# Finite Element Modeling and Optimization of High-Speed Aerothermoelastic Systems

by

**Micah A. Howard**

Bachelor of Science, University of Colorado, 2005

Master of Science, University of Colorado, 2007

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Aerospace Engineering Sciences

2010

This thesis entitled:
Finite Element Modeling and Optimization of High-Speed Aerothermoelastic Systems
written by Micah A. Howard
has been approved for the Department of Aerospace Engineering Sciences

_____

Professor Kurt Maute

_____

Professor Carlos Felippa

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Howard, Micah A. (Aerospace Engineering Sciences)

Finite Element Modeling and Optimization of High-Speed Aerothermoelastic Systems

Thesis directed by Professor Kurt Maute

The design of supersonic and hypersonic aerospace vehicles is by nature a multi-disciplinary problem requiring the close integration of compressible fluid dynamics, heat transfer, and structural dynamics. The transient flow around the body must be accurately characterized in order to assess its affect on the thermal and structural responses; conversely, the thermal and structural behavior may significantly alter the aerodynamic performance. The core of this dissertation effort is concerned with the development and demonstration of an analysis and design capability for the aerothermoelastic behavior of high-speed aerospace vehicles. This nominally involves coupling of the compressible Navier-Stokes equations for the fluid dynamics, the transient heat equation for the thermal response, and the elastodynamic equations for the structural dynamics. The streamline upwind Petrov-Galerkin (SUPG) stabilized finite element method is used for solving the compressible flow problem. Both a standard Galerkin and stabilized Galerkin gradient least squares (GGLS) finite element method are utilized for solving the heat equation, and a standard Galerkin method is used for solving the elastodynamic equations. The transient and steady-state responses of a problem are determined via a single, simultaneously coupled nonlinear system, thus bypassing accuracy and stability issues of classical staggered multi-physics coupling strategies. A gradient-based optimization framework is developed for designing transient coupled aerothermoelastic systems via adjoint-based sensitivity analysis. This framework is used to optimize the design of a structure in regard to thermal and structural performance. The efforts of this thesis have yielded a state-of-the-art approach for coupled aerothermoelastic analysis and design optimization.

## Dedication

Dedicated to my wife and parents, for all the support they have given me.

## Acknowledgements

A great deal of thanks is owed to my advisor, Professor Kurt Maute, for lending his time, knowledge, and support over the course of my Ph.D. studies. Kurt has a gift for clear and concise teaching and an enthusiasm and energy for research like few others; for this, I am lucky to have been a part of his research group for the last 5 years. I would like to acknowledge the members of my committee for reviewing my work, providing feedback, and directing the course of this research. I gratefully acknowledge the Office of Naval Research and the MIDE Corporation, who provided funding support over the course of several phases of the Naval Electromagnetic Railgun Small Business Technology Transfer (STTR). I have the utmost respect and thanks for Dr. Artie Mabbett, who served as the technical program manager throughout this project. I am inspired by his expertise in the field of hypersonics and aeroheating. It was through the railgun project and interactions with Dr. Mabbett that I received much of the motivation and many of the ideas for this thesis. Acknowledgments are also due to several members of the technical staff at Sandia National Laboratories. Over the summer of 2008, I had the privilege of working with Drs. Garth Reese and Tim Walsh in the Computational Structural Dynamics group. Through the suggestion of Dr. Basil Hassan, I returned to Sandia the following summer to work with the Computational Thermal and Fluid Mechanics group and take advantage of several common research interests. Many thanks are owed to Drs. Ryan Bond and Steve Bova, who provided countless suggestions and many fruitful discussions, which in particular greatly influenced the development and improvement of the stabilized finite element formulation used for the compressible fluid dynamics portion of this thesis. I would also like to acknowledge Dr. Jeff Payne, manager of the Aerosciences Department

and Sandia National Laboratories, who generated some of the data used to verify the fluid flow solver developed for this thesis. I am grateful for the help and teaching I received from to Dr. Manuel Barcelos, who was instrumental in getting me started with this work during my early days as a graduate student. Recognition is owed to Dr. Gary Weickum, Sebastian Kreissl, and Chris DeLuca, my fellow office mates over the years, for their friendship and willingness to listen to and discuss research issues. Finally, I want to thank my wife and parents for their support throughout my entire graduate school tenure and for their support of all my goals in life.

# Contents

# Tables

**Table**

# Figures

**Figure**

# Chapter 1

# Introduction

## 1.1 High-Fidelity Aerothermoelastic Analysis of High-Speed Vehicles

The design of modern aerospace structures that are capable of flying at hypersonic speeds requires not only careful analysis of the relevant individual physical phenomena, for instance the compressible fluid flow around the body, heat transfer through it, and the structural response, but also necessitates special consideration of the interplay between them. Examples of real-world aerospace structures requiring multi-disciplinary design treatment are that of supersonic aircraft and both blunt and slender body atmospheric re-entry vehicles. For these structures the interplay of the aerodynamics, heat transfer, and structural dynamics simply cannot be ignored. The objective of this dissertation is to develop a tightly coupled aerothermoelastic computational analysis and design methodology, and to demonstrate its uses and advances for solving problems relating to transient high-speed flight conditions. Towards this end, this work utilizes the finite element method for solving the compressible viscous flow, heat transfer, and structural response problems in a coupled fashion, and uses numerical optimization techniques to demonstrate the design potential of these methods.

Much effort has gone into the development of analysis tools for specific disciplines. For example, the use of computational fluid dynamics to estimate the drag of a complete aircraft is now a routine calculation performed by many organizations [2]. Likewise, the practice of computational structural dynamics is a commonplace activity in nearly every aerospace engineering organization. Traditionally though, these methods are used almost exclusively as stand alone analysis tools,

which assume loads and boundary conditions that crudely represent the physical affects of another interacting field. This has led to the so-called "throw it over the wall" analysis and design approach where individual disciplines are solved with minimal interaction between one another. Clearly, this method of analysis is highly modular and easy to implement but this comes at the expense of solution fidelity, especially for transient and/or highly coupled physical phenomena such as aeroelastic or aeroheating problems.

Due to the deficiencies of the aforementioned design process, efforts have been made to improve upon computational methods for coupled systems and in particular we focus here on coupled aerodynamics problems. For instance, advances to the state-of-the art have been made by Farhat and coworkers [40] for coupled transient aeroelastic problems, Tran et al. [140] worked on coupled aerothermoelasticity, Hassan et al. [51, 52] have developed a simulation tool for transient aeroheating and ablation problems along with similar work by Candler [26] and Martin and Boyd [103]. There are two predominant themes which these efforts all share and where this dissertation work makes a departure from the *status quo* for aeroheating and aerothermoelastic simulations: the choice of flow solver and the choice of coupling strategy.

### 1.1.1 Compressible Flow Solver

Each of the previously mentioned efforts all use a finite volume method for solution of the flow problem. This choice is with good reason as the *de facto* standard in modern computational methods for compressible gas dynamics is the finite volume method. Numerous commercial (e.g. Fluent [77]), government (e.g. DPLR [131]), and academic (e.g. AERO-F [42]) codes exist for solving such flow problems, all which utilize the finite volume method. While the finite element method has been employed and used almost exclusively for heat transfer and structural dynamics problems (such as in ABAQUS [73], ANSYS [76], and NASTRAN [31]) it has also been successfully applied (in a stabilized form) to solve compressible flow problems [136, 137, 71, 125, 4, 70]. Though the stabilized finite element method has not gained as much popularity as the finite volume method, several promising developments have clearly demonstrated its applicability for high speed, shock

dominated flows. In particular, Chalot [28] applied it to solve reacting hypersonic flows in chemical equilibrium, Tezduyar et al [138] have developed new shock capturing techniques, and Kirk [85] and Bova [23] have highlighted several improvements to the basic formulation.

Due to the successes of the stabilized finite element method for high-speed flows, this work has adopted and implemented the streamline-upwind Petrov-Galerkin (SUPG) finite element formation for solving compressible gas dynamics problems. It should also be noted that discontinuous Galerkin methods are another popular class of finite element based methods for fluid flow problems and have been successfully used for compressible flows (see for instance [12, 14]). While this decision diverges from the popular choice of a finite volume solver, it was also a strategic decision as discussed next.

### 1.1.2 Coupling Strategy

The other predominant similarity between the previously listed multi-disciplinary analysis efforts is their choice of how they couple the individual disciplines. All of these works use what is referred to by Felippa [45] and Farhat [43, 44] as a partitioned or staggered coupling strategy. Coupling is achieved by sequentially advancing one physical problem in time, passing relevant boundary conditions back to another physical problem and advancing it in time, and then passing results and boundary conditions from the second problem to the first and continuing on in this fashion. Farhat [40] and Tran [140] provide clear schematics for staggered aeroelasticity and aerothermoelasticity while Hassan et al. [52] show a staggered coupling flow chart for transient ablation and aeroheating along a flight trajectory.

The primary reason the partitioned coupled approach has been so popular is that it easily facilitates coupling by integrating existing but often separate analysis codes. As previously mentioned, the finite volume method is primarily used to solve compressible fluid dynamics problems while the finite element method is typically used for heat transfer and structural dynamics. Historically and with few exceptions, the analysis capabilities are encapsulated in separate codes since they use different spatial discretization methods. In fact, it is not uncommon for the heat transfer and structural dynamics solvers (which usually both use the finite element method) to be contained

in different software packages. Hence, the idea of partitioned coupling is a natural one due to the modular nature of most analysis codes.

The partitioned coupled approach, however, has a few common pitfalls that are not easily avoided. The first is accuracy and implicitness. The partitioned coupled approach is, to some extent, just an automated "throw it over the wall" procedure, and since lines are drawn according to which code solves which discipline important interactions may be neglected. For instance, when an chemically ablating material encounters a chemically reacting flow the chemical species from the two will most likely be continuously interacting. With the partitioned coupling strategy though, as one discipline is advances in time the other(s) remain behind in time, thus it is much more difficult to achieve a truly implicit scheme as the equations are satisfied using solution information at an old time level. This problem is lessened by reducing the time step, but the main advantage of using an implicit method in the first place was being able to take large time steps afforded by the unconditional stability of an implicit scheme. While it is possible to achieve a partitioned coupling strategy that is second order accurate and unconditionally stable, it is far more difficult than for monolithically coupled schemes.

The idea of the partitioned coupling method has also led to the sometimes controversial term "multi-physics" simulation. Some may argue that there are not multiple physics in the sense that the Navier-Stokes equations governing fluid flow are somehow different from the heat equation or the elastodynamic equation. Rather, these are equations which all obey the same laws of physics such as conservation of mass, momentum, and energy. Along this vain of thought, one can easily argue that it may not make sense that the energy equation for fluid flow is solved separately from the energy equation for the heat transfer in aeroheating problems, or that the momentum equation for fluid flow is solved separately from the momentum equation for structural dynamics. In the interacting system, mass, momentum, and energy must be conserved at each point in time so why pull the equations apart and satisfy them separately (as the partitioned approach does)? As a result, the coupling strategy in this thesis differs from the partitioned approach by handling all of the governing equations in a single nonlinear system. This leads to a dilemma which was already

encountered. In order to solve the conservation equations coupled across fluid and solid boundaries, a code is needed that is capable of delivering those equations into a single time integration method and a single nonlinear system solver. Most simulation codes typically focus on solving a single problem type and thus are not designed to solve the equations across fluid and solid boundaries simultaneously – the work contained herein addresses this shortcoming.

### 1.1.3 Additional Comments

The decision to use the finite element method in this work to discretize the equations for fluid flow, heat transfer, and structural response was made to better facilitate the one code, one nonlinear system philosophy. Certainly, it is in the realm of possibility to follow this philosophy and still discretize the equations partially with a finite volume method and the rest with a finite element method; however, maintaining a consistent discretization approach has its benefits in terms of implementation simplicity. While the finite volume method has been applied to solving the heat and elasticity equations, it has seen little acceptance by the computational mechanics community for these purposes.

A comment must also be made in regards to the "high-fidelity" term used in the title of this section. There are many computational techniques from which an engineer has to choose from to perform an analysis or simulation of a real-world event. Many of these tools are regarded as approximation methods, which make rather significant assumptions or neglect important portions of the physics that are being simulated. Examples of such techniques are lattice vortex methods and boundary layer approximation codes for compressible gas dynamics or one-dimensional heat transfer codes applied to multi-dimensional problems. This is not to suggest that these codes are inaccurate or inferior when applied to the appropriate problem, but it is up to the engineer to decide what assumptions are appropriate and which part of the physics are negligible. The methods used in this thesis fall under the category referred to by the computational mechanics community as "high-fidelity" because every attempt is made to discretize the governing equation in its entirety and by a method which exhibits at least second-order accuracy in space and at least second-order

accuracy in time for transient problems.

## 1.2 Design Optimization for Aerothermoelastic Problems

Design optimization for aerodynamic problems has been a field of intense research for over 30 years, highlighted by the pioneering work of Jameson [80]. Yet analysis for design purposes, especially aerodynamic analysis, is most useful when coupled with the response of the structure. Maute [105] and Martins [104] published notable efforts for aerostructural shape optimization in the context of inviscid flows at steady-state conditions. Few published efforts have been made to perform steady-state aeroheating or aerothermoelastic optimization. Since the transient behavior of aeroheating and aerothermoelasticity is often of primary interest, steady-state studies offer only limited insight. Furthermore, the concept of design optimization over a transient time period has seen little development, though a very notable exception has been recently presented [110]. Almost without doubt, the combination of transient shape optimization for aerothermal or aerothermoelastic problems is an exciting area of exploration.

Motivated by the potential for design improvements of high-speed aerospace structures, one of the objectives of this work is to begin exploring the area of transient design optimization. Given the complex nature of the interactions which exist in problems where high-speed flow is coupled with thermal and elastic response of a structure, the need for a mathematically rigorous method for navigating this design space is large. As opposed to steady-state behavior, transient responses may lead to hard to find or non-intuitive designs especially in the context of atmospheric re-entry heating. For example, it is a well known fact that the bluntedness of a body flying at supersonic speeds will effect the strength of the bow shock wave and hence the amount of drag and heating the body will experience. This is clearly evident in the design of manned re-entry capsules, which are extremely blunted to increase wave drag and reduce heating, and the design of slender body re-entry vehicles used for military purposes where a premium is placed on decreased drag at the expense of increased heating.

Given the transient nature of a re-entry problem, finding an optimal design that leads to

desired drag, heating, and structural dynamics characteristics is difficult at best, even for seasoned engineers. This thesis aims at demonstrating such a computational methodology to speed this process and help in generating more efficient structural designs. Granted that a computational tool is no replacement for sound engineering intuition and experience, it is the hope of the author that this effort will serve as a demonstration for the potential of this optimization technique.

One of the unique features in this thesis is the application of topology optimization methods to design the internal material layout of a structure under transient loading. Topology optimization is the most general of the three primary design optimization techniques (the other two being size/parameter optimization and shape optimization) as it assumes no initial design but rather begins from an arbitrarily defined domain inside which the optimization is to be preformed. The concept pursued in this thesis is to use topology optimization to determine the layout of solid material within the structure with different thermodynamic and elastodynamic properties to beneficially alter the transient heat flow and stress conditions. Similar efforts have been made by Maute [106] and James and Martins [79] for generating conceptual designs of wing structures under steady-state conditions.

While shape optimization methods are useful for altering the *external* shape of the structure to minimize internal heating and stresses, the absorption of thermal energy by the body is unavoidable. Consequently, good designs in terms of heat mitigation and management must take into account the behavior of the material within the entire volume in addition to the behavior of the material at the surface.

A simple design concept for the interior of a structure is to consider the best layout of any two given materials – for instance, a material with a high heat capacity and low thermal conductivity (such as aluminum) and another material with a high density and melting point and better elastic properties (such a high strength steel or tungsten). Taking this idea a bit further, this thesis considers phase change effects of the materials with the added benefit of energy absorption via the latent heat effect. Incidentally, aluminum has a particularly high latent heat and its potential for absorbing thermal energy is quite promising. If a material is permitted to change phase the

elastic response of the structure will undoubtedly come into question. In this case, the elastic properties of the second material become even more important. Hence, in this effort topology optimization is performed with both thermal and structural design criteria to ensure structural integrity is maintained while mitigating internal heating.

## 1.3 Software Implementation

Software development is a necessary component of computational mechanics and unfortunately many important computer science aspects are often not understood or ignored by many in the field of computational mechanics. It is only through careful software design and leveraging of existing software packages that a simulation code can gain the flexibility and power needed to become a truly useful design tool. Achieving such an objective is no small feat especially within the scope of any dissertation effort. Given this observation, the finite element code used to complete this work was developed from the ground up (beginning in the summer of 2008) with aspirations of becoming a flexible and high-performance general purpose design tool. Towards this end, the object-oriented programming paradigm afforded by the C++ language has been used to try to maximize code reuse and extensibility. Also important to this development project is the incorporation of well established software packages in order to handle common tasks most finite element codes must perform such as mesh representation and linear equation solving. What follows is a brief description of the third-party libraries that the code developed here interfaces with.

**Mesh and Solution Data Format** The Exodus mesh format [91] developed at Sandia National Labs is used to define and read a computational grid as well as store solution data as it becomes available during a time integration process. Exodus stores information in a binary format and hence makes considerable performance gains in terms of storage and read/write access over the commonly used ASCII data format. Additionally, the Nemesis extension to Exodus is used for the purpose of representing a partitioned mesh needed for a domain decomposition parallel processing approach. The Nemesis format simply adds subdomain connectivity information to a collection of Exodus files. Care has been taken to ensure that the code is not dependent solely on

Exodus, as it uses the Exodus read/write capabilities to construct its own internal mesh objects. Hence, by providing an interface to a different mesh representation, such as that of any number of popular commercial pre- or post-processing tools, one may easily begin to uses meshes defined in a different format. Another advantage of using a well established mesh format such as Exodus is the use of powerful mesh generation tools that can write directly to Exodus format. CUBIT [88] and ICEMCFD [74] are two extremely powerful mesh generators that export directly to Exodus.

**Domain Decomposition and Mesh Utilities** The open-source version of the SEACAS software [94] developed at Sandia National Labs contains tools that perform mesh manipulation. Commonly needed mesh tasks include: domain decomposition operations (using SEACAS' "loadbal" and "epu"), converting an Exodus file to and from an ASCII format (using "exotxt" and "txtexo"), interrogating an Exodus file for results information (using "grope"), comparing the results contained in two Exodus files (using "exodiff"), merging the meshes of two Exodus files (using "gjoin"), and mapping solution data from one mesh to another (using "mapvar"). These tools help in performing the basic tasks every user of a finite element code often needs to do. Without the availability of this software project, the code developed here simply would not be as capable or convenient to use.

**Sparse Matrix Format** Storage of sparse matrices is performed using the Epetra library [90] or the Petsc library [96] that is included as a part of the Trilinos project [95] and Petsc project [96] developed at Sandia National Laboratories and Argonne National Laboratory, respectively. The code accesses Epetra and Petsc through abstract interfaces so it is not explicitly dependent on the these classes, thus the use of another sparse matrix format may be adopted by providing interfaces to any other sparse matrix class. The Epetra and Petsc sparse matrix classes are used by their respective solvers, however the ability to use external classes through interfaces applies to all of the solver capabilities listed below. Epetra and Petsc also provide a convenient means for assembling and solving a distributed linear system in parallel. Hence, by using this sparse matrix format, parallel solution capabilities "come for free".

**Direct Linear Solvers** The Amesos package [89] within Trilinos provides interfaces to several external direct linear solvers such as UMFPACK [139] and SuperLU/SuperLU_dist [126]. Petsc provides similar interfaces to a collection of direct linear solvers as well. Once a linear system has been constructed with the interfaces to the appropriate linear algebra format, a direct linear solution is easily available. In practice, the direct solvers are used for development and debugging purposes while iterative solvers are preferred for larger, memory intensive problems.

**Iterative Linear Solvers and Preconditioners** The AztecOO library [87] within Trilinos provides several common iterative solver implementations such as a preconditioned conjugate gradient (PCG) solver and a generalized minimum residual (GMRES) solver. AztecOO provides many of the common iterative solver preconditioners such as a Jacobi, ILU, and ILUT preconditioner. Petsc's Krylov Subspace (KSP) package provides nearly analogous capabilities.

**Nonlinear Solvers** The nonlinear solver used here is a Newton-Raphson solver built into the code base that has been developed for this thesis work. However, it is a subject of future work to include interfaces to the NOX library [92] provided with Trilinos in order to take advantage of its Jacobian-free Newton-Krylov nonlinear solver capabilities.

**Time Integrators** Three time integrators have been developed for this code to support implicit integration of the first and second-order ordinary differential equations (ODEs) obtained via semi-discretization of governing equations. For first-order ODEs arising from fluid and heat transfer equations, a backward differentiation formula (BDF) scheme and the so-called $\Theta$-scheme are provided. For second-order ODEs arising from structural dynamic equations, a generalized-$\alpha$ time integrator has been implemented. Future work will include an interface to the Rythmos package [93] in Trilinos which provides a BDF time stepping algorithm and also capabilities for transient adjoint integration.

**Solution Visualization** One of the conveniences of using a well established mesh format is also gaining access to a number of powerful post-processing visualization tools. The Exodus format is read directly by the freely available ParaView visualizer [78] as well as the commercially available EnSight visualizer.

## 1.4 Thesis Objectives

Any dissertation effort would not be complete without a clear picture of how the author envisions the work contributing to the state-of-art in the research field. The following is a description of the objectives of this dissertation, and how it fits into the overall landscape of computational mechanics research for aerothermoelastic problems.

**Compressible Fluid Dynamics Solver Development** While the stabilized finite element method developed and used in this thesis is not a new contribution, its use in simulating high-speed, tightly coupled fluid-structure interaction problems is at the forefront of this research area. As previously described, the bulk of all similar analyses are performed with finite volume flow solvers. The use of this discretization procedure for coupled aerothermal and aerothermoelastic problems puts it along side state-of-the-art work being done concurrently by staff members at NASA [84] and Sandia National Labs [23].

**Monolithic Coupling Strategy** As previously discussed in this chapter, the single nonlinear system approach to coupling equations is a departure to the well established approach of partitioned coupling. This work that has been conceived independently from but parallel to an identical approach being used by staff at Sandia National Labs [24]. It is the opinion of the author that the coupling strategy contained herein offers a viable alternative to the partitioned or staggered coupling approaches that is a truly tightly coupled implicit scheme for solving aeroheating and aerothermoelastic problems.

