux at the centerline ($\frac{\partial T^*}{\partial s^*}(1) = 0$) gives the final temperature profile:

$$T^* = \frac{T_i}{T_a} - \frac{\beta}{12}\left(s^{*4} - 4s^{*3} + 6s^{*2} - 4s^*\right) \qquad \text{(B.8)}$$

Coordinate Transformation and T_a

To determine the *physical*-coordinate temperature profile, the reference temperature, T_a, is derived from the coordinate transformation definition:

$$
\begin{aligned}
T^* \partial s^* &= \partial y^* \qquad \text{(B.9)} \\
\int_0^1 T^* \partial s^* &= \int_0^1 \partial y^* \\
\int_0^1 T^* \partial s^* &= y^*(1) - y^*(0) \\
\frac{T_i}{T_a} + \frac{\beta}{15} &= 1
\end{aligned}
$$

Since $\epsilon \propto 1/T_a^2$ and consequently $\beta \propto 1/T_a^5$, a new variable that is independent of T_a, $\phi = \beta T_a^5$, is defined. Combining this with the result above yields a fifth order system for T_a:

$$
T_a^5 - T_i T_a^4 - \frac{\phi}{15} = 0 \qquad \text{(B.10)}
$$

The solution to this system gives the value of T_a, which is used to fully define the velocity and temperature profile. Having the velocity and temperature profile defined, the density distribution is determined from the state equation using an assumed constant pressure.

B.5 Solid Slab Solution

The solid's temperature profile is governed by the heat conduction equation:

$$
energy: \quad -\kappa_s \frac{\partial^2 T}{\partial y^2} = 0
$$

For convergence studies, it is desired to have an analytical solution that is not within the polynomial space. Therefore, a source term must be added to ensure a solution outside the polynomial space. In this case, a sinusoidal source term is added:

$$-\kappa_s \frac{\partial^2 T}{\partial y^2} = \tau \kappa_s \sin(y + H)$$

where H is the slab thickness. This results in the following profile:

$$T_s = \tau \sin(y + H) + C_1 y + C_2 \tag{B.11}$$

For the conjugate problem, temperature and heat flux continuity at the interface must be enforced. By setting the solid wall heat flux equal to that of the fluid ($\kappa_s \frac{\partial T}{\partial y}_s = \kappa_f \frac{\partial T}{\partial y}_f$) at the interface, the solid-side temperature gradient is determined to be:

$$\left. \frac{\partial T}{\partial y}_s \right|_{interface} = \left(\frac{\kappa_f T_a^2 \beta}{\kappa_s 3 h T_i} \right) \tag{B.12}$$

This along with the interface temperature condition ($T(0) = T_i$) gives the resulting temperature profile:

$$T_s = \tau \sin(y + H) + \left(\frac{\kappa_f T_a^2 \beta}{\kappa_s 3 h T_i} - \tau \cos(H) \right) y + T_i - \tau \sin(H) \tag{B.13}$$

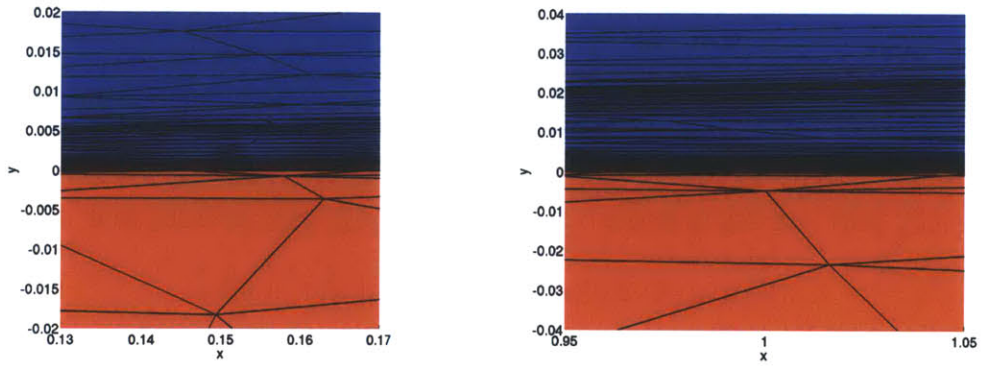
In order to properly bound this profile, an exact temperature state is required on the lower wall boundary:

$$T_w = \left(\frac{\kappa_f T_a^2 \beta}{\kappa_s 3 h T_i} - \tau \cos(H) \right) (-H) + T_i - \tau \sin(H) \tag{B.14}$$