**Transient Design Optimization for Aerothermoelastic Problems** This contribution is made to an area that is quite unique and relatively unexplored. Transient optimization using adjoint sensitivity analysis is an emerging research area, see for example [110], and its application to topology optimization has seen few developments. Undoubtedly, the use of transient adjoint sensitivity techniques using topology optimization for structural design of aerothermoelastic problems is an untouched application of these methods. This thesis demonstrates the promise of this approach.

The transient adjoint sensitivity analysis methods developed and used in this thesis effort may also be applied to several other classes of contemporary and emerging research interests. Uncertainty quantification and error estimation techniques also make use of adjoint sensitivity analysis, and as such, the methods developed herein could be directly applied to these techniques.

**Extensible Platform for Multi-Disciplinary Analysis and Optimization**

Another outcome of this thesis work is the development of a software tool that will hopefully prove to allow easy extension and modification for incorporating new capabilities. Every effort has been made to use sound software engineering practices and mature external packages with the intent of leveraging capabilities where ever possible.

## 1.5 Thesis Organization

Stated briefly, the remainder of this thesis is organized in the following manner: we begin with a description of the methods used to analyze fluid, thermal, and structural response, then discuss coupling of the fields, numerical optimization, and transient design optimization, and finally make concluding remarks. A slightly more in-depth summary of the objectives of each chapter is now described.

Chapter 2 discusses the stabilized finite element formulation used to solve the compressible fluid dynamics problems in this thesis. In particular, the Navier-Stokes governing equations and streamline-upwind Petrov-Galerkin finite element discretization are presented. In order to accommodate moving boundary problems, the spatially discretized form is cast into an arbitrary Lagrangian-Eulerian formulation which is then discussed along with both a first and second order accurate backward difference scheme for time integrating the resulting semi-discrete equations. This chapter is concluded with several numerical example problems which highlight the performance of the implementation for solving representative fluid flow problems demonstrated later on in this thesis.

Standard Galerkin and Galerkin gradient least squares stabilized finite element methods used for solving the transient heat equation are the subject of Chapter 3. Methods for accounting for

phase change and ablation are presented as needed for the later chapter on design optimization are discussed. Again, an arbitrary Lagrangian-Eulerian formulation is used to account for moving boundary problems, and a "Θ-scheme" time integration method used for solving the transient problem is presented. Numerical examples are then presented which demonstrate the transient heat transfer capabilities.

Chapter 4 presents the standard Galerkin implementation for structural dynamics problems and thermoelastic implementation used to simultaneously simulate the coupled structural and thermal response as needed for aerothermoelastic optimization. The generalized alpha class of Newmark time integrators is presented along with several numerical problems that display coupled thermoelastic simulation.

Chapter 5 considers the tightly coupled simulation strategy adopted in this thesis effort. Two methods of coupling the equations across interfaces were initially considered for this work. One, simply called the "residual based" coupling method, acts directly on the residual equations and is based on a similar scheme developed at Sandia National Labs [24]. Another common coupling method is a weak formulation approach such as a mortar method, which has advantage of coupling non-matching, non-contiguous meshes. The residual based approach was adopted for its simplicity and ease of implementation. Numerical examples are presented which use these methods to couple the compressible fluid dynamics equations to the heat and elasticity equations.

Chapter 6 is merely a background and theory chapter on general design optimization procedures. The formulation of an arbitrary nonlinear constrained optimization problem is presented along with sensitivity equations and solution via two popular nonlinear optimization algorithms.

Transient topology optimization is then considered in Chapter 7. A transient adjoint based sensitivity analysis method is described and applied to design the internal material layout of a structure to meet its structural and thermal optimization criteria

Chapter 8 concludes this thesis by summarizing the analysis and design procedures developed and implemented to complete the work contained herein. Statements are made to recap the contributions of this thesis toward advancing the state-of-the-art in the field, and the chapter will

end with a discussion of future work to be performed.

# Chapter 2

# Compressible Computational Fluid Dynamics

## 2.1 Introduction

The intent of this chapter is to discuss the governing equations, discretization, and solution of compressible fluid flow problems. In this work, the solution of the partial differential equations (PDEs) of supersonic flows for problems with moving boundaries is the primary concern. Several numerical example problems are presented that demonstrate the finite element based solution capabilities developed herein to complete the studies contained later on in this thesis.

In the context of computational fluid dynamics, several numerical procedures exist for solving the governing PDEs of fluid flow. Namely, finite difference, finite volume, and finite element techniques are commonly employed to solve such equations. Each of these methods may be viewed, to some extent, as a variation of each other as they each have the end goal of producing a discrete representation of the original PDE on a computational mesh. Also common amongst the methods described below is the concept of numerical upwinding which is needed to eliminate non-physical solution behavior that results from discrete difference approximations inherent to each of these schemes.

**Finite Difference Methods** Finite difference methods are the oldest of the three PDE discretization methods mentioned above and are conceptually the most straightforward. Finite differences employ a direct discretization of the governing equations by replacing difference terms with approximate finite difference formulas such as forward, backward, or central difference schemes. Finite differences have the benefit of ease of implementation but are often limited to simple geometries

since structured meshes are necessary for finite difference solutions. The Lax-Wendroff [98] and MacCormack [101] methods are two of the most popular finite difference schemes for solving the PDEs arising from supersonic flows. While finite difference techniques are mature, useful methods, they are not currently a heavily active area of research as some of the newer methods offer greater promise for addressing the difficulty of solving complex flow problems.

**Finite Volume Methods** Finite volume methods were first introduced by MacCormack and Paullay [102] and have since enjoyed widespread acceptance for solving compressible as well as incompressible flow problems. These methods center around integrating the flux terms of the PDE over a control volume. As opposed to the finite difference method, finite volume methods can more easily account for arbitrary geometries through the use of unstructured meshes. While a large majority of fluid flow solvers utilize the finite volume discretization, this was not the method of choice for reasons that will become more apparent later on in this document.

**Finite Element Methods** The finite element method was conceived in the 1950s with one of the first publications on the topic attributed to Turner, Clough, Martin and Topp [141] for use in structural engineering applications. It has subsequently been applied to problems in heat transfer, fluid flow, acoustics, and electromagnetics with great success and has since become one of the most widely use methods for solving engineering problems of all types. In regards to compressible fluid dynamics, the streamline upwind Petrov-Galerkin (SUPG) stabilized finite element method was developed by Hughes and Tezduyar [71] in the early 1980s to provide the numerical upwinding needed to produce non-oscillatory solutions for advection dominated flows. Further efforts yielded the Galerkin least-squares method [63], the Taylor-Galerkin method [34, 100], and more recently the discontinuous Galerkin method [13, 12, 14, 33] for compressible fluid flow.

The SUPG stabilized finite element method was chosen for solving the compressible fluid flow problems in this thesis. This chapter will discuss the salient features of the SUPG formulation for laminar compressible flows, as well as present several numerical example problems that highlight the capabilities of its implementation.

## 2.2 Navier-Stokes Equations

The Navier-Stokes equations govern the flow of a viscous fluid and represent the conservation of mass, momentum, and energy [9, 135]. The Navier-Stokes equations are often divided into two distinct forms, one governing the behavior of an incompressible fluid and the other governing the behavior of a compressible fluid. When the fluid exhibits little compressibility effects (e.g. a liquid) or the fluid is a gas with a relatively low flow velocity, it may be treated as incompressible. In the case of gas dynamics, a fluid moving at a speed roughly greater than three-tenths the speed of sound ($M_\infty > 0.3$ ) is generally treated as a compressible fluid. Most aerospace applications involve compressible flows so only the compressible form of the Navier-Stokes equations will be considered from this point forward.

The compressible flow regime is often categorized in the follow manner. Subsonic flows occur when $M_\infty < 1.0$. Transonic flows occur in the vicinity of $M_\infty = 1.0$ and contain regions of locally supersonic flow where shock waves form. Supersonic flows occur when $M_\infty > 1.0$ and the presence of shock waves is clearly evident. A hypersonic flow is categorized as one in which the commonly used ideal gas assumptions break down and high-temperature effects of the gas such as molecular vibration and chemical reactions become important. The lower Mach number bound for a hypersonic flow is generally accepted to occur around $M_\infty = 5.0$.

This thesis considers what is referred to herein as "high-speed" flows. This definition encompasses supersonic and low hypersonic flows where the formal treatment of real gas effects may be neglected.

### 2.2.1 Differential Form of the Navier-Stokes Equations

The equation governing the conservation of mass is written in differential form as

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho v_i)}{\partial x_i} = S^c \tag{2.1}$$

where $t$ and $x_i$ represent time and the spatial coordinates, $\rho$ is the density of the fluid, $v_i$ is the fluid velocity in the spatial directions, and $S^c$ accounts for any mass source terms (e.g. a mass flux

at a boundary or a mass source over a volume). In the equations presented in this chapter, index notation is implied for $i, j, k = 1, \ldots, n_{sd}$ where $n_{sd}$ represents the number of spatial dimensions. The mass conservation equation states that the time rate of change of the mass and the divergence of the mass flux must be balanced by the source terms, if any.

The momentum conservation equations are written in differential form as

$$\frac{\partial(\rho v_j)}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho v_i v_j + p\delta_{ij}\right) = \frac{\partial \tau_{ij}}{\partial x_i} + S_i^m \tag{2.2}$$

where $p$ is the pressure of the fluid, $\tau_{ij}$ is the deviatoric, shear, or viscous stress tensor, and $S_i^m$ are momentum source terms. The viscous stress tensor, which may be represented as a combination of the volumetric and deviatoric components ($\sigma_{ij} = -p\delta_{ij} + \tau_{ij}$), is split to clearly delineate the inviscid pressure term ($-p\delta_{ij}$) from the viscous stress term ($\tau_{ij}$). The conservation of momentum equation states that the time rate of change of the momentum plus the divergence of the inviscid momentum flux must be balanced by the divergence of the viscous stresses and any momentum source terms.

Conservation of energy is written in differential form as

$$\frac{\partial(\rho E)}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho E v_i + p v_i\right) = \frac{\partial(\tau_{ij} v_j)}{\partial x_i} - \frac{\partial q_i}{\partial x_i} + S^e \tag{2.3}$$

where $E$ is the total energy per unit mass, $q_i$ is the heat flux vector, and $S^e$ are any energy source terms (e.g. volumetric heat sources). This equation states that the time rate of change of the energy plus the net energy flux must be balanced by the rate of work done due to viscous forces, heat flux, and energy sources.

The $\tau_{ij}$ and $q_i$ terms in equations 2.1 – 2.3 represent the diffusive effects of the fluid. In addition to the advection transport mechanism associated with the motion of the fluid, the fluid has the ability to transport momentum and energy via a diffusion process. In the absence of any diffusion, the viscous Navier-Stokes equations reduce to the so-called inviscid Euler equations which account solely for advection. Since viscous effects are of primary concern for aerodynamic heating problems, the Euler equations will not be further discussed.

The viscous stress tensor requires a constitutive equation which relates the viscosity and spatial derivatives of the velocity to the stresses. For a Newtonian fluid (i.e. one which has a linear stress/strain relationship) the deviatoric stress tensor is often written as

$$\tau_{ij} = \mu \left( \frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) + \lambda \delta_{ij} \left( \frac{\partial v_k}{\partial x_k} \right) \tag{2.4}$$

where $\mu$ is the viscosity and $\lambda$ is the bulk viscosity of the fluid. For a Newtonian fluid the bulk viscosity is often expressed as $\lambda = -2\mu/3$.

The heat flux vector $q_i$ appearing in equation 2.3 is a measure of the thermal energy flow and is typically written using Fourier's law

$$q_i = -\kappa \frac{\partial T}{\partial x_i} \tag{2.5}$$

where $\kappa$ is the gas thermal conductivity and $T$ is the gas temperature.

### 2.2.2 Vector Form of the Navier-Stokes Equations

The conservation equations 2.1 – 2.3 are often grouped into a vector form. The Navier-Stokes equations may then be represented as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i(\boldsymbol{U})}{\partial x_i} - \frac{\partial \boldsymbol{G}_i(\boldsymbol{U})}{\partial x_i} - \boldsymbol{S} = \boldsymbol{0} \tag{2.6}$$

where $\boldsymbol{U}$ is the vector of conservative variables, $\boldsymbol{F}_i$ are inviscid fluxes, $\boldsymbol{G}_i$ are the viscous fluxes, and $\boldsymbol{S}$ is the source term vector.

The conservative variable (or state variable) vector $\boldsymbol{U}$ in equation 2.6 is written in vector form as

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho v_j \\ \rho E \end{pmatrix} \tag{2.7}$$

where $\rho$ is density of the fluid, $\rho v_j$ are the momentum variables, and $\rho E$ denotes the density times the total energy per unit mass. The total energy is the sum of the fluid's internal energy $e$ and

kinetic energy and can be written as

$$E = e + \frac{1}{2}(v_j v_j). \tag{2.8}$$

The inviscid flux vector $\boldsymbol{F}_i$ is defined as

$$\boldsymbol{F}_i(\boldsymbol{U}) = \begin{pmatrix} \rho v_i \\ \rho v_i v_j + P\delta_{ij} \\ \rho E v_i + P v_i \end{pmatrix} \tag{2.9}$$

and the viscous flux vector $\boldsymbol{G}_i$ is written by

$$\boldsymbol{G}_i(\boldsymbol{U}) = \begin{pmatrix} 0 \\ \tau_{ij} \\ \tau_{ij} v_j - q_i \end{pmatrix}. \tag{2.10}$$

The source vector $\boldsymbol{S}$ accounts for any inputs to the system such as a mass flux, gravity, or chemical reactions.

The system of equations presented in 2.6 are completed by a set of boundary conditions

$$\boldsymbol{b}(\boldsymbol{U}) = \bar{\boldsymbol{b}}(\boldsymbol{U}, \boldsymbol{x}, t) \quad \text{on} \quad \Gamma_f \tag{2.11}$$

which prescribe the values $\bar{\boldsymbol{b}}$ of a general nonlinear boundary condition $\boldsymbol{b}$ through time. Flux boundary conditions may be imposed such that

$$\boldsymbol{F}_i(\boldsymbol{U}) = \bar{\boldsymbol{F}}_i(\boldsymbol{U}, \boldsymbol{x}, t) \quad \text{on} \quad \Gamma_{\bar{f}} \tag{2.12}$$

and

$$\boldsymbol{G}_i(\boldsymbol{U}) = \bar{\boldsymbol{G}}_i(\boldsymbol{U}, \boldsymbol{x}, t) \quad \text{on} \quad \Gamma_{\bar{f}} \tag{2.13}$$

Additionally, the state values $\boldsymbol{U}$ must be specified at each point $\boldsymbol{x}$ as initial conditions at $t = 0$

$$\boldsymbol{U}(\boldsymbol{x}, t = 0) = \boldsymbol{U}_0(\boldsymbol{x}) \quad \text{in} \quad \Omega_f \tag{2.14}$$

Equation 2.6 is also often written in quasi-linear form as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \mathbf{A}_i \frac{\partial \boldsymbol{U}}{\partial x_i} - \frac{\partial}{\partial x_i}\left(\mathbf{K}_{ij} \frac{\partial \boldsymbol{U}}{\partial x_j}\right) - \boldsymbol{S} = \mathbf{0} \tag{2.15}$$

where $\mathbf{A}_i$ are the inviscid flux Jacobian matrices, which are expressed as

$$\mathbf{A}_i = \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{U}} \tag{2.16}$$

and $\mathbf{K}_{ij}$ are diffusion matrices accounting for the effects of viscosity and thermal conductivity, hence the viscous fluxes may be written by

$$\boldsymbol{G}_i = \mathbf{K}_{ij} \frac{\partial \boldsymbol{U}}{\partial x_j}. \tag{2.17}$$

For convenience, equation 2.15 may written compactly as the product of a quasi-linear operator $\boldsymbol{\mathcal{L}}$ and the state vector $\boldsymbol{U}$ minus any source terms $\boldsymbol{S}$ as

$$\boldsymbol{\mathcal{L}U} - \boldsymbol{S} = \mathbf{0} \tag{2.18}$$

where the operator $\boldsymbol{\mathcal{L}}$ is defined as

$$\boldsymbol{\mathcal{L}} \equiv \underbrace{\frac{\partial}{\partial t}}_{\boldsymbol{\mathcal{L}_t}} + \underbrace{\mathbf{A}_i \frac{\partial}{\partial x_i}}_{\boldsymbol{\mathcal{L}_{adv}}} - \underbrace{\frac{\partial}{\partial x_i}\left(\mathbf{K}_{ij}\frac{\partial}{\partial x_j}\right)}_{\boldsymbol{\mathcal{L}_{diff}}}. \tag{2.19}$$

and $\boldsymbol{\mathcal{L}}_t$, $\boldsymbol{\mathcal{L}}_{adv}$, and $\boldsymbol{\mathcal{L}}_{diff}$ represent the temporal, advection, and diffusion operators.

**State Variables**

Many choices of state variables are possible for solving the Navier-Stokes equations (e.g. conservation variables, primitive variables, entropy variables) and the performance of each set of variables depends on the physics of the flow. For compressible flows, the conservation variables are the standard variable of choice in the finite volume and finite difference communities, however, within the finite element literature the choice of variables for solving compressible flow problems is split between conservation and entropy variables. Hauke [53] compares the relative merits of each choice of state variable vectors and draws conclusions on the effectiveness of each set. Shakib [124] successfully employed entropy variables for compressible flow applications ranging from subsonic to supersonic while Chalot [28] demonstrated the effectiveness of entropy variables for equilibrium chemically reacting flow problems. Le Beau [15] showed that conservation and entropy variables produce results with indistinguishable differences for inviscid flows. Aliabadi [3] was among the

first to demonstrate the use of conservation variables for viscous laminar flows. In this work, conservation variables have been chosen and will be used throughout.

### 2.2.3 Boundary Conditions for Supersonic Viscous Flow Problems

Appropriate treatment of the boundary conditions for supersonic compressible flows is needed to uniquely define the problem and its solution. For external viscous flow problems the domain boundaries can be divided into three distinct regions: the inflow, wall, and outflow boundary. Each of the boundary condition types will be considered next.

**Supersonic Inflow Boundary** A supersonic inflow boundary specifies all components of the state vector $\boldsymbol{U}$ as Dirichlet boundary conditions. The mathematical reasoning for this is nicely described by Donea [35] and has to do with the fact that all the characteristic lines of the flow (for supersonic flow the characteristic lines are Mach lines) are entering the domain at a supersonic inflow boundary.

**Wall Boundary**

The wall boundary for a viscous flow problem may be divided into two different type based on thermodynamic considerations: an adiabatic wall and an isothermal wall. Common amongst these two types is the treatment of the momentum equations – the wall is treated in such a way that the velocity and hence the momentum at the wall is prescribed. In the case the wall is not moving, the velocity is prescribed to be zero. However, the wall could be moving as in the case of aeroelasticity, in which case each point on the wall takes on a value equal to the local velocity vector. If the wall is adiabatic, the heat flux normal to the wall ($q_i \cdot n_i$) is zero. If the wall is isothermal, a constant temperature of the wall $T_w$ is prescribed. This work chooses to use a Lagrange multiplier based constraint equation method to strongly impose the wall temperature. Lagrange multipliers are an elegant way to apply nonlinear boundary conditions; in the case of an isothermal wall, the temperature at the wall is a function of the conservative state variables and is simply handled by constraining the total energy to be $\rho E = \rho c_v T_w + 0.5\rho(v_i v_i)$. Details of such an approach may be found in [6].

**Supersonic Outflow Boundary** As described by Donea [35], for a supersonic outflow boundary all of the characteristics of the flow point of the domain. Consequently, no Dirichlet boundary conditions are given here. A boundary integration my arise from a discretization such as the Galerkin finite element method when the divergence of the viscous flux term ($\partial \boldsymbol{G}_i/\partial x_i$) is integrated by parts. In such a case, performing this boundary integration is necessary as shown by Shakib [125].

### 2.2.4 Transport Coefficient Models

The viscous stress tensor $\tau_{ij}$ and heat flux vector $q_i$ rely on transport coefficients the determine the rate of the diffusion process. The viscosity coefficient for a gas is a macroscopic approximation of momentum transport within the flow as a result of molecular diffusion. Several models for the viscosity of a gas exist, with the most common probably being Sutherland's law [123]. The Sutherland formula is written in two coefficient form as

$$\mu(T) = \mu_{ref} \frac{T^{3/2}}{T + T_{ref}}. \tag{2.20}$$

For air at temperatures below roughly $1000K$ and pressures below around $1 \times 10^6\ N/m^2$, valid reference values are $\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110.4\ K$. The Sutherland formula may also be written in a three coefficient form as

$$\mu(T) = \mu_{ref}\ \left(\frac{T}{T_{ref}}\right)^{3/2} \frac{T_{ref} + S}{T + S} \tag{2.21}$$

where $\mu_{ref} = 1.716 \times 10^{-5}\ kg/m \cdot s$, $T_{ref} = 273.11\ K$ and $S = 110.56\ K$. Additionally, Keyes law may also be used to compute the viscosity

$$\mu(T) = a_0 \frac{T^{3/2}}{T + 10a^{\frac{-a_1}{T}}} \tag{2.22}$$

where for air the SI unit viscosity model constants are: $a = 5$, $a_0 = 1.488 \times 10^{-6}$, and $a_1 = 122.1$. Since the viscosity of a gas is derived from kinetic theory, a viscosity model directly related to the molecular kinetics is also possible via the formula

$$\mu(T) = 2.67 \times 10^{-6} \frac{\sqrt{M_w T}}{\sigma^2 \Omega_\mu} \tag{2.23}$$

where $M_w$ is the molecular weight, $\Omega_\mu = \Omega_\mu(T^*)$, $T^* = T/(\epsilon/k_B)$, and $\sigma$ and $\epsilon/k_B$ are the so-called Lennard-Jones parameters. The reader is referred to reference [77] for more details on the use of this viscosity model.

The coefficient of thermal conductivity needed for the heat flux computation is a measure of the energy transport resulting from molecular collisions. The thermal conductivity of a gas is often modeled as a relation of the Prandtl number and viscosity by the equation

$$\kappa = \frac{c_p \mu}{Pr} \tag{2.24}$$

where $c_p$ is the specific heat of the gas at constant pressure, $Pr$ is the Prandtl number and $\mu$ is the viscosity of the fluid. The Prandtl number is the ratio of the viscous diffusion rate to the thermal diffusion rate and for laminar flow of air at moderate temperatures the Prandtl number is assumed to be constant and equal to approximately 0.71. Since thermal conductivity is also based on kinetic theory, a kinetic theory model is also possible. One such kinetic theory model is given via the equation

$$\kappa = \frac{15}{4} \frac{R}{M_w} \mu \left( \frac{4}{15} \frac{c_p M_w}{R} + \frac{1}{3} \right) \tag{2.25}$$

where $M_w$ is the molecular weight and $R$ is a gas constant. Further details on use of this thermal conductivity model may be found in reference [77].

### 2.2.5 Gas Models

The compressible Navier-Stokes equations are closed by a fluid model that describes the thermodynamic behavior of the fluid; this model is often referred to as an equation of state. There are a number gas models including the ideal gas model, equilibrium real gas models, the frozen real gas models, and non-equilibrium real gas models. The equations of state associated with each of these models provide a thermodynamic relation between the density, pressure, internal energy, enthalpy, and temperature of the gas. For the purposes of this thesis, only the ideal gas model is considered. However, the real gas models are discussed here because they are important for

understanding the hypersonic gas flow regime and the validity of the ideal gas model for high Mach number flows.

***Ideal Gas Model***

The precise definition of an ideal gas varies throughout the literature and textbooks; here an ideal gas is defined as a calorically perfect gas. The perfect gas assumption implies that intermolecular (e.g. Van der Waals) forces are negligible. Thus, a perfect gas may be modeled by the perfect gas equation of state

$$p = \rho R T \tag{2.26}$$

where $R$ is a constant specific to the type of gas (for air $R = 287.1\ J/kg/K$). The calorically perfect gas assumption adds the following requirements: (1) the gas is in thermal equilibrium, (2) the gas is not chemically reacting, (3) the specific heats ($c_v$ and $c_p$) are constant, and (4) the internal energy and enthalpy are dependent only on temperature . In accordance with these assumptions, the specific heats may be written as

$$c_v = \frac{R}{\gamma - 1} \quad , \quad c_p = \frac{\gamma R}{\gamma - 1} \tag{2.27}$$

where $\gamma$ is the ratio of specific heats (for air $\gamma = 1.4$) and is expressed as

$$\gamma = \frac{c_p}{c_v}, \tag{2.28}$$

and the internal energy and enthalpy are computed by the equations

$$e(T) = c_v T \quad , \quad h(T) = c_p T\ . \tag{2.29}$$

An alternative but equivalent form of the perfect gas equation of state can be obtained by writing the temperature as $T = e(\gamma - 1)/R$. Inserting this temperature expression into equation 2.26 we obtain the following form of the ideal gas equation

$$p = (\gamma - 1)\rho e \tag{2.30}$$

The ideal gas assumptions begin to break down when the temperature of the fluid reaches roughly 600 $K$. The differences between an ideal gas model and a real gas model become important

(on the order of 15 - 25 %) around 2000 $K$. Above this, an ideal gas approximation will significantly over-predict gas temperatures. It is noted that this limit usually coincides with a Mach number of approximately 5, which is generally considered the hypersonic limit where chemical reactions and thermal non-equilibrium begin to occur. Chalot [28] provides a nice motivation for the necessity of incorporating real gas effects for hypersonic flows.

***Thermochemical Equilibrium Real Gas Model***

As the gas temperature increases (above roughly 800 $K$ for air), the calorically perfect assumption is no longer valid as gas molecules become vibrationally excited. The vibrational excitation means that the specific heats are no longer constant but now also a function of temperature. At this point the gas is referred to as thermally perfect. At even higher temperatures (above 2000 $K$ for air), the thermally perfect gas assumption is no longer valid as the molecules of the gas will start to dissociate and individual gas species will begin to chemically react. The equilibrium assumption of the gas means that the chemical reactions are taken to occur instantaneously. Assuming the gas is not at a state of high pressure and low temperature, the intermolecular forces of the chemical species may be ignored and the gas may be treated as a mixture of thermally perfect gases. As such, the mixture density is written as

$$\rho = \sum_{s=1}^{n_s} \rho_s \tag{2.31}$$

where the $s$ subscript represents an individual species. Following this, the mass conservation equation (2.1) must take into account the density of the individual species and is modified to become

$$\frac{\partial \rho_s}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho_s v_i - \rho D_s \frac{\partial \chi_s}{\partial x_i}\right) = \dot{\omega}_s \tag{2.32}$$

where $D_s$ are the species diffusion coefficients, $\chi_s = \rho_s/\rho$ are the species mass fractions, and $\dot{\omega}_s$ are the species reaction rates which go to $\infty$ under equilibrium conditions. The equation for conservation of energy must also be re-written to account for the mass diffusion, hence equation 2.3 becomes

$$\frac{\partial(\rho E)}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho E v_i + P v_i\right) = \frac{\partial(\tau_{ij} v_j)}{\partial x_i} - \frac{\partial q_i}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\rho \sum_{s=1}^{n_s} h_s D_s \frac{\partial \chi_s}{\partial x_i}\right) + S^e \tag{2.33}$$

The equation of state for the mixture of thermally perfect gases is written as

$$P = \sum_{s=1}^{n_s} \rho_s R_s T \tag{2.34}$$

The internal energy and enthalpy are now computed by the equations

$$e(T) = \sum_{s=1}^{n_s} \chi_s e_s^0 + \sum_{s=1}^{n_s} \chi_s c_{v_s}(T) T \quad , \quad h(T) = \sum_{s=1}^{n_s} \chi_s h_s^0 + \sum_{s=1}^{n_s} \chi_s c_{p_s}(T) T \ . \tag{2.35}$$

where $e_s^0$ and $h_s^0$ are formation energies for individual species. For the thermochemical equilibrium assumption, it is assumed that the vibrational modes of the molecule are at a steady-state, and the energy quantities in equation 2.35 are computed by a single temperature.

***Thermochemical Non-equilibrium Real Gas Model***

The case of thermochemical non-equilibrium assumes the time scales associated with the molecular vibration modes and chemical reactions are much less than that of the fluid flow. Non-equilibrium of the chemical reactions implies that they no longer occur at an instantaneous rate but instead at a finite-rate while the gas is being advected. This means that finite values of $\dot{\omega}_s$ must be used for the mixture mass conservation equation (2.32), otherwise equations 2.31 through 2.34 are used to model chemical non-equilibrium effects.

Thermal non-equilibrium occurs when the translational, rotational, and vibrational modes of a molecule are excited at different temperatures. A two-temperature energy model is often assumed where the translational and rotational modes occur at one temperature $T$ and the vibrational mode occurs at another temperature $T_v$. Accordingly, the internal energy may now be written as

$$e(T, T_v) = \sum_{s=1}^{n_s} \chi_s e_s^0 + \sum_{s=1}^{n_s} \chi_s c_{v_s}^{tr}(T) T + e_v \tag{2.36}$$

where $c_{v_s}$ is the translational/rotational temperature dependent specific heat of specie $s$ and $e_v$ is computed by solving an additional energy equation. Additional information on the two-temperature energy model can be found in reference [114, 115].

## 2.3 Finite Element Discretization of the Navier-Stokes Equations

Solving the Navier-Stokes system posed by equation 2.15 is hardly a trivial task. As mentioned in the introduction to this chapter, standard finite difference, finite volume, and finite element methods typically result in a central difference scheme that produces non-physical oscillations in the solution for advection dominated flows. A well practiced cure for this problem is the so-called "upwinding" technique, whereby the difference scheme is weighted more heavily in the upwind direction. This section discusses the details of the streamline-upwind Petrov-Galerkin (SUPG) finite element method used to solve the Navier-Stokes equations.

### 2.3.1 Galerkin Formulation

The standard Galerkin discretization proceeds by dividing the domain of interest into elements, weighting the strong form of the residual equation (2.15) with element test functions and integrating over the domain. This produces the following Galerkin weak statement:

$$\int_{\Omega_f} \boldsymbol{W} \cdot \left[ \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - \frac{\partial \boldsymbol{G}_i}{\partial x_i} - \boldsymbol{S} \right] d\Omega = \mathbf{0}. \tag{2.37}$$

where $\boldsymbol{W}$ are the set test functions and the dot operator implies a scalar product. The viscous term in equation 2.37 involves a second-order spatial derivative, so it is commonly integrated by parts to reduce its order. Consequently, equation 2.37 may now be re-written as

$$\int_{\Omega_f} \boldsymbol{W} \cdot \left( \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - \boldsymbol{S} \right) + \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \boldsymbol{G}_i \; d\Omega - \int_{\Gamma_f} \boldsymbol{W} \cdot \boldsymbol{G}_i \hat{n}_i \; d\Gamma = \mathbf{0} \tag{2.38}$$

which is the familiar Galerkin weak form which includes a boundary integration of the viscous flux terms $\boldsymbol{G}_i$.

### 2.3.2 Streamline Upwind Petrov-Galerkin Formulation

Among the first occurrences of the upwinding technique being used in the context of finite element methods can be traced to work by Tabata [134]. However, the first generally recognized success is attributed to the seminal paper by Brooks and Hughes [25] in which the SUPG method

was introduced. While the first applications of SUPG were for solving the incompressible Navier-Stokes equations, Hughes and Tezduyar applied the method to the compressible Euler equations in the early 1980s [137, 71] and later to the compressible Navier-Stokes equations. Hughes et al. [64, 69, 67, 68, 62, 65, 61, 63, 124, 125] then later generalized the SUPG method and called it the Galerkin least-squares (GLS) method.

The fundamental concept of the SUPG method is to add an upwind bias to the standard test functions $\boldsymbol{W}$. The SUPG method achieves this by using a modified test function written as

$$\hat{\boldsymbol{W}} = \boldsymbol{W} + \frac{\partial \boldsymbol{W}}{\partial x_k} \mathbf{A}_k \boldsymbol{\tau}_{supg} \ . \tag{2.39}$$

This amounts to adding a perturbation to the test functions $\boldsymbol{W}$ that is proportional to the test function gradient ($\partial \boldsymbol{W}/\partial x_i$) and in the upwind direction (via the inviscid flux derivatives contained in the $\mathbf{A}_i$ matrices). The upwind perturbation is then appropriately scaled through the stabilization parameter $\boldsymbol{\tau}_{supg}$. The specifics of computing $\boldsymbol{\tau}_{supg}$ are deferred to a later section. By substituting the SUPG modified test function (2.39) into equation 2.38 we arrive at the SUPG weak form

$$\int_{\Omega_f} \boldsymbol{W} \cdot \left( \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - \boldsymbol{S} \right) + \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \boldsymbol{G}_i \ d\Omega - \int_{\Gamma_f} \boldsymbol{W} \cdot \boldsymbol{G}_i \hat{n}_i \ d\Gamma + \sum_{e=1}^{n_e} \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}}{\partial x_k} \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \left[ \frac{\partial \boldsymbol{U}}{\partial t} + \mathbf{A}_i \frac{\partial \boldsymbol{U}}{\partial x_i} - \frac{\partial}{\partial x_i} \left( \mathbf{K}_{ij} \frac{\partial \boldsymbol{U}}{\partial x_j} \right) - \boldsymbol{S} \right] d\Omega = \mathbf{0} \ . \tag{2.40}$$

In this equation, the summation over the elements $e = 1, ..., n_e$ implies that the stabilization occurs only over the element interior. As such it applies to the strong form of the Navier-Stokes residual equation (2.15). Equation 2.39 may also be represented by the more compact operator notation as

$$\int_{\Omega} \boldsymbol{W} \cdot \left( \frac{\partial \boldsymbol{U}}{\partial t} + \mathbf{A}_i \frac{\partial \boldsymbol{U}}{\partial x_i} - \boldsymbol{S} \right) + \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \left( \mathbf{K}_{ij} \frac{\partial \boldsymbol{U}}{\partial x_j} \right) d\Omega - \int_{\Gamma} \boldsymbol{W} \cdot \boldsymbol{G}_i \hat{n}_i \ d\Gamma + \sum_{e=1}^{n_e} \int_{\Omega^e} \boldsymbol{\mathcal{L}}_{adv}^T \boldsymbol{W} \ \boldsymbol{\tau}_{supg} \cdot (\boldsymbol{\mathcal{L}} \boldsymbol{U} - \boldsymbol{S}) \, d\Omega = \mathbf{0} \ . \tag{2.41}$$

The GLS method is obtained by replacing the $\boldsymbol{\mathcal{L}}_{adv}^T \boldsymbol{W}$ term in the previous equation by the full Navier-Stokes operator acting on the test functions $\boldsymbol{\mathcal{L}}^T \boldsymbol{W}$. For the purposes of compressible gas dynamics, the SUPG stabilization has proven to be successful [4, 5] and the GLS method will not be considered any further.

In the case of supersonic gas dynamics, shock waves prove particularly difficult to handle by a numerical scheme because they exhibit steep gradients in the solution and lead to numerical oscillations. Shock capturing methods are often introduced into equation 2.41 that have the effect of adding numerical dissipation in the vicinity of the shock. Thus, we arrive at a new form of the SUPG stabilized weak statement with a dissipative shock capturing term added

$$\int_{\Omega_f} \boldsymbol{W} \cdot \left( \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - \boldsymbol{S} \right) + \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \boldsymbol{G}_i \; d\Omega - \int_{\Gamma_f} \boldsymbol{W} \cdot \boldsymbol{G}_i \hat{n_i} \; d\Gamma + \sum_{e=1}^{n_e} \int_{\Omega_f^e} \boldsymbol{\mathcal{L}}_{adv}^T \boldsymbol{W} \boldsymbol{\tau}_{supg} \cdot (\boldsymbol{\mathcal{L}} \boldsymbol{U} - \boldsymbol{S}) \, d\Omega + \sum_{e=1}^{n_e} \int_{\Omega_f^e} \delta \left( \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{U}}{\partial x_i} \right) d\Omega = \boldsymbol{0} \; . \quad (2.42)$$

where $\delta$ is a shock capturing parameter which scales the dissipation term in the vicinity of a discontinuity in the solution. Computation of shock capturing parameters are discussed later in section 2.3.5.

Equation 2.42 represents the standard SUPG formulation for compressible Navier-Stokes as found in much of the literature. More recently, several researchers have introduced modifications to the standard SUPG treatment that improve its accuracy and robustness especially for high-speed and hypersonic flows. These improvements will be discussed in further detail.

### 2.3.3 Finite Element Spatial Approximation

As briefly mentioned in section 2.3.1, the spatial discretization by the finite element method proceeds as follows: the domain $\Omega$ is divided into $n_e$ elements $\Omega^e$ and we select a suitable trial solution space $\boldsymbol{\mathcal{S}}^h$ and test function space $\boldsymbol{\mathcal{V}}^h$ for $\boldsymbol{U}^h$ and $\boldsymbol{W}^h$, respectively. The trial solution space is typically defined in the following manner:

$$\boldsymbol{\mathcal{S}}^h = \left[ \boldsymbol{U}^h \in \left[ C^0(\Omega) \right]^{n_{dof}} \; , \; \boldsymbol{U}^h \, |_{\Omega^e} \in \left[ P^k(\Omega^e) \right]^{n_{dof}} \; , \; \boldsymbol{b}(\boldsymbol{U}^h) = \bar{\boldsymbol{b}}(\boldsymbol{U}^h, \boldsymbol{x}, t) \text{ on } \Gamma_f \right] \quad (2.43)$$

which states that the trial solution must be $C^0$ continuous (a polynomial function on any side of an element is completely specified by the the degrees-of-freedom on that side), representable by an interpolation polynomial $P^k$ of order $k$, and must satisfy the prescribed boundary conditions.

Similarly, the test function space is defined by

$$\boldsymbol{\mathcal{V}}^h = \left[ \boldsymbol{W}^h \in \left[ C^0(\Omega) \right]^{n_{dof}} \ , \ \boldsymbol{W}^h \mid_{\Omega^e} \in \left[ P^k(\Omega^e) \right]^{n_{dof}} \ , \ \boldsymbol{b}(\boldsymbol{W}^h) = \boldsymbol{0} \text{ on } \Gamma_f \right] \ . \tag{2.44}$$

The definition of the test function space is similar to equation 2.43 only that the values of the test functions must be zero on the boundary. An appropriate choice of interpolation polynomial order $P^k$ is often dependent on the behavior of the partial differential equation being solved. For the smooth solution behavior of elliptic PDEs (such as in elasticity) and parabolic PDEs (as in transient heat transfer) a higher order test function polynomial will often lead to more accurate solutions. However, if the PDE is hyperbolic in nature and exhibits discontinuities in the solution, a linear test function polynomial is often the best choice as shock capturing methods reduce the solution to first-order accuracy at the discontinuity. Hence, the so-called "h-refinement" practice of introducing more elements of a given basis rather than the "p-refinement" practice of increasing the interpolation order is the best way of obtaining a more accurate solution for a shock dominated flow. The finite element implementation of this work permits the use of linear or quadratic element bases, however, in practice only linear basis functions are used for the supersonic flow solutions contained in this thesis.

Given these function spaces, the solution can be represented at any spatial point $\boldsymbol{x}$ in time through an interpolation from the nodal solution values using the finite element shape/basis functions $\boldsymbol{N}$

$$\boldsymbol{U}^h(\boldsymbol{x}, t) = \sum_{n=1}^{n_n} N^n(\boldsymbol{x}) \cdot \boldsymbol{U}^n(t) \ . \tag{2.45}$$

In addition to interpolating the solution variables, other nonlinear function quantities such as the flux terms (equations 2.9 and 2.10 ) need to be interpolated for the purposes of performing numerical integration over the element. The idea of interpolating inviscid flux terms from nodal quantities was suggested in an SUPG context can be traced back to reference [71]. More recently, Kirk [85] showed that interpolating the inviscid fluxes from nodal values leads to better stability and accuracy properties in an SUPG context. The nodal inviscid flux interpolation is the expressed

via

$$\boldsymbol{F}_i^h(\boldsymbol{x},t) = \sum_{n=1}^{n_n} N^n(\boldsymbol{x}) \cdot \boldsymbol{F}_i^n(\boldsymbol{U},t). \tag{2.46}$$

This is opposed to computing the fluxes directly at the nodes using interpolated state values $\boldsymbol{U}^h$. Continuing along this same line of reasoning, the work contained in this thesis also computes any nonlinear quantity that does not depend directly on spatial derivatives (velocity, internal energy $e$, temperature, pressure, viscosity, and thermal conductivity) at the nodes and interpolates the nodal values to the integration points. This is opposed to computing the quantity at the integration point from interpolated values of the conservative variables $\boldsymbol{U}^h$. In the case of elements with linear basis functions, the degrees of freedom are represented in a piecewise linear fashion; likewise, the scheme is arguably more stable if any quantities which are derived from the degrees of freedom are also piecewise linear. It is noted that the viscous flux terms, which depend on spatial gradients of the solution, are interpolated from nodal state values in the usual finite element fashion using shape function derivatives.

### 2.3.4 SUPG Stabilization Parameter

The stabilization parameter $\boldsymbol{\tau}_{supg}$ found in equation 2.39 is used to scale the amount of stabilization needed without making the solution overly or underly diffuse. The values in $\boldsymbol{\tau}_{supg}$ are often referred to as intrinsic time scales that adapts the upwinding provided by the $\partial \boldsymbol{W}/\partial x_i \, \mathbf{A}_i$ term accordingly. Several forms of the stabilization parameter have been devised, which are split here into spatially dependent and equation specific stabilization parameters.

**Spatially Dependent $\boldsymbol{\tau}_{supg}$** One of the earliest and simplest stabilization parameters was introduced by Hughes [71] where a separate $\tau$ is computed for each spatial direction

$$\tau_i^s = \frac{\alpha h_i}{a_i} \tag{2.47}$$

where $\alpha$ is a parameter related to the accuracy of the time integrator ($\alpha = 1$ for first-order accurate solutions and $\alpha = 1/2$ for second-order accurate solutions), $h_i$ are element length scales, and $a_i$ are the spectral radii of the flux Jacobian matrices $\mathbf{A}_i$. The spectral radii are computed by finding the

maximum eigenvalue of the individual $\mathbf{A}_i$ matrices; the maximum eigenvalue is computed as $c + v_i$ where $c$ is the local speed of sound. The perturbed weight functions are then computed by

$$\hat{\boldsymbol{W}} = \boldsymbol{W} + \frac{\partial \boldsymbol{W}}{\partial x_i} \mathbf{A}_i \tau_i^s \ . \tag{2.48}$$

A single stabilization parameter may be found by taking the maximum of the individual $\tau_i$ values, as is done by Le Beau [15]. There are numerous ways to compute the element length scale $h_i$. Hughes suggests computing it via

$$h_i = \mathcal{C} \left[ \left( \frac{\partial x_i}{\partial \xi} \right)^2 + \left( \frac{\partial x_i}{\partial \eta} \right)^2 + \left( \frac{\partial x_i}{\partial \zeta} \right)^2 \right]^{1/2} \tag{2.49}$$

for three-dimensional hexahedral (quadrilateral in two-dimensions) elements and via

$$h_i = \mathcal{C} \left[ \left( \frac{\partial x_i}{\partial \zeta_1} \right)^2 + \left( \frac{\partial x_i}{\partial \zeta_2} \right)^2 + \left( \frac{\partial x_i}{\partial \zeta_3} \right)^2 + \left( \frac{\partial x_i}{\partial \zeta_4} \right)^2 \right]^{1/2} \tag{2.50}$$

for three-dimensional tetrahedral (triangular in two-dimensions) elements. $\mathcal{C}$ in the previous two equations is the parametric length of the element, thus for quadrilateral/hexahedral elements $\mathcal{C} = 2$ and for triangular/tetrahedral elements $\mathcal{C} = 1$. Le Beau [15] simply computes the $h_i$ values as the length of the element in the $x_i$ direction.