Appendix C

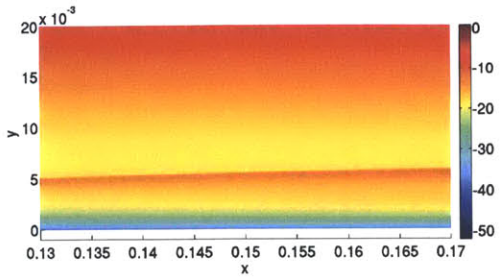
RANS Boundary Layer Adjoint Jump

The conjugate RANS case developed in SECTION 5.2 yields optimal meshes with highly resolved features along the edge of the boundary layer. With the solid Dirichlet condition set to a fourth of the fluid's free-stream total temperature, the interface encounters high heat transfer from the fluid to the solid. FIGURE C-1 shows the refinement near the boundary layer for a $p = 2$ 26k DOF case at two different locations along the solid wall interface. In this high heat transfer case, jumps in the heat flux adjoint for the mass and momentum equations are observed. FIGURE C-2 illustrates the heat flux adjoint solutions to the mass, x-momentum, energy, and SA equations at $x = 0.15$ and $x = 1.0$.

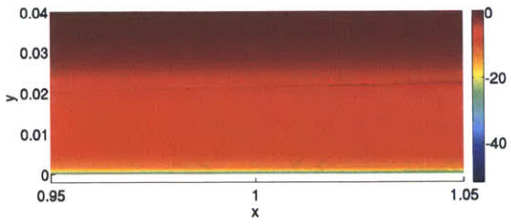


(a) p=2 26k DOF $x = 0.15$ (b) p=2 26k DOF $x = 1.0$

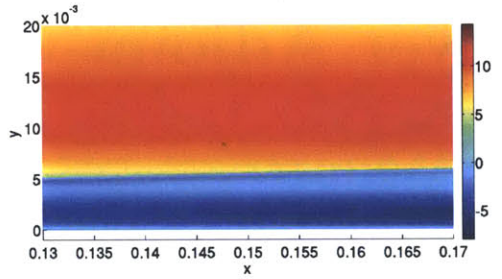
FIGURE C-1: RANS flat slab heat flux adapted mesh



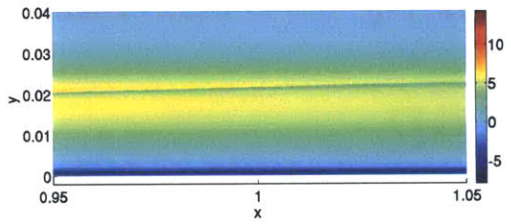
(a) ψ_{heat} (mass) p=2 26k DOF $x = 0.15$



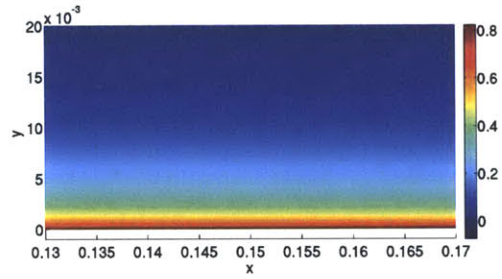
(b) ψ_{heat} (mass) p=2 26k DOF $x = 1.0$



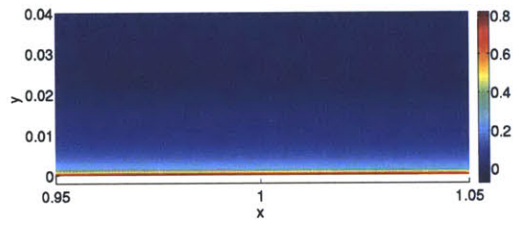
(c) ψ_{heat} (x-mom) p=2 26k DOF $x = 0.15$