**Equation Dependent $\boldsymbol{\tau}_{supg}$** It is also possible to compute a separate scalar stabilization parameter for each conservation equation. Using this approach the $\boldsymbol{\tau}_{supg}$ in equation 2.39 is represented as a diagonal matrix; the three dimensional form is written as

$$\boldsymbol{\tau}_{supg} = \text{diag}\left(\tau_c, \tau_m, \tau_m, \tau_m, \tau_e\right) \tag{2.51}$$

Tezduyar [138] proposed some of the original forms for computing these individual $\tau$ terms as

$$\tau_c = \left[ \left( \frac{1}{h_{\boldsymbol{v}}} \right)^r + \left( \frac{2}{\Delta t} \right)^r \right]^{-1/r} \tag{2.52}$$

$$\tau_m = \left[ \left( \frac{1}{h_{\boldsymbol{v}}} \right)^r + \left( \frac{2}{\Delta t} \right)^r + \left( \frac{4\mu}{\rho h_m} \right)^r \right]^{-1/r} \tag{2.53}$$

$$\tau_e = \left[ \left( \frac{1}{h_{\boldsymbol{v}}} \right)^r + \left( \frac{2}{\Delta t} \right)^r + \left( \frac{4\nu_e}{h_e} \right)^r \right]^{-1/r} \tag{2.54}$$

where the three individual length scales $h_{\boldsymbol{v}}$, $h_m$, and $h_e$ are defined as

$$h_{\boldsymbol{v}} = \left( \sum_{n=1}^{n_n} \mid \boldsymbol{v} \cdot \nabla N_n \mid \right)^{-1} \tag{2.55}$$

$$h_m = 2 \left( \sum_{n=1}^{n_n} \mid \boldsymbol{r}_m \cdot \nabla N^n \mid \right)^{-1} \quad \text{where} \quad \boldsymbol{r}_m = \frac{\nabla \|\boldsymbol{v}\|}{\|\nabla \|\boldsymbol{v}\|\|} \tag{2.56}$$

$$h_e = 2 \left( \sum_{n=1}^{n_n} \mid \boldsymbol{r}_e \cdot \nabla N^n \mid \right)^{-1} \quad \text{where} \quad \boldsymbol{r}_e = \frac{\nabla T}{\|\nabla T\|} \ , \tag{2.57}$$

$\nu_e$ is the "kinematic viscosity" for the energy equation (which is often represented as $\rho c_p/\kappa$). Equations 2.52 – 2.54 are written using the "$r - switch$" inverse norm which allows for a smooth variation between its different contributions. Typically $r = 2$.

More recently, Bova [23] has proposed an alternate form for $\tau_c$, $\tau_m$, $\tau_e$ which are written as

$$\tau_c = \left[ \left( \frac{\|\boldsymbol{v}\| + c}{h_{\boldsymbol{v}}} \right)^2 + (dcp)^2 \right]^{-1/2} \tag{2.58}$$

$$\tau_m = \left[ \left( \frac{\|\boldsymbol{v}\| + c}{h_{\boldsymbol{v}}} \right)^2 + (dcp)^2 + \frac{\mu}{\rho h_{\boldsymbol{v}}^2} \right]^{-1/2} \tag{2.59}$$

$$\tau_e = \left[ \left( \frac{\|\boldsymbol{v}\| + c}{h_{\boldsymbol{v}}} \right)^2 + (dcp)^2 + \frac{\kappa}{\rho c_p h_{\boldsymbol{v}}^2} \right]^{-1/2} \tag{2.60}$$

where $dcp$ represents the discontinuity capturing parameter (typically $\delta$ or $\nu$) which is discussed in the next section. Kirk [85] defined the flow-aligned length scale $h_{\boldsymbol{v}}$ similar to Tezduyar's but as

$$h_{\boldsymbol{v}} = \left( \sum_{n=1}^{n_n} \mid \hat{\boldsymbol{v}} \cdot \nabla N^n \mid \right)^{-1} \tag{2.61}$$

where $\hat{\boldsymbol{v}}$ is the unit velocity vector. Bova uses a different definition of $h_{\boldsymbol{v}}$ based the discussion in reference [11]

$$h_{\boldsymbol{v}} = \mathcal{C} \sqrt{\frac{v_k v_k}{v_i g_{ij} v_j}} \tag{2.62}$$

where $g_{ij}$ is the inverse metric tensor defined by

$$g_{ij} = \frac{\partial \xi_k}{\partial x_j} \frac{\partial \xi_k}{\partial x_i} \tag{2.63}$$

**Nodal $\boldsymbol{\tau}_{supg}$** There are several choices of where to evaluate the $\boldsymbol{\tau}_{supg}$ term in equation 2.42. The classical approach is to evaluate this term directly at the integration points. However, as

noted by Bova [23], the flow-aligned length scale $h_{\boldsymbol{v}}$ may involve element specific quantities (such as $g_{ij}$ in the case of equation 2.62) which result in a discontinuous stabilization field across the elements that support a given node. This means that the upwinding may be inconsistently applied and recent work has suggested that a scheme will exhibit improved stability if the stabilization parameter associated with a node is constant amongst its patch of elements [11].

Since all of the terms in equations 2.39 and 2.58–2.60 can be computed directly at the nodes except for $h_{\boldsymbol{v}}$, this quantity must be computed in a nodally averaged sense from element quantities. Bova performs a global $L-2$ projection to obtain $h_{\boldsymbol{v}}$ at the nodes. The work presented here uses a local projection technique. The nodal averaging procedure implemented here progresses as follows:

(1) Compute $h_{\boldsymbol{v}}$ quantities at the integration points as a pre-processing step before any element integration begins

(2) Project integration point values of $h_{\boldsymbol{v}}$ to the nodes using a local $L-2$ projection or extrapolation operator and sum the nodal values

(3) Compute a nodal $h_{\boldsymbol{v}}$ by averaging the $h_{\boldsymbol{v}}$ sum amongst the patch of elements that support that node

(4) Before the integration of an individual finite element begins compute $\boldsymbol{\tau}_{supg}$ at each node using the nodally averaged $h_{\boldsymbol{v}}$

(5) Interpolate the nodal values of $\boldsymbol{\tau}_{supg}$ to the integration points during the finite element integration of equation 2.42

The code developed for this thesis has implemented both the "classic" integration point evaluation of $\boldsymbol{\tau}_{supg}$ and the nodally reconstructed version of $\boldsymbol{\tau}_{supg}$. Experience has shown that the nodally reconstructed stabilization parameter provides a modest improvements in stability and robustness.

### 2.3.5 Discontinuity Capturing Parameter

As mentioned in section 2.3.2, the discontinuity capturing parameter $\delta$ in equation 2.42 is responsible for adding numerical dissipation in order to limit steep gradients and solution oscillation in the presence of a shock wave. Several forms of the discontinuity capturing operator are discussed in this section as well as a few improvements devised by Kirk and Bova [23].

**$\delta$ Discontinuity Capturing Operator** The form of the discontinuity capturing operator shown in equation 2.42 was originally listed in a 1991 paper by Le Beau [16]. Repeated here for clarity, this form of the operator reads

$$\sum_{e=1}^{n_e} \int_{\Omega_e} \delta \left( \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{x}_i} \cdot \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{x}_i} \right) d\Omega \tag{2.64}$$

The calculation of the $\delta$ parameter, as used by Kirk [85], is written in its three-dimensional form as

$$\delta_{91} = \left[ \frac{\left\| \frac{\partial \boldsymbol{U}}{\partial t} + \mathbf{A}_i \frac{\partial \boldsymbol{U}}{\partial x_i} - \frac{\partial}{\partial x_i} \left( \mathbf{K}_{ij} \frac{\partial \boldsymbol{U}}{\partial x_j} \right) \right\|_{\mathbf{A}_0^{-1}}}{\|\nabla \xi \cdot \nabla \boldsymbol{U}\|_{\mathbf{A}_0^{-1}} + \|\nabla \eta \cdot \nabla \boldsymbol{U}\|_{\mathbf{A}_0^{-1}} + \|\nabla \zeta \cdot \nabla \boldsymbol{U}\|_{\mathbf{A}_0^{-1}}} \right]^{1/2} \tag{2.65}$$

The $\mathbf{A}_0^{-1}$ term in this equation is a matrix which transforms entropy state variables to conservation state variables.

**$\nu$ Discontinuity Capturing Operator** Another form of the discontinuity capturing operator, originally due to Hughes and Mallet [68] may be expressed as

$$\sum_{e=1}^{n_e} \int_{\Omega_f^e} \nu \left( \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{x_i}} \cdot g^{ij} \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{x_i}} \right) d\Omega \tag{2.66}$$

where

$$\nu = \left[ \frac{\left\| \frac{\partial \boldsymbol{U}}{\partial t} + \mathbf{A}_i \frac{\partial \boldsymbol{U}}{\partial x_i} - \frac{\partial}{\partial x_i} \left( \mathbf{K}_{ij} \frac{\partial \boldsymbol{U}}{\partial x_j} \right) \right\|_{\boldsymbol{A_0^{-1}}}}{\|\Delta \boldsymbol{U}\|_{\boldsymbol{A_0^{-1}}} + g^{ij} \frac{\partial \boldsymbol{U}}{\partial x_i} \mathbf{A}_0^{-1} \frac{\partial \boldsymbol{U}}{\partial x_j}} \right]^{1/2} \tag{2.67}$$

where $g^{ij}$ is the element metric tensor $\frac{\partial x_i}{\partial \xi_k} \frac{\partial x_j}{\partial \xi_k}$ and $\Delta \boldsymbol{U} = \frac{\partial \boldsymbol{U}}{\partial t} \Delta t$ is a measure of the unsteadiness of the flow.

**Node Based Discontinuity Capturing Parameter** As discussed by Bova [23], for the essentially the same reason it is desirable to have a node based $\tau_{supg}$ it is also desirable to have a

node based discontinuity capturing parameter. The reasoning for this is because a discontinuous numerical dissipation field may lead to jumps in the solution gradients $\frac{\partial \boldsymbol{U}}{\partial x_j}$. Similar to the previous discussion on construction of a nodally averaged length scale, the same procedure is applied here to construct a nodally averaged discontinuity capturing parameter. One note, however, is that the extrapolation procedure may produce negative values of $\delta$ or $\nu$. This is not physical as the discontinuity capturing parameter is defined to always be greater than zero. A lumped mass $L-2$ projection or simple clipping of negative values ensures that the discontinuity capturing parameter always remains positive. In practice, the nodally averaged discontinuity capturing parameter is vital for producing smooth shock boundaries since the discontinuity capturing parameter is in general element-wise discontinuous.

The nodal averaging procedure implemented here for the discontinuity capturing parameter proceeds as follows:

(1) Compute discontinuity capturing parameters at the integration points as a pre-processing step before any finite element integration begins

(2) Project integration point values to the nodes using a local $L-2$ projection or extrapolation operator and sum the nodal values

(3) Compute a nodal discontinuity capturing parameter by averaging the summed values amongst the patch of elements that support each node

(4) Interpolate the nodal values of the discontinuity capturing parameter to the integration points during the finite element integration of equation 2.42

**Conservation of Enthalpy** The addition of numerical dissipation for the purpose of capturing shocks can effect an important property of the energy equation; that is the total enthalpy of a flow must be constant along a streamline. As discussed by Kirk [83], this can be achieved by having the discontinuity capturing operator act on $\rho H$ instead of $\rho E$ for the energy equation. In

the case of the $\nu$ shock capturing parameter, this transformation to enthalpy variables reads

$$\sum_{e=1}^{n_e} \int_{\Omega_f^e} \nu \left( \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{x_i}} \cdot g^{ij} \mathbf{A}_H \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{x_i}} \right) d\Omega \tag{2.68}$$

where $\mathbf{A}_H$ is a transformation matrix from $\rho E$ to $\rho H$ only for the energy equation. In two-dimensions, for instance, this transformation matrix is expressed as

$$\boldsymbol{A}_H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0.5 v_i v_i (\gamma - 1) & (1-\gamma)\, v_x & (1-\gamma)\, v_y & \gamma \end{bmatrix} \tag{2.69}$$

The same transformation applies for the $\delta$ shock capturing parameter as well.

## 2.4 Solution Strategies for the Navier-Stokes Equations

### 2.4.1 Nonlinear Solution via Newton's Method

Equation 2.42 represents the nonlinear Navier-Stokes residual equations which requires a suitable nonlinear method in order to solve for the conservative variables. Several nonlinear solution strategies exist include: Newton's method, quasi-Newton's method, fixed point iteration methods, and homotopy methods [111]. In the context of computational fluid dynamics, Newton's methods and quasi-Newton's method are a common nonlinear solver of choice and is the strategy used to solve the nonlinear problems in this thesis.

Solving a nonlinear system of equations via Newton's method is conceptually quite simple. We begin by defining a nonlinear residual vector $\boldsymbol{\mathfrak{R}}$ that is a function of a state vector $\boldsymbol{U}$

$$\boldsymbol{\mathfrak{R}}(\boldsymbol{U}) = \mathbf{0} \tag{2.70}$$

Writing out the Taylor series expansion of $\boldsymbol{\mathfrak{R}}$ but truncating after the first order derivative we get

$$\boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{(m)}) + \left[ \frac{\partial \boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{(m)})}{\partial \boldsymbol{U}} \right] \Delta \boldsymbol{U}^{(m+1)} = \mathbf{0} \tag{2.71}$$

where $\boldsymbol{U}^{(m)}$ represents the solution state at the last nonlinear iteration and $\boldsymbol{U}^{(m+1)}$ is an as of yet unknown solution state. Equation 2.71 can be rearranged to yield a linear equation system expression with unknowns $\Delta\boldsymbol{U}^{(m+1)}$

$$\left[\frac{\partial\boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{m})}{\partial\boldsymbol{U}}\right]\Delta\boldsymbol{U}^{(m+1)} = -\boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{m}) \tag{2.72}$$

By solving equation 2.72 for $\Delta\boldsymbol{U}^{(m+1)}$, the previously known solution state $\boldsymbol{U}^{(m)}$ can be updated to yield a new solution state that hopefully is closer to satisfying the original nonlinear residual equation (2.70) via

$$\boldsymbol{U}^{(m+1)} = \boldsymbol{U}^{(m)} + \Delta\boldsymbol{U}^{(m+1)} \tag{2.73}$$

This procedure continues iteratively until certain stopping criteria have been met. The norm of the residual is a useful quantity to watch and the nonlinear system is typically considered to have converged when the residual norm has dropped a predefined amount relative to the initial residual norm at the start of the nonlinear solve. This condition is expressed as

$$\frac{\left\|\boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{(m+1)})\right\|}{\left\|\boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{(m=0)})\right\|} < \epsilon \tag{2.74}$$

where epsilon is a small number, usually something on the order of $1 \times 10^{-4}$ to $5 \times 10^{-1}$ depending on the problem being solved.

It is important to note that the $\partial\boldsymbol{\mathfrak{R}}/\partial\boldsymbol{U}$ term in equation 2.72 represents a first-order linearization of $\boldsymbol{\mathfrak{R}}(\boldsymbol{U}^{(m)})$ and is often referred to as the Jacobian matrix. Because the truncation error is of $\mathcal{O}^2$ and provided $\boldsymbol{U}^{(m=0)}$ is "sufficiently near" the final solution and the Jacobian matrix is exact, Newton's method will exhibit second-order convergence. However, computing the exact Jacobian matrix is both computationally very expensive. In many cases the exact Jacobian is not needed to produce good convergence behavior and in some cases, when the initial guess $\partial\boldsymbol{U}^{(m=0)}$ is far from the solution, may actually produce worse convergence than a approximate Jacobian. This work uses an approximate Jacobian which will soon be discussed.

#### 2.4.1.1 Semi-Discrete Residual Equation

In applying Newton's method to solve the Navier-Stokes equation, the Navier-Stokes total residual vector, as defined by equation 2.42, is re-written here as

$$\mathfrak{R}_f(\dot{\boldsymbol{U}},\boldsymbol{U}) \equiv \int_{\Omega_f} \boldsymbol{W}\cdot\left(\dot{\boldsymbol{U}}+\frac{\partial \boldsymbol{F}_i}{\partial x_i}-\boldsymbol{S}\right)+\frac{\partial \boldsymbol{W}}{\partial x_i}\cdot\boldsymbol{G}_i\,d\Omega-\int_{\Gamma_f}\boldsymbol{W}\cdot\boldsymbol{G}_i\hat{n}_i\,d\Gamma+ \sum_{e=1}^{n_e}\int_{\Omega_f^e}\frac{\partial \boldsymbol{W}}{\partial x_k}\mathbf{A}_k\boldsymbol{\tau}_{supg}\cdot(\boldsymbol{\mathcal{L}}\boldsymbol{U}-\boldsymbol{S})\,d\Omega+\sum_{e=1}^{n_e}\int_{\Omega_f^e}\delta\left(\frac{\partial \boldsymbol{W}}{\partial x_i}\cdot\frac{\partial \boldsymbol{U}}{\partial x_i}\right)d\Omega=\boldsymbol{0} \quad (2.75)$$

$\mathfrak{R}_f(\dot{\boldsymbol{U}},\boldsymbol{U})$ indicates that the total fluid residual equation is both a function of the state $\boldsymbol{U}$ and its time derivative $\dot{\boldsymbol{U}}$. The weight functions can be represented in matrix form for elements with multiple degrees-of-freedom at each node, as is the case for solving the compressible Navier-Stokes equations. $\boldsymbol{W}$ is constructed from the nodal test function values according to

$$\boldsymbol{W}^e=[W_1\boldsymbol{I}\quad W_2\boldsymbol{I}\quad ...\quad W_n\boldsymbol{I}] \quad (2.76)$$

where $W_1$, $W_2$, ..., $W_n$ are the test functions for each elemental node and $\boldsymbol{I}$ is the identity matrix with order equal to the number of elemental degrees-of-freedom. Using the test function matrix, an approximate solution may interpolated from the element-wise nodal solution vector $\boldsymbol{U}^e$ and its time derivatives $\dot{\boldsymbol{U}}^e$ via

$$\boldsymbol{U}^h=\boldsymbol{W}^e\boldsymbol{U}^e \quad (2.77)$$

$$\dot{\boldsymbol{U}}^h=\boldsymbol{W}^e\dot{\boldsymbol{U}}^e \quad (2.78)$$

The above equation can be cast into a generic semi-discrete equation of the following form:

$$\mathfrak{R}_f(\dot{\boldsymbol{U}},\boldsymbol{U})\equiv\boldsymbol{\mathcal{R}}_f(\dot{\boldsymbol{U}},\boldsymbol{U})+\boldsymbol{R}_f(\boldsymbol{U}) \quad (2.79)$$

The nonlinear dynamic residual $\boldsymbol{\mathcal{R}}_f(\dot{\boldsymbol{U}},\boldsymbol{U})$ is assembled from element-level dynamic residual vectors using a standard finite element assembly procedure

$$\boldsymbol{\mathcal{R}}_f(\dot{\boldsymbol{U}},\boldsymbol{U})=\sum_{e=1}^{n_e}\boldsymbol{R}_f^e(\dot{\boldsymbol{U}}^e,\boldsymbol{U}^e) \quad (2.80)$$

where the element-level dynamic residual is defined as

$$\boldsymbol{\mathcal{R}}_f^e(\dot{\boldsymbol{U}}^e, \boldsymbol{U}^e) \equiv \boldsymbol{\mathcal{R}}_{f,galerkin}^e + \boldsymbol{\mathcal{R}}_{f,supg}^e \tag{2.81}$$

and

$$\boldsymbol{\mathcal{R}}_{f,galerkin}^e = \int_{\Omega_f^e} \boldsymbol{W}^{e^T} \cdot \dot{\boldsymbol{U}}^h \, d\Omega \tag{2.82}$$

$$\boldsymbol{\mathcal{R}}_{f,supg}^e = \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_k} \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \dot{\boldsymbol{U}}^h \, d\Omega \tag{2.83}$$

The Navier-Stokes nonlinear static residual $\boldsymbol{R}_f(\boldsymbol{U})$ is assembled from element-level static residual vectors defined by

$$\boldsymbol{R}_f(\boldsymbol{U}) = \sum_{e=1}^{n_e} \boldsymbol{R}_f^e(\boldsymbol{U}^e) \tag{2.84}$$

where the element-level static residual is represented as

$$\boldsymbol{R}_f^e(\boldsymbol{U}) \equiv \boldsymbol{R}_{f,galerkin}^e + \boldsymbol{R}_{f,supg}^e + \boldsymbol{R}_{f,dco}^e \tag{2.85}$$

and

$$\boldsymbol{R}_{f,galerkin}^e = \int_{\Omega_f^e} \boldsymbol{W}^{e^T} \cdot \frac{\partial \boldsymbol{F}_i^h}{\partial x_i} \, d\Omega \; + \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_i} \cdot \boldsymbol{G}_i^h \, d\Omega \; - \int_{\Gamma_f^e} \boldsymbol{W}^{e^T} \cdot \boldsymbol{G}_i^h \hat{n}_i \, d\Gamma \tag{2.86}$$

$$\boldsymbol{R}_{f,supg}^e = \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_k} \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \left( \frac{\partial \boldsymbol{F}_i^h}{\partial x_i} - \frac{\partial \boldsymbol{G}_i^h}{\partial x_i} \right) \, d\Omega \tag{2.87}$$

$$\boldsymbol{R}_{f,dco}^e = \int_{\Omega_f^e} \delta \left( \frac{\partial \boldsymbol{W}^{e^T}}{\partial x_i} \cdot \frac{\partial \boldsymbol{U}^h}{\partial x_j} \right) \, d\Omega \tag{2.88}$$

It is important to discuss the nature of the $\partial \boldsymbol{G}_i / \partial x_i$ term appearing in equation 2.87. The viscous fluxes $\boldsymbol{G}_i$ include spatial gradients of the velocity and temperature (as shown by equations 2.10, 2.4, 2.10) and taking the divergence of these fluxes yields second order spatial derivatives. If finite elements with linear basis functions are used, the majority of the second-order derivative terms are zero because the basis functions have no ability to represent a function that is higher order than itself. In the case of skewed bilinear or trilinear elements the mixed derivative terms are in general non-zero, however. Nonetheless, due to the above observation the viscous flux divergence in the strong form of the residual used in the stabilization term is often dropped. As Jansen [81] points

out, ignoring this contribution to the residual leads to inconsistency of the scheme and may only be safe to do so when diffusional effects of the problem are small. To remedy this situation he proposes a method for approximating this term. In this work, which considers problems with high advection in relation to diffusion, this term has been ignored. The verification problems presented at the end of this chapter confirm that the choice to neglect this term is indeed safe.

#### 2.4.1.2 Jacobian Calculation

Differentiating the total fluid residual $\mathfrak{R}$ with respect to $\boldsymbol{U}$ we obtain the expression

$$\begin{aligned}
\frac{\partial \mathfrak{R}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} \equiv \\
\int_{\Omega_f} \boldsymbol{W} \cdot \left[ \frac{\partial \dot{\boldsymbol{U}}}{\partial \boldsymbol{U}} + \frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{x}_i} \right) - \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}} \right] d\Omega + \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} \, d\Omega - \int_{\Gamma_f} \boldsymbol{W}^T \cdot \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} \hat{n}_i \, d\Gamma + \\
\sum_{e=1}^{n_e} \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}}{\partial x_k}^T \left[ \frac{\partial \mathbf{A}_k}{\partial \boldsymbol{U}} \boldsymbol{\tau}_{supg} \cdot (\boldsymbol{\mathcal{L}} \boldsymbol{U} - \boldsymbol{S}) + \mathbf{A}_k \frac{\partial \boldsymbol{\tau}_{supg}}{\partial \boldsymbol{U}} \cdot (\boldsymbol{\mathcal{L}} \boldsymbol{U} - \boldsymbol{S}) \right] + \\
\frac{\partial \boldsymbol{W}}{\partial x_k}^T \left[ \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \left( \frac{\partial \dot{\boldsymbol{U}}}{\partial \boldsymbol{U}} + \frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{x}_i} \right) - \frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{x}_i} \right) - \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}} \right) \right] d\Omega + \\
\sum_{e=1}^{n_e} \int_{\Omega_f^e} \frac{\partial \delta}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{U}}{\partial x_i} \right) + \delta \left( \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{W}}{\partial x_i} \right) d\Omega \quad (2.89)
\end{aligned}$$

which is the consistent linearization of $\mathfrak{R}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})$ and will result in second-order convergence behavior when used with Newton's method. However, the implementation of every term in equation 2.89 leads to high computational cost and not necessarily required to converge the nonlinear problem. A common practice is to drop some of the lesser important terms in the linearization; this results in an approximate Jacobian matrix.

The approximate Jacobian matrix used for the calculations performed in this thesis is written

as

$$\frac{\partial \mathfrak{R}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} \equiv$$
$$\int_{\Omega_f} \boldsymbol{W} \cdot \left[ \frac{\partial \dot{\boldsymbol{U}}}{\partial \boldsymbol{U}} + \frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{x}_i} \right) - \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}} \right] d\Omega + \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} d\Omega - \int_{\Gamma_f} \boldsymbol{W} \cdot \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} \hat{n}_i \, d\Gamma +$$
$$\sum_{e=1}^{n_e} \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}}{\partial x_k}^T \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \left[ \frac{\partial \dot{\boldsymbol{U}}}{\partial \boldsymbol{U}} + \frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{x}_i} \right) - \frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{x}_i} \right) - \frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}} \right] d\Omega +$$
$$\sum_{e=1}^{n_e} \int_{\Omega_f^e} \delta \left( \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{W}}{\partial x_i} \right) d\Omega \quad (2.90)$$

The derivatives of the inviscid flux Jacobians $\mathbf{A}_k$, the stabilization parameter matrix $\boldsymbol{\tau}_{supg}$, and the shock capturing parameter $\delta$ or $\nu$ with respect to $\boldsymbol{U}$ have been dropped. Issues related to evaluating some of the terms in this equation will now be discussed.

**Computing $\partial \dot{\boldsymbol{U}} / \partial \boldsymbol{U}$**

The partial derivatives $\partial \dot{\boldsymbol{U}} / \partial \boldsymbol{U}$ depends on the specific time integration scheme. For instance, a first-order backward difference formula (BDF-1) will have a different $\partial \dot{\boldsymbol{U}} / \partial \boldsymbol{U}$ than a second-order BDF-2. This issue will be considered further in the section on time integration (2.4.2).

**Computing $\partial (\partial \boldsymbol{F}_i / \partial x_i) / \partial \boldsymbol{U}$**

The partial derivative of inviscid flux divergence term $\partial \boldsymbol{F}_i / \partial x_i$ with respect to the state vector $\boldsymbol{U}$ may be computed by

$$\frac{\partial}{\partial \boldsymbol{U}} \left( \frac{\partial \boldsymbol{F}_i}{\partial x_i} \right) = \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{U}} \frac{\partial \boldsymbol{W}}{\partial x_i} = \mathbf{A}_i \frac{\partial \boldsymbol{W}}{\partial x_i} \quad (2.91)$$

This term can easily be evaluated at the integration points by interpolation of the spatial gradients of the $\mathbf{A}_i$ matrices from nodal values.

**Computing $\partial \boldsymbol{G}_i / \partial \boldsymbol{U}$**

The derivatives of the viscous fluxes $\boldsymbol{G}_i$ may be computed directly by differentiating equation 2.10 with respect to $\boldsymbol{U}$ to yield viscous flux Jacobian matrices similar to how the $\mathbf{A}_i$ matrices were derived (note that $\mathbf{K}_{ij}$ are viscous flux coefficient matrices used for building $\boldsymbol{G}_i$ from $\partial \boldsymbol{U} / \partial x_j$ and not Jacobian matrices).

$$\mathbf{D}_i = \frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} \quad (2.92)$$

Alternatively, if the viscous fluxes $\boldsymbol{G}_i$ are written in quasi-linear form their derivatives may be computed according to

$$\frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} = \frac{\partial}{\partial \boldsymbol{U}}\left(\mathbf{K}_{ij}\frac{\partial \boldsymbol{U}}{\partial x_i}\right) = \frac{\partial \mathbf{K}_{ij}}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{U}}{\partial x_j} + \mathbf{K}_{ij}\frac{\partial \boldsymbol{W}}{\partial x_j} = \frac{\partial \mathbf{K}_{ij}}{\partial x_j} + \mathbf{K}_{ij}\frac{\partial \boldsymbol{W}}{\partial x_j} \ . \tag{2.93}$$

As implemented here, the quasi-linear form is used to avoid having to implement and compute the viscous flux Jacobians $\boldsymbol{D}_i$. In practice the $\partial \mathbf{K}_{ij}/\partial x_i$ term is often ignored yielding an approximate viscous flux derivative which is written as

$$\frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} = \mathbf{K}_{ij}\frac{\partial \boldsymbol{W}}{\partial x_j} \ . \tag{2.94}$$

**Computing $\partial(\partial \boldsymbol{G}_i/\partial x_i)/\partial \boldsymbol{U}$**

The derivatives of the viscous flux divergence term $\partial \boldsymbol{G}_i/\partial x_i$ with respect to $\boldsymbol{U}$ may also be represented in two forms. The first leaves the viscous fluxes in their consistent form; the partial derivative with respect to $\boldsymbol{U}$ may then be represented as

$$\frac{\partial}{\partial \boldsymbol{U}}\left(\frac{\partial \boldsymbol{G}_i}{\partial x_i}\right) = \frac{\partial \boldsymbol{W}}{\partial x_i}\frac{\partial \boldsymbol{G}_i}{\partial \boldsymbol{U}} = \frac{\partial \boldsymbol{W}}{\partial x_i}\mathbf{D}_i \ . \tag{2.95}$$

If the viscous fluxes are written in quasi-linear form, the differentiation reads

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{U}}\left[\frac{\partial}{\partial x_i}\left(\mathbf{K}_{ij}\frac{\partial \boldsymbol{U}}{\partial x_j}\right)\right] = \frac{\partial}{\partial \boldsymbol{U}}\left[\frac{\partial \mathbf{K}_{ij}}{\partial x_i}\frac{\partial \boldsymbol{U}}{\partial x_j} + \mathbf{K}_{ij}\frac{\partial^2 \boldsymbol{U}}{\partial x_i \partial x_j}\right] = \\ \frac{\partial}{\partial \boldsymbol{U}}\left(\frac{\partial \mathbf{K}_{ij}}{\partial x_i}\right)\frac{\partial \boldsymbol{U}}{\partial x_j} + \frac{\partial \mathbf{K}_{ij}}{\partial x_i}\frac{\partial \boldsymbol{W}}{\partial x_j} + \frac{\partial \mathbf{K}_{ij}}{\partial \boldsymbol{U}}\frac{\partial^2 \boldsymbol{U}}{\partial x_i \partial x_j} + \mathbf{K}_{ij}\frac{\partial^2 \boldsymbol{W}}{\partial x_i \partial x_j} \ .\end{aligned} \tag{2.96}$$

Clearly, either of these options is non-trivial to implement. However, as explained in section 2.4.1.1, the viscous part of the residual is left out of the stabilization term; hence, computing the Jacobian of the viscous flux divergence is not needed.

**Matrix Form of the Approximate Jacobian**

Using the approximations above, the Jacobian as written in equation 2.90 can now be expressed in as

$$\frac{\partial \boldsymbol{\mathfrak{R}}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} \equiv \frac{\partial \boldsymbol{\mathcal{R}}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} + \frac{\partial \boldsymbol{R}_f(\boldsymbol{U})}{\partial \boldsymbol{U}} \ . \tag{2.97}$$

The Jacobian of the nonlinear dynamic residual $\boldsymbol{\mathcal{R}}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})$ is assembled from element-level dynamic Jacobian contributions via

$$\frac{\partial \boldsymbol{\mathcal{R}}_f(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} = \sum_{e=1}^{n_e} \frac{\partial \boldsymbol{\mathcal{R}}_f^e(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} \tag{2.98}$$

where the elemental dynamic Jacobian is defined as

$$\frac{\partial \boldsymbol{\mathcal{R}}_f^e(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} \equiv \boldsymbol{\mathcal{J}}_{f,galerkin}^e + \boldsymbol{\mathcal{J}}_{f,supg}^e \ . \tag{2.99}$$

The dynamic Galerkin and SUPG Jacobians are defined, respectively, as

$$\boldsymbol{\mathcal{J}}_{f,galerkin}^e = \int_{\Omega_f^e} \boldsymbol{W}^{e\,T} \cdot \frac{\partial \dot{\boldsymbol{U}}^h}{\partial \boldsymbol{U}} \ d\Omega \tag{2.100}$$

$$\boldsymbol{\mathcal{J}}_{f,supg}^e = \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_k} \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \frac{\partial \dot{\boldsymbol{U}}^h}{\partial \boldsymbol{U}} \ d\Omega \tag{2.101}$$

The Jacobian of the nonlinear static residual $\boldsymbol{R}_f(\boldsymbol{U})$ is assembled from elemental static Jacobian matrices defined by

$$\frac{\partial \boldsymbol{R}_f(\boldsymbol{U})}{\partial \boldsymbol{U}} = \sum_{e=1}^{n_e} \frac{\partial \boldsymbol{R}_f^e(\boldsymbol{U})}{\partial \boldsymbol{U}} \tag{2.102}$$

where the static Jacobian as computed by each element is assembled from the Galerkin, SUPG, and discontinuity capturing terms

$$\frac{\partial \boldsymbol{R}^{s,e}(\boldsymbol{U})}{\partial \boldsymbol{U}} \equiv \mathbf{J}_{f,galerkin}^e + \mathbf{J}_{f,supg}^e + \mathbf{J}_{dco}^{s^e} \tag{2.103}$$

The individual static Jacobians are expressed as

$$\mathbf{J}_{f,galerkin}^e = \int_{\Omega_f^e} \boldsymbol{W}^{e\,T} \cdot \mathbf{A}_i \frac{\partial \boldsymbol{W}^e}{\partial x_i} + \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_i} \cdot \mathbf{K}_{ij} \frac{\partial \boldsymbol{W}}{\partial x_j} \ d\Omega \ - \int_{\Gamma_f^e} \boldsymbol{W}^{e\,T} \cdot \mathbf{K}_{ij} \frac{\partial \boldsymbol{W}^e}{\partial x_j} \hat{n_i} \ d\Gamma \tag{2.104}$$

$$\mathbf{J}_{f,supg}^e = \int_{\Omega_f^e} \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_k} \mathbf{A}_k \boldsymbol{\tau}_{supg} \cdot \left( \mathbf{A}_i \frac{\partial \boldsymbol{W}^e}{\partial x_i} \right) \ d\Omega \tag{2.105}$$

$$\mathbf{J}_{f,dco}^e = \int_{\Omega_f^e} \delta \left( \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_i} \cdot \frac{\partial \boldsymbol{W}^{e\,T}}{\partial x_j} \right) \ d\Omega \tag{2.106}$$

#### 2.4.1.3 Finite Difference Jacobian Calculation

The fully consistent Jacobian (equation 2.89) can also be approximated by assembling elemental Jacobian contributions that have been computed via finite differencing of the dynamic and static residual equation (2.75). Accordingly, the dynamic Jacobian matrix may be computed by finite differences by

$$\frac{\partial \widetilde{\boldsymbol{\mathcal{R}}_f^e(\boldsymbol{U})}}{\partial \boldsymbol{U}} \cong \frac{\boldsymbol{\mathcal{R}}_f^e(\dot{\boldsymbol{U}} + \epsilon, \boldsymbol{U}) - \boldsymbol{\mathcal{R}}_f^e(\dot{\boldsymbol{U}} - \epsilon, \boldsymbol{U})}{2\epsilon} \cdot \frac{\partial \dot{\boldsymbol{U}}}{\partial \boldsymbol{U}} \tag{2.107}$$

and the static Jacobian matrix is computed via

$$\frac{\partial \widetilde{\boldsymbol{R}_f^e(\boldsymbol{U})}}{\partial \boldsymbol{U}} \cong \frac{\boldsymbol{\mathfrak{R}}_f^e(\dot{\boldsymbol{U}}, \boldsymbol{U} + \epsilon) - \boldsymbol{\mathfrak{R}}_f^e(\dot{\boldsymbol{U}}, \boldsymbol{U} - \epsilon)}{2\epsilon} \ . \tag{2.108}$$

Hence, the total Jacobian of equation 2.97 can be approximated by assembling the elemental contributions as follows

$$\frac{\partial \boldsymbol{\mathfrak{R}}_f^e(\dot{\boldsymbol{U}}, \boldsymbol{U})}{\partial \boldsymbol{U}} \cong \frac{\partial \widetilde{\boldsymbol{\mathcal{R}}_f^e(\boldsymbol{U})}}{\partial \boldsymbol{U}} + \frac{\partial \widetilde{\boldsymbol{R}_f^e(\boldsymbol{U})}}{\partial \boldsymbol{U}} \tag{2.109}$$

Computing the approximate Jacobian in this fashion is much slower than via the analytic form given in the previous section because it requires the nonlinear dynamic and total residual to be calculated twice for every element during each nonlinear iteration. Other procedures for computing the Jacobian via finite differences with fewer operations are possible but have not been explored in this work. Nonetheless, this form yields a very good approximation to the consistent Jacobian because the finite differencing of the residual vector captures all of the nonlinearities contained therein. For instance, the derivatives of the stabilization matrix $\boldsymbol{\tau}_{supg}$, the shock capturing parameter $\delta$ or $\nu$, and the derivatives $\partial \mathbf{A}_k / \partial \boldsymbol{U}$ from equation 2.89 that were dropped in the approximate Jacobian of equation 2.90 are all accounted for by the finite difference approach of equation 2.109.

### 2.4.2 Time Integration via Backward Differentiation Formulas

By and large, the time integrator of choice for the semi-discrete Navier-Stokes equations given by equation 2.75 and subsequently equation 2.79 is the class of backward differentiation formula (BDF) time integrators. The first-order accurate BDF time integrator yields the familiar backward

Euler scheme while the second-order accurate BDF integrator is the 3-point backward difference scheme. These methods have become the standard choice of many computational fluid dynamics codes (e.g. [42, 77]) by virtue of the fact that for time-accurate flow computations the 3-point BDF formula exhibits slightly more numerical dissipation than a counterpart scheme such as the Crank-Nicholson method [41].

**Fixed Time Step Backward Differentiation Formulas**

For a fixed time step $\Delta t$, the first-order accurate backward Euler formula is simply written as

$$\dot{\boldsymbol{U}} = \frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\boldsymbol{U}^{n+1} - \boldsymbol{U}^{n}}{\Delta t} \tag{2.110}$$

while the second-order accurate 3-point BDF scheme is written as

$$\dot{\boldsymbol{U}} = \frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\frac{3}{2}\boldsymbol{U}^{n+1} - 2\boldsymbol{U}^{n} + \frac{1}{2}\boldsymbol{U}^{n-1}}{\Delta t} \tag{2.111}$$

where $\boldsymbol{U}^{n+1}$, $\boldsymbol{U}^{n}$, $\boldsymbol{U}^{n-1}$ are the solution state vectors at the new, previous, and second previous time iterates. These equations are commonly found in any number of textbooks including numerical differentiation formulas. Time integration proceeds by using these formulas to compute approximations for the time derivatives of the state vector $\boldsymbol{U}$ needed by equation 2.79.

**Adaptive Time Stepping**

One of the challenges of solving the Navier-Stokes equations is time integrating in such a way that one can quickly and efficiently obtain a solution, whether it be a steady-state solution or a transient flow analysis. Unfortunately, it is intractable to solve the stationary form of equation 2.79 in which any time dependent terms are ignored (as is often done in heat transfer or elasticity) in order to obtain a steady-state solution. This is because the Navier-Stokes equations are simply too nonlinear for such an approach to be feasible. In this case, the common practice for steady-state flow solutions is to time integrate the transient equations until $\boldsymbol{U}$ no longer changes. Using a fixed time step $\Delta t$ to integrate to steady-state is prohibitively slow since the time step size used to begin the solution process is typically very small (on the order of $10^{-9}$ to $10^{-3}$ depending on the problem). Clearly, using such a small time step after the flow has had a chance to develop is not necessary.

In order to provide a faster solution process, two methods are commonly used to time integrate the Navier-Stokes equations: adaptive time stepping and nodal time stepping. In the nodal time stepping approach, each node is assigned a time step based on the Courant-Freidrichs-Levy (CFL) number. This condition provides an stable local time step based on the state of the flow and the element size. This approach, however, is only intended for steady-state flows. Adaptive global time stepping provides a way of selecting a new time step to gradually increase or decrease the value in order to maintain stability while integrating the equations in an efficient manner and also applies to integrating the transient equations in a time accurate fashion.

The adaptive time step selection method used in this work is due to Gresho [49], and is written as

$$\Delta t^{n+1} = \Delta t^n \left( b \frac{\epsilon^t}{d^{n+1}} \right)^a \tag{2.112}$$

where $a = 1/2$ and $b = 2$ for a first-order time integrator and $a = 1/3$ and $b = 3(1 + \Delta t^{n-1}/\Delta t^n)$ for a second-order time integrator. The quantity $d^{n+1}$ is a measure of the difference between the initial or predicted nonlinear iterate $\boldsymbol{U}^{(m=0)}$ and the converged state vector $\boldsymbol{U}^{(m+1)}$ for the current nonlinear iteration and is represented as

$$d^{n+1} = \frac{1}{\sqrt{n_{dof}} \|\boldsymbol{U}\|_\infty} \left[ \left( \boldsymbol{U}^{(m+1),n+1} - \boldsymbol{U}^{(m=0),n+1} \right)^2 \right]^{1/2} . \tag{2.113}$$

Experience has shown that this technique does a good job of detecting the solution behavior and adjusting the time step accordingly. However, it is also common practice to limit the time step size increase to be no more than 10% to 20% larger than the previous time step size.

**Adaptive Time Step Backward Differentiation Formulas**

If the time steps are permitted to vary, the backward differentiation formulas previously presented are no longer valid. In order to obtain adaptive time step compatible formulas, we follow the derivation presented by Kirk [82]. Beginning with a Taylor series expansions for $\boldsymbol{U}^n$ and $\boldsymbol{U}^{n-1}$

$$\boldsymbol{U}^n = \boldsymbol{U}^{n+1} + \frac{\partial \boldsymbol{U}^{n+1}}{\partial t} \left( t^n - t^{n+1} \right) + \frac{\partial^2 \boldsymbol{U}^{n+1}}{\partial t^2} \frac{\left( t^n - t^{n+1} \right)^2}{2} + \mathcal{O}\left[ (t^n - t^{n+1})^3 \right] \tag{2.114}$$

$$\boldsymbol{U}^{n-1} = \boldsymbol{U}^{n+1} + \frac{\partial \boldsymbol{U}^{n+1}}{\partial t} \left( t^{n-1} - t^{n+1} \right) + \frac{\partial^2 \boldsymbol{U}^{n+1}}{\partial t^2} \frac{\left( t^{n-1} - t^{n+1} \right)^2}{2} + \mathcal{O}\left[ (t^{n-1} - t^{n+1})^3 \right] \tag{2.115}$$

Making the following definitions for the time step sizes

$$\Delta t^{n+1} = t^{n+1} - t^n \quad , \quad \Delta t^n = t^n - t^{n-1} \quad , \quad \Delta t^{n+1} + \Delta t^n = t^{n+1} - t^{n-1} \tag{2.116}$$

equations 2.114 and 2.115 can be rearranged to yield

$$\frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\boldsymbol{U}^{n+1}}{\Delta t^{n+1}} - \frac{\boldsymbol{U}^n}{\Delta t^{n+1}} + \frac{\partial^2 \boldsymbol{U}^{n+1}}{\partial t^2} \frac{\left(\Delta t^{n+1}\right)^2}{2} - \mathcal{O}\left[\left(\Delta t^{n+1}\right)^2\right] \tag{2.117}$$

$$\frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\boldsymbol{U}^{n+1}}{\Delta t^{n+1} + \Delta t^n} - \frac{\boldsymbol{U}^n}{\Delta t^{n+1} + \Delta t^n} + \frac{\partial^2 \boldsymbol{U}^{n+1}}{\partial t^2} \frac{\left(\Delta t^{n+1+\Delta t^n}\right)^2}{2} - \mathcal{O}\left[\left(\Delta t^{n+1+\Delta t^n}\right)^2\right] \tag{2.118}$$

Ignoring the higher-order terms of equation 2.117 produces the first-order backward Euler scheme

$$\frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\boldsymbol{U}^{n+1}}{\Delta t^{n+1}} - \frac{\boldsymbol{U}^n}{\Delta t^{n+1}} \tag{2.119}$$

while a combination of equations 2.117 and 2.118 will produce the 3-point backward difference scheme

$$\frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \left[\frac{1}{\Delta t^{n+1}} + \frac{1}{\Delta t^n} - \frac{\Delta t^{n+1}}{\Delta t^n(\Delta t^{n+1} + \Delta t^n)}\right] \boldsymbol{U}^{n+1} - \left[\frac{1}{\Delta t^{n+1}} + \frac{1}{\Delta t^n}\right] \boldsymbol{U}^n + \left[\frac{\Delta t^{n+1}}{\Delta t^n(\Delta t^{n+1} + \Delta t^n)}\right] \boldsymbol{U}^{n-1} \tag{2.120}$$

It is easily verified that if $\Delta t^{n+1} = \Delta t^n = \Delta t$ then equation 2.120 reproduces the fixed time step scheme given in equation 2.111.