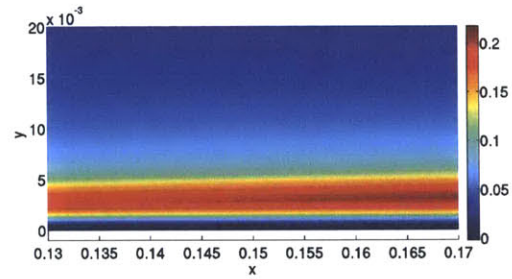
(d) ψ_{heat} (x-mom) p=2 26k DOF $x = 1.0$



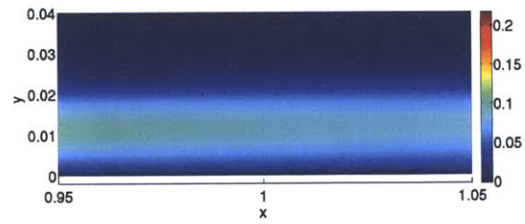
(e) ψ_{heat} (energy) p=2 26k DOF $x = 0.15$



(f) ψ_{heat} (energy) p=2 26k DOF $x = 1.0$



(g) ψ_{heat} (SA) p=2 26k DOF $x = 0.15$

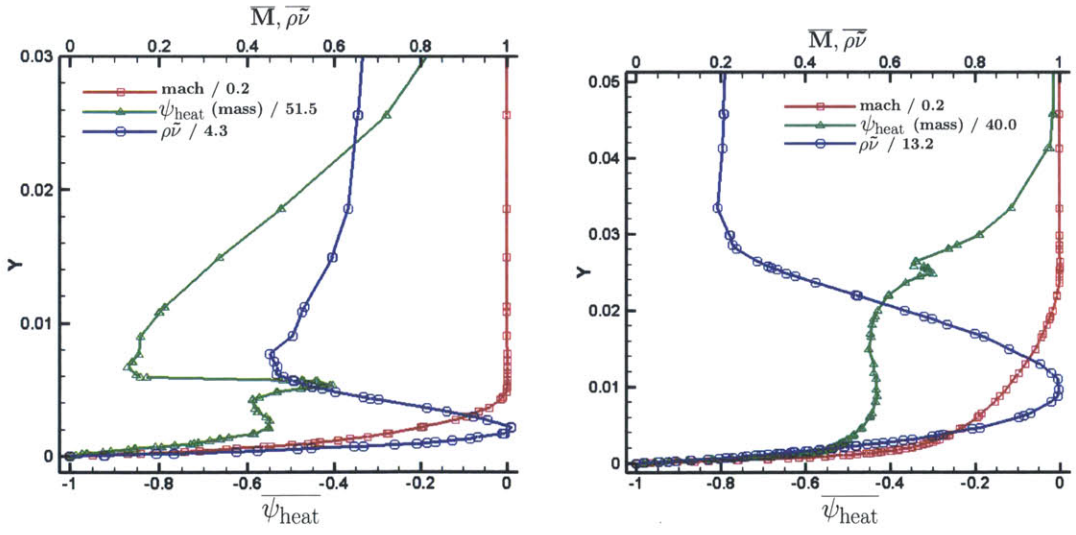


(h) ψ_{heat} (SA) p=2 26k DOF $x = 1.0$

FIGURE C-2: RANS flat slab adjoint profiles

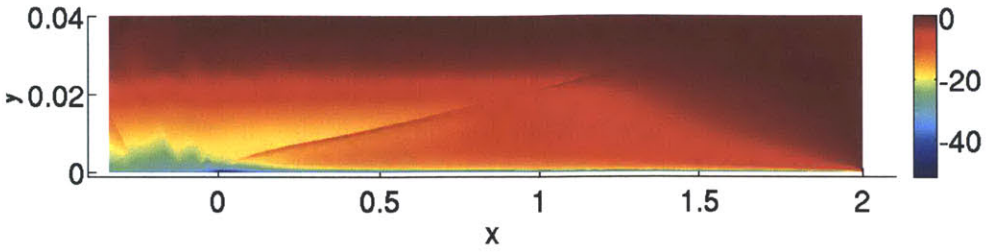
For the heat flux adjoint solutions to the mass and momentum equations, a 'reverse' adjoint boundary layer still forms, though an additional forward convecting jump is