## 2.5 Arbitrary Lagrangian/Eulerian Form of the Navier-Stokes Equations

When a fluid/structure interface moves in time, as is often the case in aeroelastic or ablation problems, the numerical framework must appropriately account for the changes to the boundary. Several techniques have been developed to handle such problems; these approaches can be divided into fixed mesh and moving mesh methods.

**Fixed Mesh Methods** The aim of fixed methods is to approximate the boundary using a numerical model as it passes through a stationary mesh. The immersed boundary method (IBM) [116, 117, 108] or the extended finite element method (XFEM) [32, 17] are two examples of such schemes. The IBM and XFEM approaches capture the interface as it moves through the mesh

and are well suited for treating arbitrary interface changes. Nonetheless, the methods require significant implementation effort and are arguably not as good at maintaining the solution quality at the interface unless mesh adaptivity is employed.

**Moving Mesh Methods** A moving mesh method tracks the interface by moving the nodal positions of the mesh to adjust to the boundary as it changes. The arbitrary Lagrangian/Eulerian (ALE) [56, 66] formulation is probably the best known moving mesh method. The ALE formulation is used to solve the governing equations on a mesh that is moved to track the fluid/structure interface. This method is straightforward to implement and because the mesh moves as the interface moves is much better suited for ensuring sufficient mesh resolution at the boundary than fixed mesh methods. Since capturing the boundary layer effects of a compressible viscous flow is of utmost importance this is significant advantage. However, the primary drawback of a moving mesh technique is its inability to handle complex boundary deformations that may severely warp or distort the mesh. Mesh adaptivity can be employed to better handle complex interface changes but this comes at the expense of implementation simplicity.

Given that most aerodynamic problems do not typically involve complex boundary deformations, as is the case with aeroelasticity and ablation problems, the moving mesh ALE approach is used in this work. The ALE method is a generalization of the classical Eulerian and Lagrangian frames of reference. The predominant frame of reference for a computational fluid dynamics problem is the Eulerian frame of reference, however, when the mesh moves it gains a Lagrangian reference component. Correctly accounting for this is the key to the ALE formulation.

### 2.5.1 ALE Form of the Conservation Laws

In order to correct for the effect the deformation of the mesh has on the flux terms, the mass, momentum, and energy conservation equations (2.1 – 2.3) must be written in an ALE frame of reference. Here the ALE form of the mass conservation equation is derived; the reader is referred to Donea [35] for a more complete exposition of ALE methods in a finite element context.

**ALE Form of the Mass Conservation Equation** The conservation of mass states that the time rate of change of mass in a control volume will remain constant. This can be simply stated in equation form as

$$\frac{D}{Dt}\int_{\Omega_t} \rho \; d\Omega = 0 \; . \tag{2.121}$$

Using the Reynolds transport theorem, this previous expression can be re-written as

$$\int_{\Omega_t} \frac{\partial \rho}{\partial t} \; d\Omega + \int_{\Gamma_t} \rho v_i \cdot \hat{n_i} \; d\Gamma = 0 \; . \tag{2.122}$$

The Divergence theorem, which relates a volume integral of a divergence term to a boundary integral via $\int_\Omega \frac{\partial \phi_i}{\partial x_i} = \int_\Gamma \phi_i \cdot n_i d\Gamma$, may be applied to equation 2.122 to arrive at the following

$$\int_{\Omega_t} \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho v_i\right) d\Omega = 0 \; . \tag{2.123}$$

The previous integral holds for any $\Omega_t$ so we can drop the integral and obtain the differential form of the mass conservation law in an Eulerian frame of reference as previously expressed in equation 2.1

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho v_i\right) = 0 \; . \tag{2.124}$$

By expanding the divergence term in equation 2.124 it may be re-written as

$$\frac{\partial \rho}{\partial t} + v_i \frac{\partial \rho}{\partial x_i} + \rho \frac{\partial v_i}{\partial x_i} = 0 \; . \tag{2.125}$$

Next, we recall the definition of the total derivative, which is written as,

$$\frac{D\phi}{Dt} \equiv \left.\frac{\partial \phi}{\partial t}\right|_X + v_i \frac{\partial \phi}{\partial x_i} \tag{2.126}$$

where $\big|_X$ means the term is evaluated with respect to the reference or initial configuration. This definition may be substituted into equation 2.125 to produce

$$\frac{D\rho}{Dt} + \frac{\partial v_i}{\partial x_i} d\Omega = 0 \; . \tag{2.127}$$

Given equation 2.127, we can cast this into an ALE frame of reference by making use of the following fundamental ALE relation

$$\frac{D\phi}{Dt} \equiv \left.\frac{\partial \phi}{\partial t}\right|_\chi + c_i \frac{\partial \phi}{\partial x_i} = 0 \; . \tag{2.128}$$

where $\big|_\chi$ means the term is evaluated with respect to the so-called referential configuration. The convective velocity $c_i$ in equation 2.128 is the relative velocity between the material and the mesh and expressed as

$$c_i \equiv v_i - v_i^m \tag{2.129}$$

Using the definition of the ALE total derivative, the mass conservation equation can be written as

$$\left.\frac{\partial \rho}{\partial t}\right|_\chi + c_i \frac{\partial \rho}{\partial x_i} + \rho \frac{\partial v_i}{\partial x_i} = 0 \tag{2.130}$$

and by substituting the definition of $c_i$ into the previous equation we obtain

$$\left.\frac{\partial \rho}{\partial t}\right|_\chi + v_i \frac{\partial \rho}{\partial x_i} + \rho \frac{\partial v_i}{\partial x_i} - v_i^m \frac{\partial \rho}{\partial x_i} = 0 \ . \tag{2.131}$$

Recognizing that the second and third terms in the previous equation are simply the divergence of the momentum $\rho v_i$ we may then express the ALE form of the mass conservation equation as

$$\left.\frac{\partial \rho}{\partial t}\right|_\chi + \frac{\partial}{\partial x_i}\left(\rho v_i\right) - \hat{v}_i \frac{\partial \rho}{\partial x_i} = 0 \ . \tag{2.132}$$

**Vector Form of the ALE Navier-Stokes Equations** The previous derivation of the ALE form of the mass conservation equation extends readily to the momentum and energy equations. After these expressions are obtained, the vector form of the ALE based Navier-Stokes equations, previously expressed as equation 2.6, becomes

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - v_i^m \frac{\partial \boldsymbol{U}}{\partial \boldsymbol{x}_i} - \frac{\partial \boldsymbol{G}_i}{\partial x_i} - \boldsymbol{S} = 0 \ . \tag{2.133}$$

### 2.5.2 Time Accurate ALE Integrators

Time integration of the semi-discrete Navier-Stokes equations written in an ALE frame of reference requires special treatment to preserve the accuracy of the integrator. The reason for this has to do with the fact that the computation of the flux terms on a moving grid must be done in a fashion that satisfies the so-called discrete geometric conservation laws (DGCL). The underlying premise of the DGCL is that the computation of the geometric quantities (i.e. the mesh coordinates $x_i$ and the mesh velocities $v_i^m$ in equation 2.133) must done in fashion that is consistent with the

time integration scheme in order to preserve the order of accuracy the same integrator would achieve on a fixed mesh. The DGCL states that a time-accuracy preserving integration scheme will preserve the state of a uniform flow on a moving mesh. Farhat and coworkers have studied the implications of the DGCL extensively [40, 41, 38] and presented several ALE time integrators that maintain the same accuracy as their fixed grid counter parts [39]. This section serves as an overview of ALE time integrator techniques laid out by Farhat et al. for solving the Navier-Stokes equations via a finite volume method (reference [39]) and extends these concepts to the Navier-Stokes equations solved via finite element methods.

As expressed in reference [39], a choice must be made as to where the inviscid fluxes $\boldsymbol{F}_i$ and viscous fluxes $\boldsymbol{G}_i$ should be evaluated when integrating across a time slab from $t^n$ to $t^{n+1}$. One option is to time average the flux terms $\boldsymbol{F}_i$ and $\boldsymbol{G}_i$ computed at a distinct point and the other is to time average the mesh configurations over the time slab and compute a single flux term using the result.

**Flux Time-Averaging** With this averaging option, the flux terms are computed at predetermined positions within the time slab and the resulting fluxes are time averaged according to the equations

$$\bar{\boldsymbol{F}}_i^{ALE} = \sum_{k=1}^{k_F} \alpha_k \ \boldsymbol{F}_i(\boldsymbol{U}^{n+1}, x_{F,i}^k, v_i^{m,k}) \tag{2.134}$$

$$\bar{\boldsymbol{G}}_i = \sum_{k=1}^{k_G} \beta_k \ \boldsymbol{G}_i(\boldsymbol{U}^{n+1}, x_{G,i}^k) \tag{2.135}$$

where $x_{F,i}^k$ and $x_{G,i}^k$ are the spatial coordinates for time plane $k$ used for evaluating the inviscid fluxes ($\boldsymbol{F}_i$)and viscous fluxes ($\boldsymbol{G}_i$), respectively, $v_i^{m,k}$ are the time plane mesh velocities.

**Mesh Time-Averaging** In this option, a time averaged mesh configuration is generated and a single flux term is computed according to

$$\hat{\boldsymbol{F}}_i^{ALE} = \boldsymbol{F}_i^{ALE}(\boldsymbol{U}^{n+1}, \bar{n}_i, \bar{\kappa}_i) \tag{2.136}$$

$$\hat{\boldsymbol{G}}_i = \boldsymbol{F}_i(\boldsymbol{U}^{n+1}, \bar{\boldsymbol{x}}) \tag{2.137}$$

where, according to reference [39], for a finite volume scheme the mesh averaged quantities are defined by

$$\bar{\boldsymbol{\nu}} = \sum_{k=1}^{k_F} \alpha_k \; \psi(\boldsymbol{x}_F^k) \tag{2.138}$$

$$\bar{\boldsymbol{\kappa}} = \sum_{k=1}^{k_F} \alpha_k \; \boldsymbol{v}^m \cdot \psi(\boldsymbol{x}_F^k) \tag{2.139}$$

$$\bar{\boldsymbol{x_G}} = \sum_{k=1}^{k_G} \beta_k \; \boldsymbol{x}_G^k \tag{2.140}$$

and the normal vector $\boldsymbol{n}$ is a nonlinear function of the mesh position vector $\boldsymbol{x}$ according to $\boldsymbol{n} = \psi(\boldsymbol{x})$.

From a computational perspective, the mesh time averaging option is more efficient. However, this scheme, as presented above, is only applicable to a finite volume method by virtue of the nonlinear function $\psi$ used to compute cell normal vectors. Hence, this option is not readily applicable to finite element methods and only the flux averaging option will be explored further.

#### 2.5.2.1 ALE Version of the 3-Point Backward Difference Formula

The ALE form of the 3-point backward difference scheme presented in reference [39] is shown in this section. For simplicity the fixed time step form of the differential formula for computing the time derivatives of the conservative variable vector $\boldsymbol{U}$ in equation 2.111 is repeated here

$$\dot{\boldsymbol{U}} = \frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\frac{3}{2}\boldsymbol{U}^{n+1} - 2\boldsymbol{U}^n + \frac{1}{2}\boldsymbol{U}^{n-1}}{\Delta t} \tag{2.141}$$

Computation of the inviscid flux evaluation coordinates ($\boldsymbol{x}_F^k$) and velocities $\boldsymbol{v}^k$ as well and viscous flux evaluation coordinates $\boldsymbol{x}_G^k$ are done according to the parameterizations

$$\boldsymbol{x}_F^k = \zeta_k^{n+1}\boldsymbol{x}^{n+1} + \zeta_k^n\boldsymbol{x}^n + \zeta_k^{n-1}\boldsymbol{x}^{n-1} \tag{2.142}$$

$$\boldsymbol{v}^k = \frac{\theta_k^{n+1}\boldsymbol{x}^{n+1} + \theta_k^n\boldsymbol{x}^n + \theta_k^{n-1}\boldsymbol{x}^{n-1}}{\Delta t} \tag{2.143}$$

$$\boldsymbol{x}_G^k = \eta_k^{n+1}\boldsymbol{x}^{n+1} + \eta_k^n\boldsymbol{x}^n + \eta_k^{n-1}\boldsymbol{x}^{n-1} \tag{2.144}$$

Farhat proposes two sets of rules for selecting the coefficients $\zeta$, $\theta$, and $\eta$ in a way that will lead to a accuracy preserving ALE time integrator. The specifics of those rules are not covered here and

the reader is referred to reference [39] for those details. The paper lists two parameter sets and hence integration schemes that maintain formal second-order time accuracy. One scheme averages across 4 time planes while the other simply uses the time plane at $t^{n+1}$. However, only the first satisfies the DGCL and hence can be guaranteed to preserve the nonlinear stability of the method. As such, only the 4 time plane averaging method is overviewed here.

This scheme uses combinations of the three solution states ($\boldsymbol{U}^{n+1}$, $\boldsymbol{U}^{n}$, $\boldsymbol{U}^{n-1}$) known by the 3-point backward difference integrator to compute 4 mesh configurations for averaging of the inviscid fluxes and simply uses the mesh configuration at $t^{n+1}$ for computing the viscous fluxes, hence $k_F = 4$ and $k_G = 1$. The coefficients used for the time averaging needed by equations 2.134 – 2.135 and 2.142 – 2.144 are

$$\alpha_1 = \alpha_2 = \frac{3}{4} \quad , \quad \alpha_3 = \alpha_4 = -\frac{1}{4} \quad , \quad \beta_1 = 1 \tag{2.145}$$

and

$$\begin{aligned}
\zeta_1^{n+1} &= \frac{1}{2}\left(1 + \frac{1}{\sqrt{3}}\right), \quad \zeta_1^{n} = \tfrac{1}{2}\left(1 - \tfrac{1}{\sqrt{3}}\right), \quad \zeta_1^{n-1} = 0, \\
\zeta_2^{n+1} &= \frac{1}{2}\left(1 - \frac{1}{\sqrt{3}}\right), \quad \zeta_2^{n} = \tfrac{1}{2}\left(1 + \tfrac{1}{\sqrt{3}}\right), \quad \zeta_2^{n-1} = 0, \\
\zeta_3^{n+1} &= 0, \quad \zeta_3^{n} = \tfrac{1}{2}\left(1 + \tfrac{1}{\sqrt{3}}\right), \quad \zeta_3^{n} = \frac{1}{2}\left(1 - \frac{1}{\sqrt{3}}\right), \\
\zeta_4^{n+1} &= 0, \quad \zeta_4^{n} = \tfrac{1}{2}\left(1 - \tfrac{1}{\sqrt{3}}\right), \quad \zeta_4^{n} = \frac{1}{2}\left(1 + \frac{1}{\sqrt{3}}\right), \\
\theta_1^{n+1} = \theta_2^{n+1} &= 1, \qquad \theta_1^{n} = \theta_2^{n} = -1, \qquad \theta_1^{n-1} = \theta_2^{n-1} = 0, \\
\theta_3^{n+1} = \theta_4^{n+1} &= 0, \qquad \theta_3^{n} = \theta_4^{n} = 1, \qquad \theta_3^{n-1} = \theta_4^{n-1} = -1, \\
\eta_1^{n+1} &= 1, \qquad \eta_1^{n} = 0, \qquad \eta_1^{n-1} = 0
\end{aligned} \tag{2.146}$$

#### 2.5.2.2 ALE Version of the Backward Euler Scheme

While not explicitly stated in either reference [41] or [39] for viscous flows, the ALE form of the first-order accurate backward Euler scheme is a simplification of the 3-point scheme and is presented here. Again for the sake of simplicity, the fixed time step form of this method is used

and repeated here from equation 2.110

$$\dot{\boldsymbol{U}} = \frac{\partial \boldsymbol{U}^{n+1}}{\partial t} = \frac{\boldsymbol{U}^{n+1} - \boldsymbol{U}^{n}}{\Delta t} \tag{2.147}$$

Computation of the inviscid flux evaluation coordinates $\boldsymbol{x}_F^k$ and velocities $\boldsymbol{v}^k$ and the viscous flux evaluation coordinates $\boldsymbol{x}_G^k$ are done according to the parameterizations

$$\boldsymbol{x}_F^k = \zeta_k^{n+1}\boldsymbol{x}^{n+1} + \zeta_k^n \boldsymbol{x}^n \tag{2.148}$$

$$\boldsymbol{v}^k = \frac{\theta_k^{n+1}\boldsymbol{x}^{n+1} + \theta_k^n \boldsymbol{x}^n}{\Delta t} \tag{2.149}$$

$$\boldsymbol{x}_G^k = \eta_k^{n+1}\boldsymbol{x}^{n+1} + \eta_k^n \boldsymbol{x}^n \tag{2.150}$$

This scheme uses combinations of the two solution states ($\boldsymbol{U}^{n+1}$, $\boldsymbol{U}^n$) known by the backward Euler integrator to compute 2 mesh configurations for averaging of the inviscid fluxes and simply uses the mesh configuration at $t^{n+1}$ for computing the viscous fluxes, hence $k_F = 2$ and $k_G = 1$. The coefficients used for the time averaging needed by equations 2.134 – 2.135 and 2.148 – 2.150 are

$$\alpha_1 = \alpha_2 = \frac{1}{2} \quad , \quad \beta_1 = 1 \tag{2.151}$$

and

$$\begin{aligned}
\zeta_1^{n+1} &= \frac{1}{2}\left(1 + \frac{1}{\sqrt{3}}\right), \quad \zeta_1^n = \tfrac{1}{2}\left(1 - \tfrac{1}{\sqrt{3}}\right) \\
\zeta_2^{n+1} &= \frac{1}{2}\left(1 - \frac{1}{\sqrt{3}}\right), \quad \zeta_2^n = \tfrac{1}{2}\left(1 + \tfrac{1}{\sqrt{3}}\right), \\
\theta_1^{n+1} &= \theta_2^{n+1} = 1, \qquad \theta_1^n = \theta_2^n = -1, \\
\eta_1^{n+1} &= 1, \qquad\qquad \eta_1^n = 0
\end{aligned} \tag{2.152}$$

## 2.6 Numerical Example Problems

This section presents several example problems that highlight the SUPG finite element flow solver developed to complete the work in later chapters. Problems are shown for viscous flows with both fixed and moving boundaries.

For verification purposes, many of the numerical examples were also run with a variety of other codes and discretizations. The results from the other codes have been included where appropriate for the purpose of verifying that the implementation at hand is indeed correct. The three primary codes used for the majority of these numerical examples are listed and described below.

- CU/FEMDOC: Stabilized finite element based code developed at the University of Colorado at Boulder for this thesis.
- SNL/Aria: Stabilized finite element based code being developed at Sandia National Laboratories. Aria is very similar to CU/FEMDOC. Notable exceptions are Aria's use of the $\nu$ shock capturing parameter its integration by parts of the inviscid flux terms.
- ANSYS/Fluent: Cell centered unstructured finite volume code that is commercially available. Details of Fluent's numerical formulation can be found in its documentation [77].

### 2.6.1 2-Dimensional Carter Flat Plate

The Carter problem [27] is a classic problem that involves low Reynolds number but supersonic flow over a flat plate. The purpose of this test problem is to compute and compare skin friction and heat transfer coefficients along the plate with other numerical solutions. This example repeats the work of Carter and Shakib [124] to verify that the current solver matches other published results. Additionally, the same problem is solved with other flow codes to verify the accuracy and performance characteristics of the current solver.

#### 2.6.1.1 Problem Description

This problem consists of a $M_\infty = 3.0$, $Re = 1000$ flow over a flat plate. Figure 2.1 shows the physical domain and boundary conditions. The problem is specified in an entirely non-dimensional fashion. The domain boundaries span the area $-0.2 \leq x \leq 1.2$ and $0.0 \leq y \leq 0.8$, and the leading edge of the plate is at the origin of the domain. The inflow boundary has all four conservative variables specified ($\rho = 1, \rho u_x = 1, \rho u_y = 0, \rho E = 2.769 \times 10^{-4}$), the symmetric boundary imposes

$u_y = \tau_{xy} = q_y = 0$, the plate satisfies the no-slip condition of $u_x = u_y = 0$ with an isothermal wall temperature equal to the stagnation temperature $T_w = 7.754 \times 10^{-4}$. The outflow boundary has no prescribed variables, however, the boundary term appearing in equation 2.38 is appropriately integrated.

The Sutherland law is used to model the viscosity of the fluid and takes the form $\mu = 0.0906T^{3/2}/(T+0.0001406)$. The thermal conductivity of the fluid is modeled by a constant Prandtl number as $\kappa = c_p\mu/Pr$, where $Pr = 0.71$.

#### 2.6.1.2 Computational Mesh and Model

A hierarchy of meshes are used for solving this problem where the domain consists of 28x16, 56x32, 112x64, 224x128, and 448x256 quadrilateral elements. Figure 2.2(a) shows the 224x128 element mesh, while Figure 2.2(b) show the 224x128 element mesh and the domain boundary used for a 16 parallel process simulation.

The problem is time integrated to steady-state via a backward Euler (BDF-1) scheme beginning with in an initial time step of $1.0 \times 10^{-4}$ $s$. The adaptive time stepping scheme discussed in section 2.4.2 is used with a time step increase limit of 10 %. A Newton scheme is used to solve the nonlinear equations. The nonlinear iteration loop exits when the nonlinear residual drops at least two orders of magnitude or when the initial nonlinear residual norm is less than $1.0 \times 10^{-3}$. The Trilinos/Aztec iterative linear solver package is used to solve the linearized equation system at each Newton step. The GMRES solver is used with a global ILU(0) preconditioner and the iterative solve exits when the linear residual norm has drop four orders of magnitude.