observed. For the energy equation, the heat flux adjoint reaches a maximum near the interface wall since a positive perturbation in the energy would lead to a larger thermal gradient and hence a higher heat flux through the interface (recall that the heat flux from the fluid to the solid is positive). Additionally, the heat flux adjoint solution to the SA equation exhibits only a forward convecting profile, suggesting that the production and destruction terms have a stronger influence over the interface heat flux than the convective terms. Since both the primal and dual solutions are important in controlling the output error, the mesh adaptation method is expected to resolve error contributions originating from both primal and dual features. Presumably, the jump in the heat flux adjoint solution to the mass equation is influenced by the SA equation since there is a correlation between the adjoint jump and an undershoot in the conservative SA working variable. FIGURE C-4 shows both the heat flux adjoint to the mass equation and the conservative SA working variable overlaid with scaled meshes in order to visualize the interaction. Furthermore, FIGURE C-3 illustrates slices of the boundary layer profiles at $x = 0.15$ and $x = 1.0$.

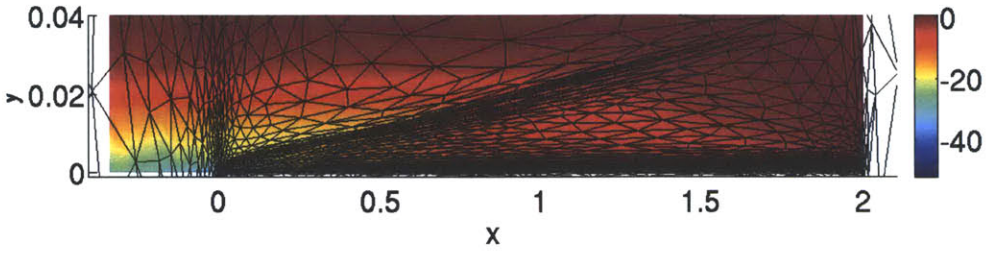


(a) $x = 0.15$ (b) $x = 1.0$

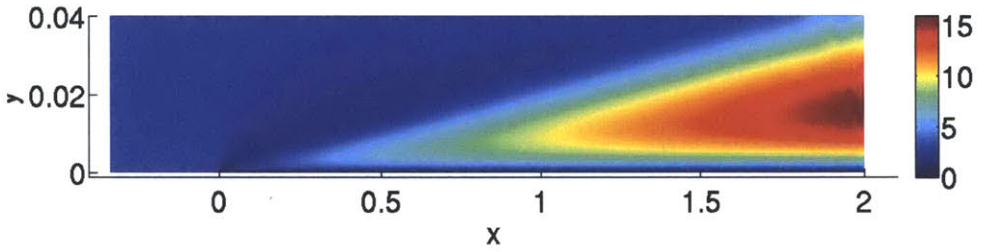
FIGURE C-3: RANS flat slab normalized boundary layer profiles



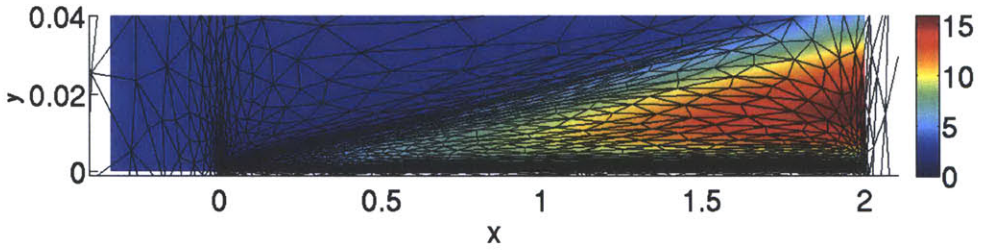
(a) ψ_{heat} (mass) p=2 26k DOF



(b) ψ_{heat} (mass) p=2 26k DOF mesh



(c) $\rho\tilde{\nu}$ p=2 26k DOF



(d) $\rho\tilde{\nu}$ p=2 26k DOF mesh

FIGURE C-4: RANS flat slab mesh correlation (NOT TO SCALE)

In addition to the results presented here, a single-disciplinary fluid case with a constant temperature wall condition was developed. The wall temperature was set equivalent to the flat slab's lower wall temperature, and yielded very similar results. Though these results may reveal a new behavior of the SA turbulence model under high heat transfer conditions, further investigation is still required.

Bibliography

[1] Steven R. Allmaras, Forrester T. Johnson, and Philippe R. Spalart. Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model. Seventh international conference on computational fluid dynamics, Big Island, Hawaii, 2012.

[2] Pereira J. C. F. Armaly B. F., Durst F. and Schonung B. Experimental and theoretical investigation of backward-facing step. *J. Fluid Mech.*, 127:473–496, 1983.

[3] D. N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19:742–760, 1982.

[4] 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.

[5] T. J. Baker. Mesh adaptation strategies for problems in fluid dynamics. *Finite Elements Anal. Design*, 25:243–273, 1997.

[6] F. Bassi, A. Crivellini, S. Rebay, and M. Savini. Discontinuous Galerkin solution of the Reynolds averaged Navier-Stokes and k-ω turbulence model equations. *Computers & Fluids*, 34:507–540, May-June 2005.

[7] 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.

[8] 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.

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

[10] 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.

[11] 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.

[12] 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.

[13] Durst F. Biswas G., Breuer M. Backward-facing step flows for various expansion rations at low and moderate reynolds numbers. *Journal of Fluids Engineering*, 126:362–374, 2004.

[14] H. Borouchaki, P. George, F. Hecht, P. Laug, and E Saltel. Mailleur bidimensionnel de Delaunay gouverné par une carte de métriques. Partie I: Algorithmes. INRIA-Rocquencourt, France. Tech Report No. 2741, 1995.

[15] Houman Borouchaki, Paul Louis George, Frédéric Hecht, Patrick Laug, and Eric Saltel. Delaunay mesh generation governed by metric specifications. Part I algorithms. *Finite Elements in Analysis and Design*, 25(1-2):61–83, 1997.

[16] Philip Caplan. An adaptive framework for high-order, mixed-element numerical simulations. Master's thesis, Mass. Inst. of Tech., Department of Aeronautics and Astronautics, 2014.

[17] A. Celic and E.H. Hirschel. Comparison of eddy-viscosity turbulence models in flows with adverse pressure gradient. *AIAA Journal*, 44(10), 2006.

[18] 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.

[19] B. Cockburn, S. Hou, and C. W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case. *Mathematics of Computation*, 54:545–581, 1990.

[20] 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.

[21] 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.

[22] 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.

[23] K. D. Cole. Conjugate heat transfer from a small heated strip. *International Journal of Heat and Mass Transfer*, 40:2709–2719, 1997.

[24] Michael Collins, Frank Vecchio, Robert Selby, and Pawan Gupta. Failure of an offshore platform. *Concrete International*, 19(8):28–35, 1997.

[25] Timothy A. Davis. Algorithm 832: UMFPACK V4.3 - An unsymmetric-pattern multifrontal method. *ACM Transactions on mathematical software*, 30(2):196–199, 2004.

[26] W. N. Dawes, P. C. Dhanasekaran, A. A. J. Demargne, W. P. Kellar, and A. M. Savill. Reducing bottlenecks in the CAD-to-mesh-to-solution cycle time to allow CFD to participate in design. *Journal of Turbomachinery*, 123(11):552–557, 2001.

[27] S. De Rango and D. W. Zingg. High-order spatial discretizatoin for turbulent aerodynamic computations. *AIAA Journal*, 39(7), 2001.