#### 2.6.1.3 Results and Discussion

Figure 2.3 shows the nonlinear residual norm and the time step sizes as the problem is integrated toward steady-state. The problem is run for a fixed number of time iterations for performance comparisons purposes.

Figure 2.4 shows solution contour plots for the finest mesh. These plots compare qualitatively

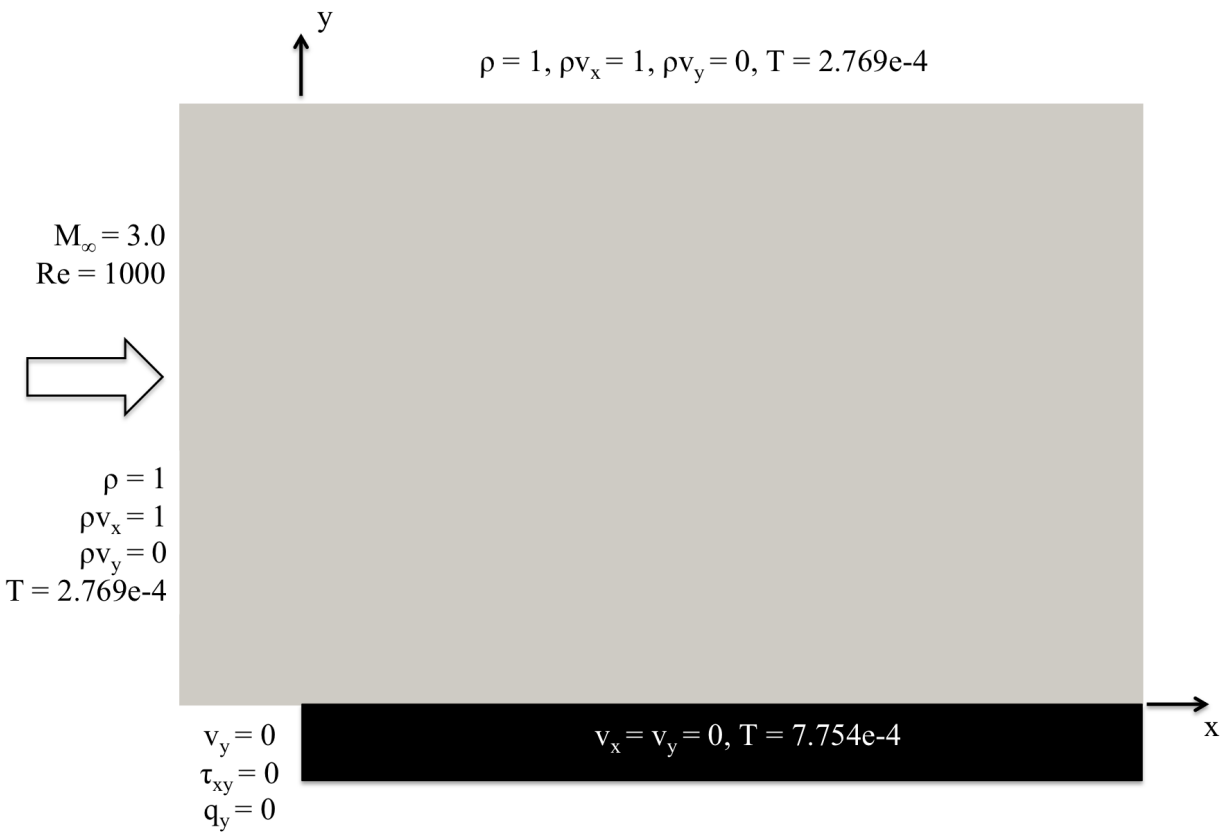
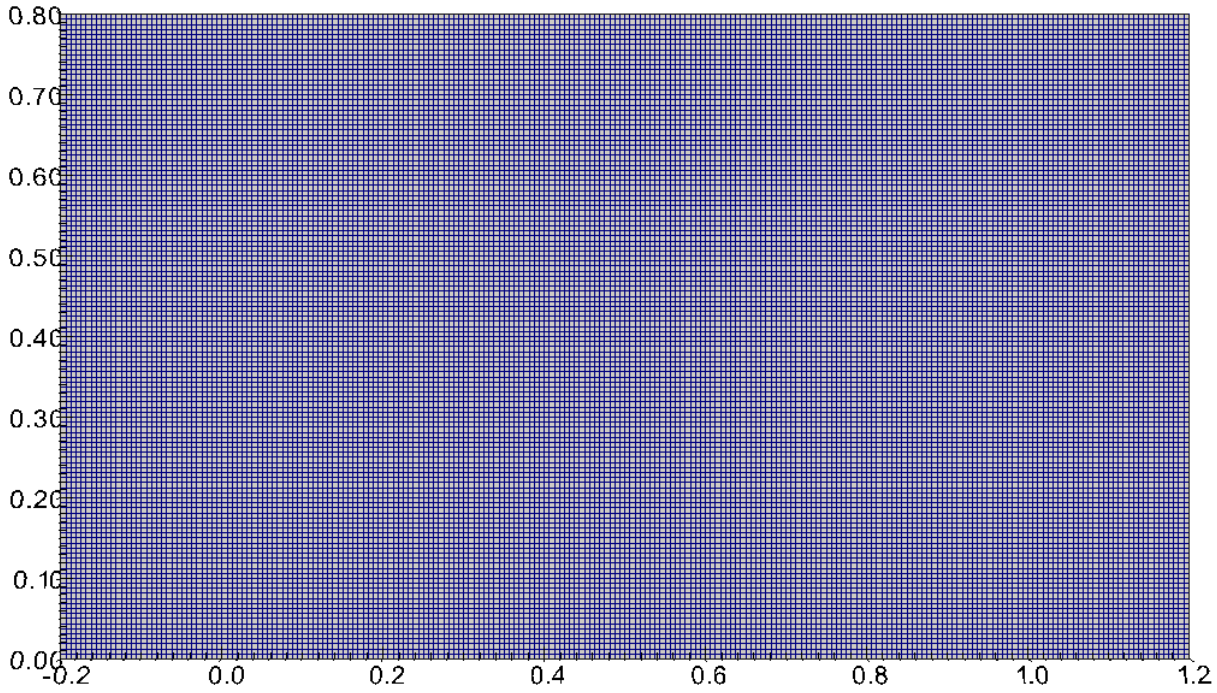
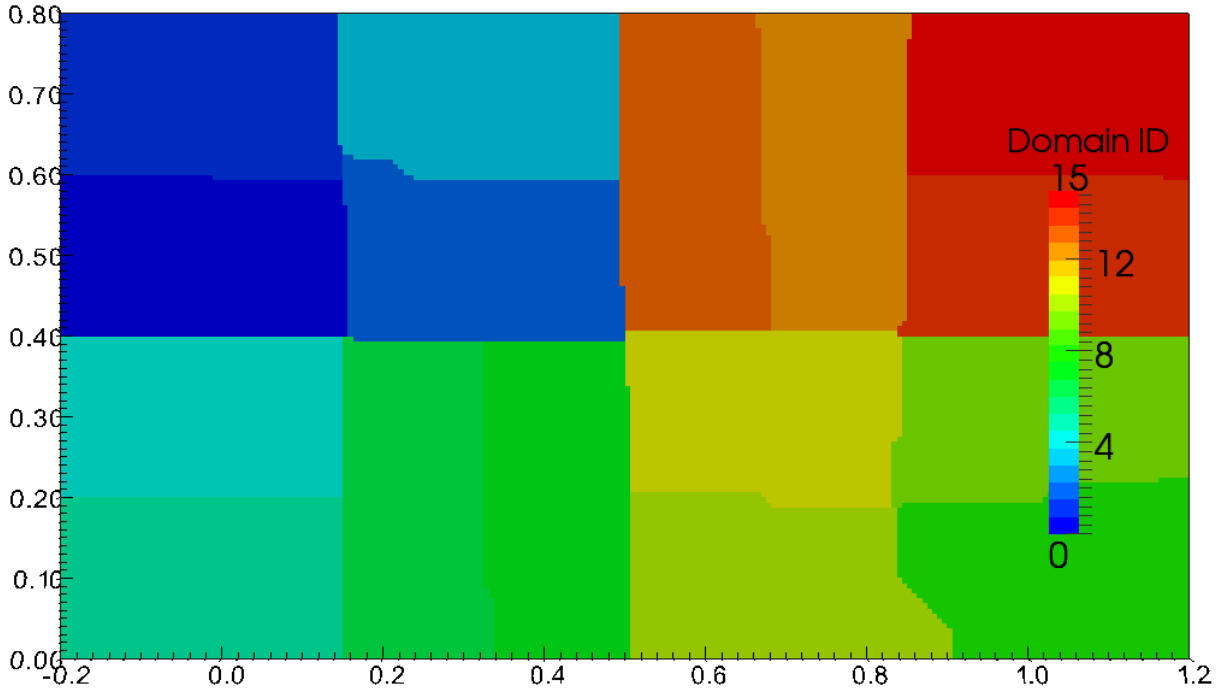



Figure 2.1: Carter flat plate domain and boundary conditions.




(a) 224x128 element mesh.




(b) 224x128 element mesh with 16 process domain decomposition.

Figure 2.2: Carter's flat plate problem computational mesh.

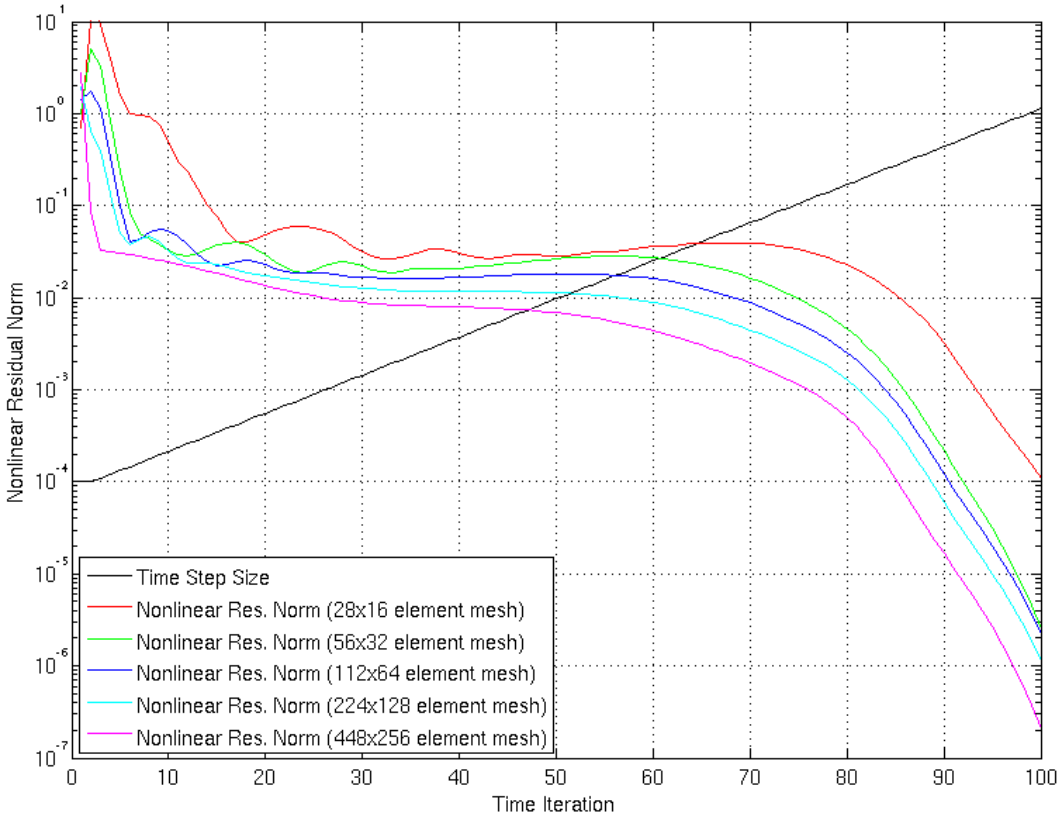



Figure 2.3: Carter's Mach 3.0 flat plate nonlinear residual convergence history.

very well with the contour plots shown in reference [125]. Due the low Reynolds number of this flow the boundary layer is quite pronounced and clearly evident in these plots.

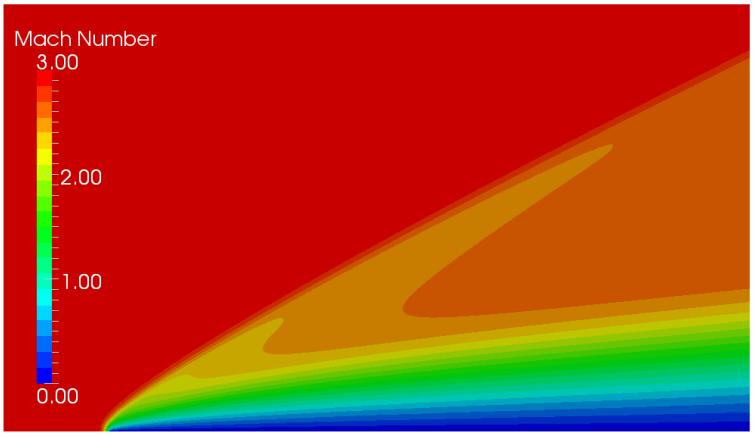
The pressure coefficient along the length of the plate is plotted in Figure 2.5 and has been computed by $C_p = (p - p_\infty)/0.5\rho_\infty v_\infty^2$. Also shown in this figure and several others that follow are the pressure, skin friction, and heat flux coefficient as computed by the Aria and Fluent computational fluid dynamics codes. The skin friction coefficient along the length of the plate is plotted in Figure 2.6 and is computed by $C_f = \tau_w/0.5\rho_\infty v_\infty^2$. The heat flux coefficient along the length of the plate is plotted in Figure 2.7 where the heat flux coefficient is computed by $C_h = -q_w/0.5\rho_\infty v_\infty^3$.

The pressure coefficient, skin friction coefficient, and heat flux coefficient all show good agreement with the results published by Shakib [124] and Carter [27]. It is interesting to note that the skin friction and heat flux coefficient profiles degrade significantly as the mesh coarsens. This is due to the fact that these quantities are dependent on the *gradient* of the solution and are computed using the basis functions of the finite elements that exist on the boundary. Thus, as the mesh is coarsened, the gradient approximation becomes less and less accurate.

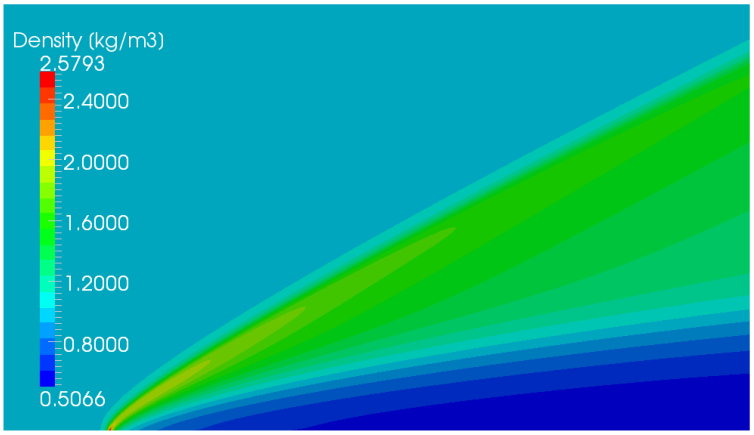
Table 2.1 shows non-dimensionalized run times for the 5 mesh refinement levels used for this problem when run in both a serial and parallel. Every effort was made to make the comparison as meaningful as possible, which is a difficult task when using different codes on different computational platforms.

The CU/FEMDOC and SNL/Aria codes are quite similar, both using stabilized finite element methods. The same time integration scheme, time increment, number of time iterations, Newton relaxation, and iterative linear solver parameters were specified. As previously mentioned both codes were run for 100 time iterations which corresponded to a 8-order of magnitude reduction in the nonlinear residual.

The ANSYS/Fluent code represents a completely different numerical approach as it is a finite volume method and takes CFL based approach toward integration to steady-state. Fluent, however, is an industry standard code well known for its computational performance and thus makes an interesting point of reference for comparison purposes. Fluent was run with the highest

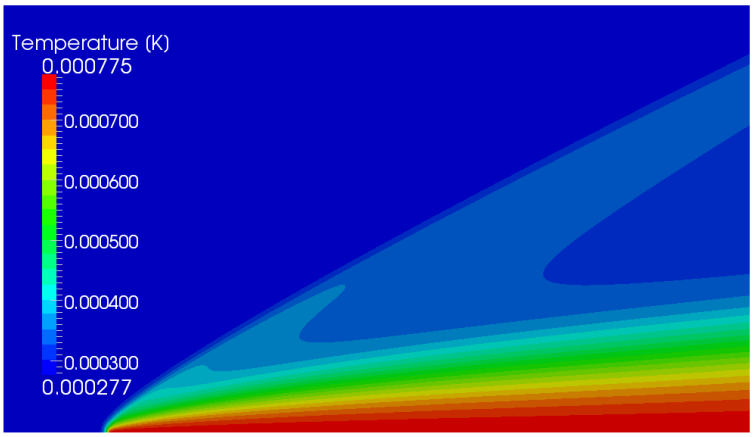


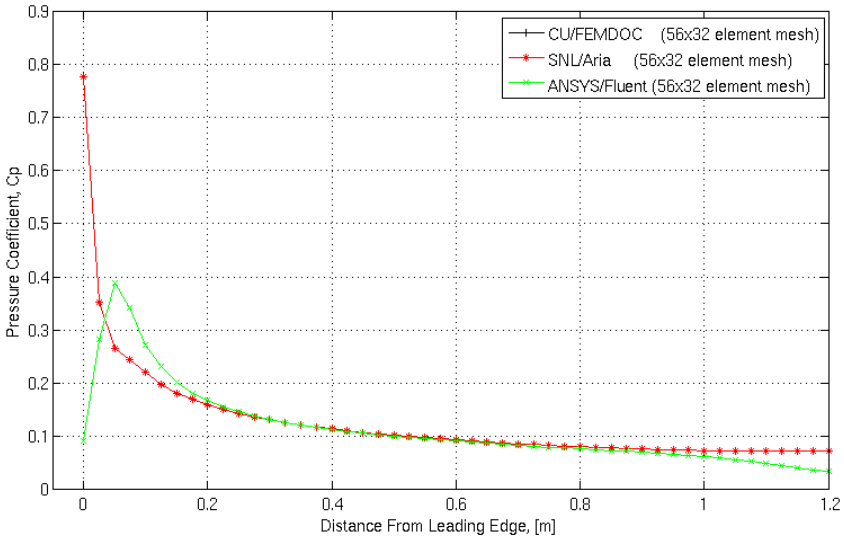

(a) Mach number contours.



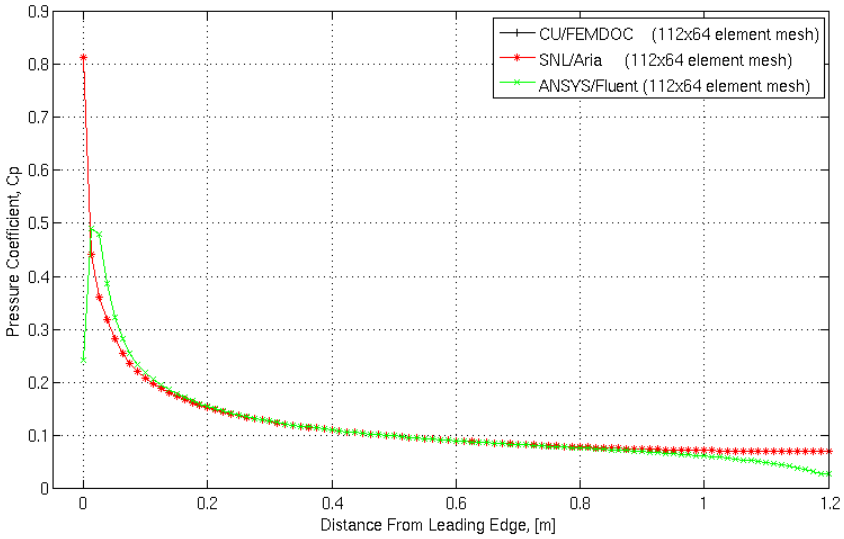

(b) Density contours.


(c) Pressure contours.

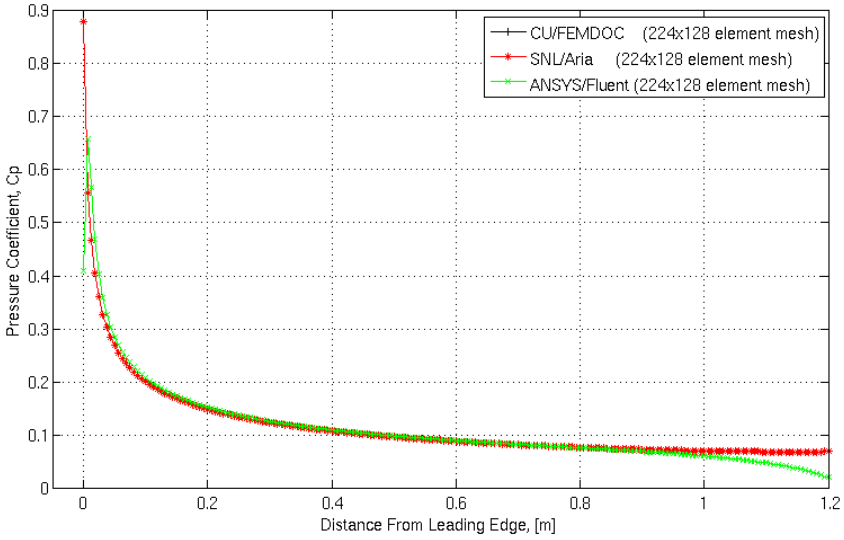




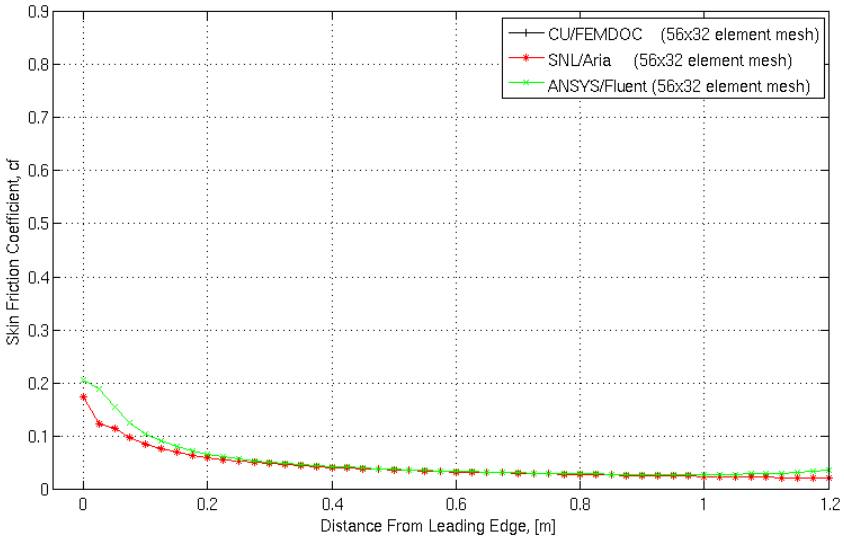

(a) 56x32 element mesh.