[28] Laslo T. Diosady and David L. Darmofal. Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations. *Journal of Computational Physics*, 228:3917–3935, 2009.

[29] Barton I. E. The entrance effect of laminar flow over a backward-facing step geometry. *International Journal for Numerical Methods in Fluids*, 25:633–644, 1997.

[30] Ronald P Fedkiw, Tariq Aslam, Barry Merriman, and Stanley Osher. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *Journal of Computational Physics*, 152(2):457 – 492, 1999.

[31] Krzysztof Fidkowski and David Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA Journal*, 49(4):673–694, 2011.

[32] Krzysztof J. Fidkowski and David 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.

[33] Roy C. Freeman J. Verification and validation of rans turbulence models in commercial flow solvers. AIAA 2012-462, 2012.

[34] 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.

[35] 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.

[36] R. Hartmann. Adjoint consistency analysis of discontinuous Galerkin discretizations. *SIAM Journal on Numerical Analysis*, 45(6):2671–2696, 2007.

[37] F. Hecht. Bamg: Bidimensional anisotropic mesh generator, 1998. `http://www-rocq1.inria.fr/gamma/cdrom/www/bamg/eng.htm`.

[38] Bernt Jakobsen and Finn Rosendahl. The Sleipner platform accident. *Structural Engineering International*, 4(3):190–193, 1994.

[39] Kudo Jun. Robust adaptive high-order rans methods. Master's thesis, Mass. Inst. of Tech., CDO, 2014.

[40] Moffatt M. K. Viscous and resistive eddies near a sharp corner. *J. Fluid Mech.*, 18:1–18, 1963.

[41] Das M. K. Kanna P. R. Conjugate forced convection heat transfer from a flat plate by laminar plane wall jet flow. *International Journal of Heat and Mass Transfer*, 48:2896–2910, 2005.

[42] Das M. K. Kanna P. R. Conjugate heat transfer study of a two-dimensional laminar incompressible wall jet over a backward-facing step. *J. Heat Transfer*, 129:220–231, 2007.

[43] Steve L. Karman. SPLITFLOW: A 3D unstructured Cartesian/prismatic grid CFD code for complex geometries. AIAA 1995-0343, 1995.

[44] T. Leicht and R. Hartmann. Error estimation and hp-adaptive mesh refinement for discontinuous Galerkin methods. In Z. J. Wang, editor, *Adaptive High-Order Methods in Computational Fluid Dynamics*, pages 67–94. World Science Books, 2011.

[45] 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.

[46] Randall J. LeVeque and Zhilin Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis*, 31(4):1019–1044, 1994.

[47] Adrien Loseille and Frédéric Alauzet. Optimal 3D highly anisotropic mesh adaptation based on the continuous mesh framework. In *Proceedings of the 18th International Meshing Roundtable*, pages 575–594. Springer Berlin Heidelberg, 2009.

[48] Adrien Loseille and Frédéric Alauzet. Continuous mesh framework part I: Well-posed continuous interpolation error. *SIAM Journal on Numerical Analysis*, 49(1):38–60, 2011.

[49] Adrien Loseille and Frédéric Alauzet. Continuous mesh framework part II: Validations and applications. *SIAM Journal on Numerical Analysis*, 49(1):61–86, 2011.

[50] James 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.

[51] James M. Modisette. *An Automated Reliable Method for Two-Dimensional Reynolds-averaged Navier-Stokes Simulations.* PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, September 2011.

[52] M. Mosaad. Laminar forced convection conjugate heat transfer over a flat plate. *Heat and Mass Transfer*, 35:371–375, 1999.

[53] R. S. Myong. A full analytical solution for the force-driven compressible Poiseuille gas flow based on a nonlinear coupled constitutive relation. *Phys. of Fluids*, 23, 2011.

[54] Shankar P. N. Moffatt eddies in the cone. *J. Fluid Mech.*, 539:113–135, 2004.

[55] N.C. Nguyen, Per-Olof Persson, and Jaime Peraire. RANS solutions using high order discontinuous Galerkin methods. AIAA 2007-0914, 2007.

[56] Todd Oliver and David Darmofal. Analysis of dual consistency for discontinuous galerkin discretizations of source terms. *SIAM Journal on Numerical Analysis*, 47(5):3507–3525, 2009.

[57] Todd A. Oliver. *A Higher-Order, Adaptive, Discontinuous Galerkin Finite Element Method for the Reynolds-averaged Navier-Stokes Equations.* PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2008.

[58] C. Péniguel. Heat transfer simulation for industrial applications: needs, limitations, expectations. *Internat. J. Heat Fluid Flow*, 19(2):102 – 114, 1998.

[59] Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A Riemannian framework for tensor computing. *International Journal of Computer Vision*, 66(1):41–66, 2006.

[60] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz. Adaptive remeshing for compressible flow computations. *Journal of Computational Physics*, 72:449–466, 1987.

[61] Per-Olof Persson and Jaime Peraire. Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier-Stokes equations. *SIAM Journal on Scientific Computing*, 30(6):2709–2722, 2008.

[62] James W. Purvis and John E. Burkhalter. Prediction of critical Mach number for store configurations. *AIAA Journal*, 17(11):1170–1177, 1979.

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

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

[65] T.A. Rizk, C. Kleinstreuer, and M.N. Özisik. Analytic solution to the conjugate heat transfer problem of flow past a heated block. *International Journal of Heat and Mass Transfer*, 35(6):1519–1525, 1992.

[66] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357–372, 1981.

[67] Thangam S. and Speziale C. G. Turbulent flow past a backward-facing step: A critical evaluation of two-equation models. *AIAA Journal*, 30:1314–1320, 1992.

[68] Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986.

[69] Yousef Saad. *Iterative Methods for Sparse Linear Systems.* Society for Industrial and Applied Mathematics, 1996.

[70] Philippe R. Spalart and Steven R. Allmaras. A one-equation turbulence model for aerodynamics flows. *La Recherche Aérospatiale*, 1:5–21, 1994.

[71] Huafei Sun. *A Robust Simplex Cut-Cell Method for Adaptive High-Order Finite Element Discretizations of Aerodynamics and Multi-Physics Problems.* PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, August 2013.

[72] Huafei Sun and David L. Darmofal. An adaptive simplex cut-cell method for high-order discontinuous galerkin discretizations of multi-material and multi-physics problems. AIAA CFD Conference, 2013.

[73] 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.

[74] Kanev K. Pop I. Vynnycky M., Kimura S. Forced convection heat transfer from a flat plate: the conjugate problem. *International Journal of Heat and Mass Transfer*, 41(1):45 – 59, 1998.

[75] Gary P. Warren, W. Kyle Anderson, James L. Thomas, and Sherrie L. Krist. Grid convergence for adaptive methods. AIAA 91-1592, 1991.

[76] M. Wheeler. An elliptic collocation-finite element method with interior penalties. *SIAM Journal on Numerical Analysis*, 15:152–161, 1978.

[77] Masayuki Yano. *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes.* PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2012.

[78] Masayuki Yano and David L. Darmofal. An optimization-based framework for anisotropic simplex mesh adaptation. *Journal of Computational Physics*, 231(22):7626–7649, September 2012.

[79] Masayuki Yano, James M. Modisette, and David Darmofal. The importance of mesh adaptation for higher-order discretizations of aerodynamic flows. AIAA 2011–3852, June 2011.

[80] David P. Young, Robin G. Melvin, Michael B. Bieterman, Forrester T. Johnson, Satish S. Samant, and John E. Bussoletti. A higher-order boundary treatment for Cartesian-grid methods. *Journal of Computational Physics*, 92:1–66, 1991.

[81] Xu Dong Zhang, Marie-Gabrielle Vallet, Julien Dompierre, Paul Labbe, Dominique Pelletier, Jean-Yves Trepanier, Ricardo Camarero, Jason V. Lassaline, Luis M. Manzano, and David W. Zingg. Mesh adaptation using different error indicators for the Euler equations. AIAA 2001-2549, 2001.