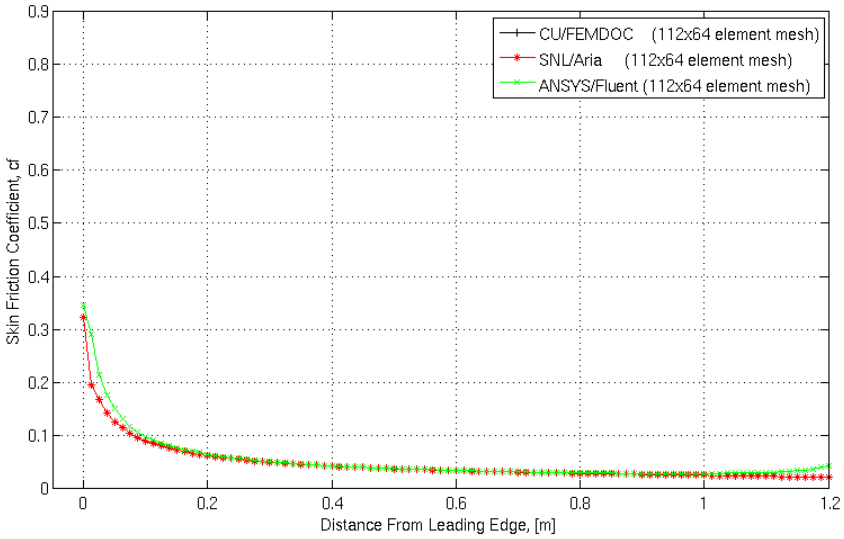

(b) 112x64 element mesh.



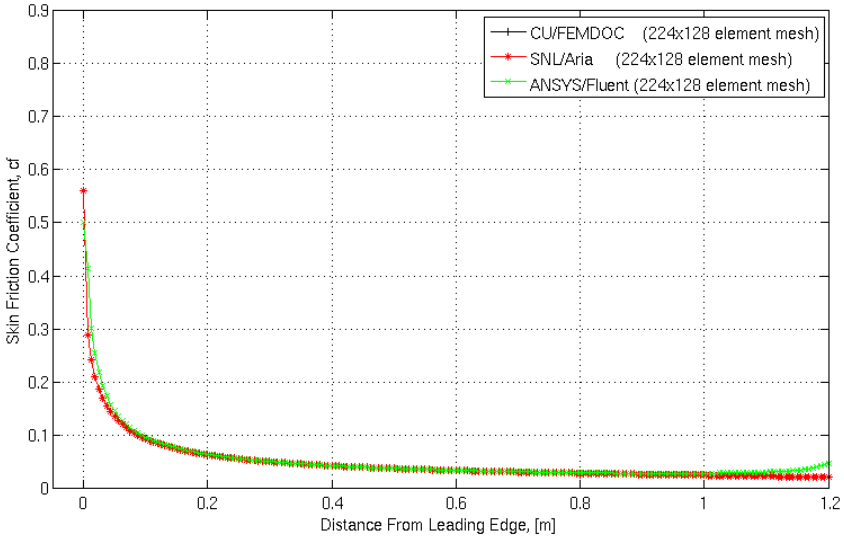

(c) 224x128 element mesh.




(a) 56x32 element mesh.




(b) 112x64 element mesh.




(c) 224x128 element mesh.

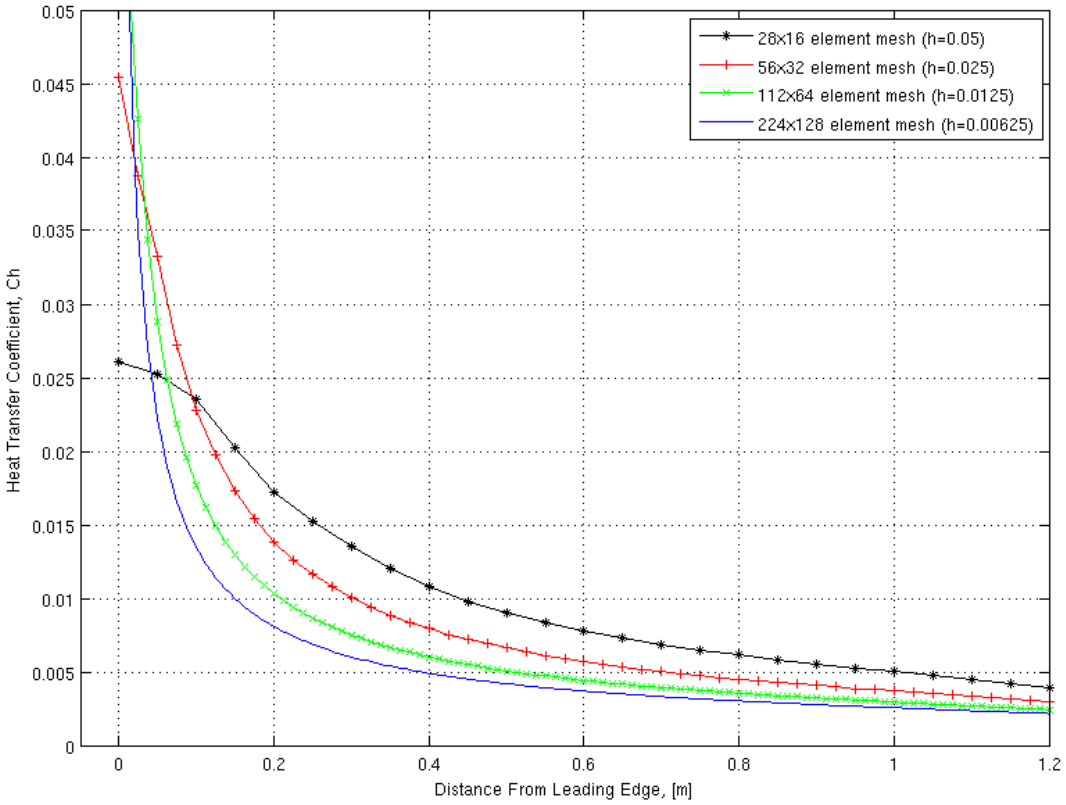



Figure 2.7: Wall heat flux coefficients for Carter's flat plate problem.

tolerable CFL number (approximately 100,000 for most problems) and run until each equations residual had dropped 8-orders of magnitude.

Since the individual codes were run on different computers, the run times needed to be scaled to account for the differences in overall system performance. This was accomplished by running a simple matrix inversion benchmark test on each of the different computers and determining a CPU specific scaling factor. Clearly, this approach does not account for several other critical aspects of the system performance such as network speed or I/O performance, but at least provides a simple quantitative measure of the CPU and memory subsystem performance.The following formula is used in an attempt to scale the results that have been run on different platforms as well as non-dimensionalize the overall run times.

$$Nondimensional\ time = \frac{t_{sim}}{t_{sim}^{ref}} \frac{t_{benchmark}^{ref}}{t_{benchmark}^{CPU}} \cdot 100 \tag{2.153}$$

Using Fluent as the point of reference for comparison purposes, the CU/FEMDOC code has run-times that are approximately 1.5 to 6 times longer than Fluent. The SNL/Aria code has run-times that range from approximately 3 to 40 times longer than Fluent. Finite volume methods are renowned for their low computational cost so the fact that the finite volume scheme computationally outperforms the finite element method is not surprising.

### 2.6.2 2-Dimensional Compression Corner

Another classic problem that is often used for computational validation and verification is Holden's compression corner [58]. Holden performed experimental studies of a supersonic compression corner and several researchers have used the data for computational code validation since. The purpose of this test problem is to compute and compare pressure, skin friction, and heat transfer coefficients along the compression corner geometry with other numerical solutions as well as Holden's experimental data.

#### 2.6.2.1 Problem Description

The problem consists of a $M_\infty = 11.68$, $Re = 248,600$ flow over a 15° compression corner. Figure 2.8 shows the physical domain and boundary conditions for the problem. Similar to the previous problem all dimensions and boundary conditions are specified non-dimensionally. The domain boundaries span the area $-1.74 \leq x \leq 28.0$ and $0.0 \leq y \leq 8.34$ with the 15° ramp beginning at $x = 17.4$.

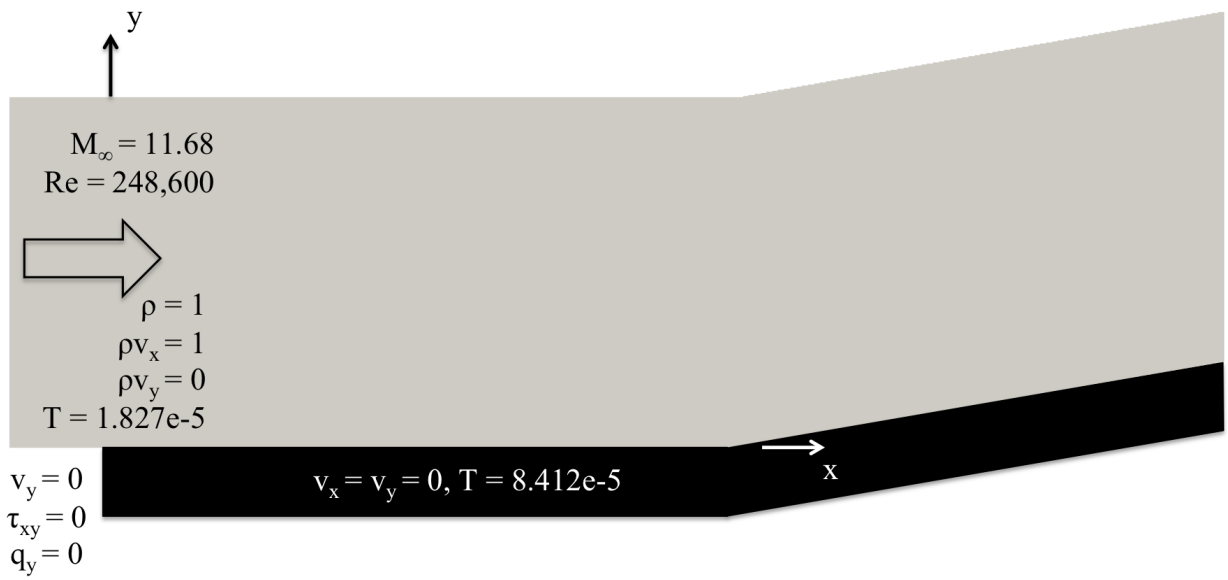



Figure 2.8: Supersonic compression corner geometry and boundary conditions.

The inflow boundary conditions are completely prescribed on this surface ($\rho = 1.0\ kg/m^3, V_\infty = 1\ m/s, T = 8.412 \times 10^{-5}\ K$). The slip boundary lying on the axis just forward of the flat plate portion of the domain imposes the conditions $u_y = \tau_{xy} = q_y = 0$. The wall boundary satisfies the no-slip condition of $u_x = u_y = 0$ with an isothermal wall with temperature $T_w = 8.412 \times 10^-5$. The outflow boundary has no prescribed variables, however, the boundary term appearing in equation 2.38 is appropriately integrated.

#### 2.6.2.2 Computational Mesh and Model

This problem is solved using three mesh levels. The coarsest mesh is grid of 39x104 quadrilateral elements, the mid-level mesh is composed of 78x208 elements, and the finest mesh contains 156x416 elements. Figure 2.9 shows the 39x104 quadrilateral element mesh. The meshes are not perfectly nested as they are in the previous example, but rather constructed to bias the mesh refinements towards the wall since the Reynolds number of this problem is higher and the boundary layer is much thinner. The initial element height from the compression ramp wall is 0.0056 m, 0.0011 m, and 0.00026 m for the 39x104, 78x208, and 156x416 element meshes, respectively.

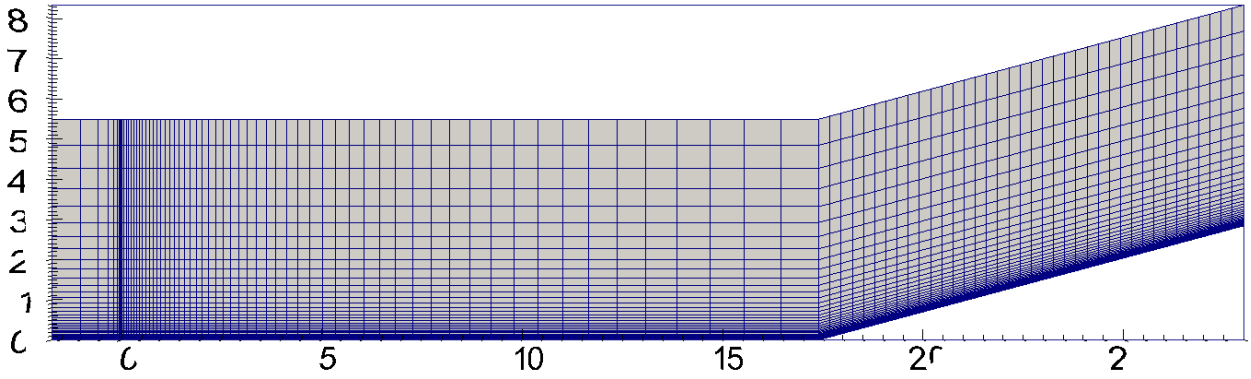



Figure 2.9: Supersonic compression corner 39x104 quadrilateral element mesh.

#### 2.6.2.3 Results and Discussion

Figure 2.3 shows the nonlinear residual norm and the time step sizes as the problem is integrated toward steady-state. Convergence is declared when the initial nonlinear residual norm for each Newton solve has dropped eight orders of magnitude from the reference nonlinear residual norm. For this problem the reference nonlinear residual is that from the second time iteration, which is the maximum value the norm takes during the time integration.

Figures 2.11(a) - 2.11(d) shows the Mach number, density, pressure, and temperature contours for the 156x416 element mesh. As in the previous example, these contour plots compare qualitatively very well with the contour plots shown in reference [125], as well as reference [82].

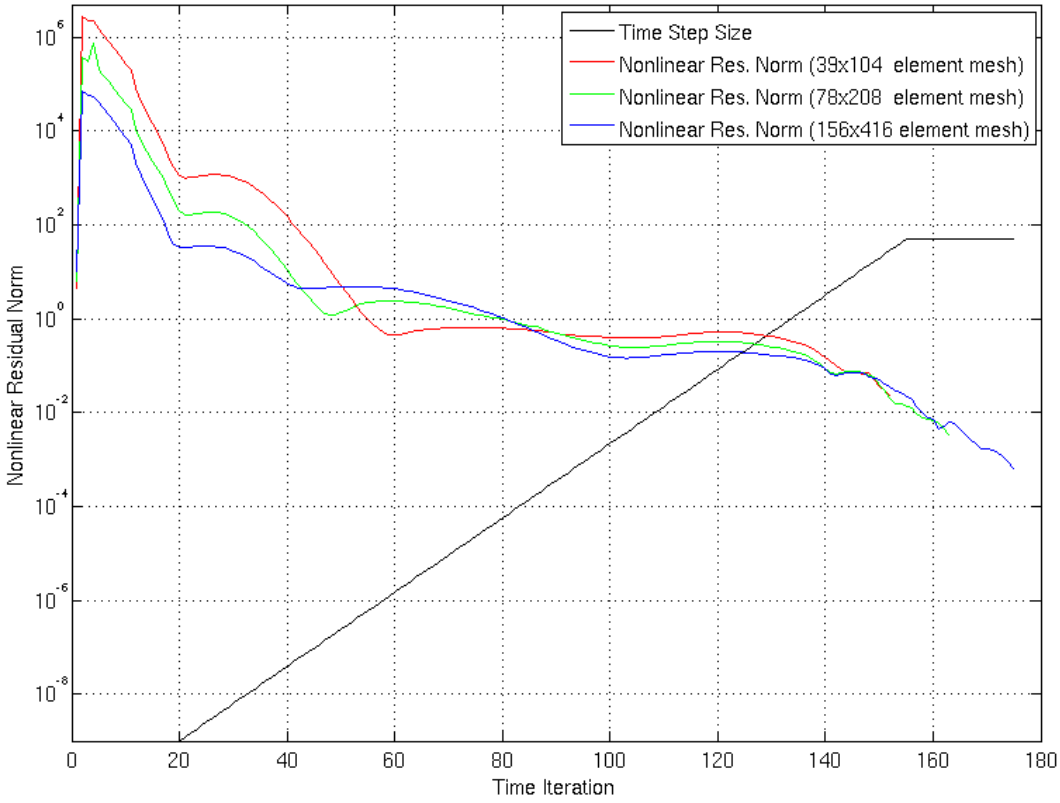
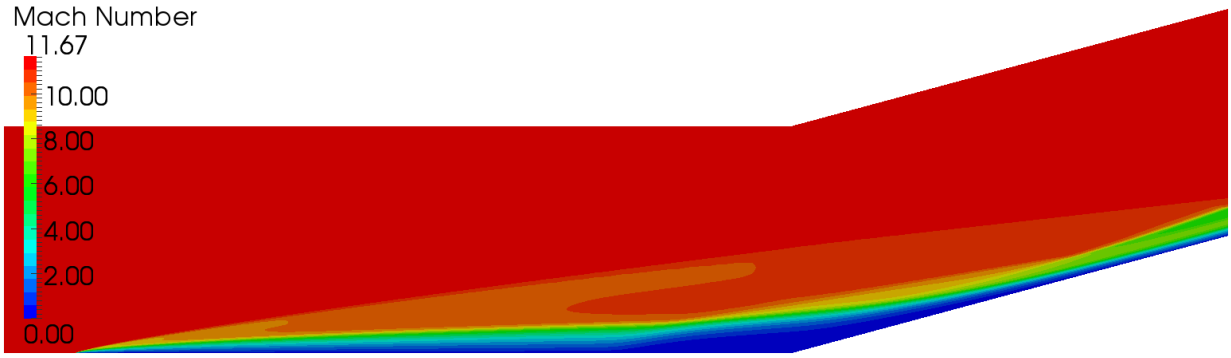


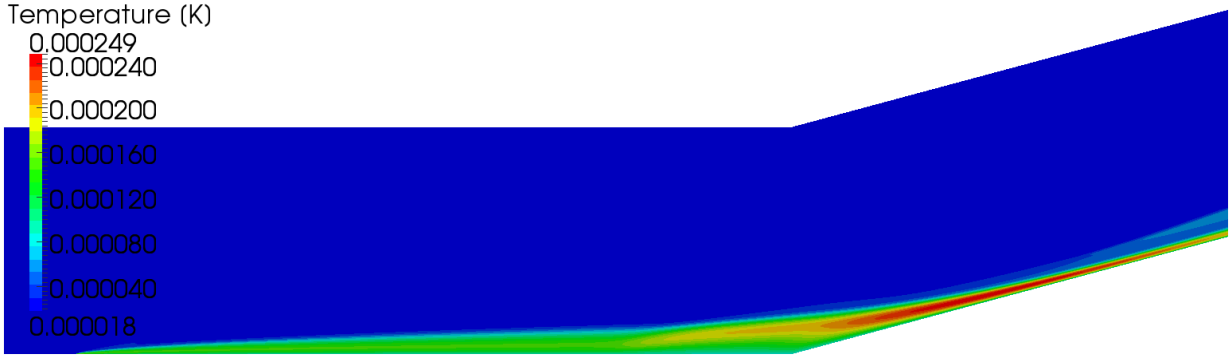

Figure 2.10: Holden's Mach 11.68 compression corner nonlinear residual convergence history.




(a) Mach number contours.


(b) Density contours.


(c) Pressure contours.




(d) Temperature contours.

Figure 2.11: Holden's Mach 11.68 compression corner solution contours (156x416 element mesh).

The pressure coefficient along the length of the plate is plotted in Figure 2.12 where the pressure coefficient is computed by $C_p = (p - p_\infty)/0.5\rho_\infty v_\infty^2$. A discrepancy between the solutions shown here and Shakib's results [125] exists for the pressure coefficient values between the leading edge of the plate and the start of the compression corner. Shakib's results converge to a $C_p$ value of approximately 0.02 before the start of the compression corner, while the solutions show here take $C_p$ values between 0.005 and 0.006 before the start of the compression corner. The skin friction coefficient along the length of the plate is plotted in Figure 2.13 where the skin friction coefficient is computed by $C_f = \tau_w/0.5\rho_\infty v_\infty^2$. The heat flux coefficient along the length of the plate is plotted in Figure 2.14 where the heat flux coefficient is computed by $C_h = -q_w/0.5\rho_\infty v_\infty^3$.

### 2.6.3 2-Dimensional Nose Tip

This example problem solves the supersonic flow about a two-dimensional nose tip geometry. The geometry and meshes used for this example come from an verification study used to assess the performance of various codes in use at Sandia National Laboratories among a few others.

In addition to the three codes described in the introduction to this section, three additions codes have been used for solution comparison purposes:

- SNL/Premo: Vertex centered unstructured finite volume code developed at Sandia National Laboratories. Premo's underlying theory is well documented in a conference proceeding [130].
- SNL/Saccara: Cell centered structured finite volume code that was adapted from INCA, a once commercially available CFD code. Information on Saccara is provided in a report published by staff members at Sandia National Laboratories [142].
- NASA/Overflow: Cell centered structured finite volume code developed at NASA. Information on Overflow can be found on its author's website [1].

Dr. Jeff Payne at Sandia National Laboratories generated the solutions shown in this section with these three different codes.

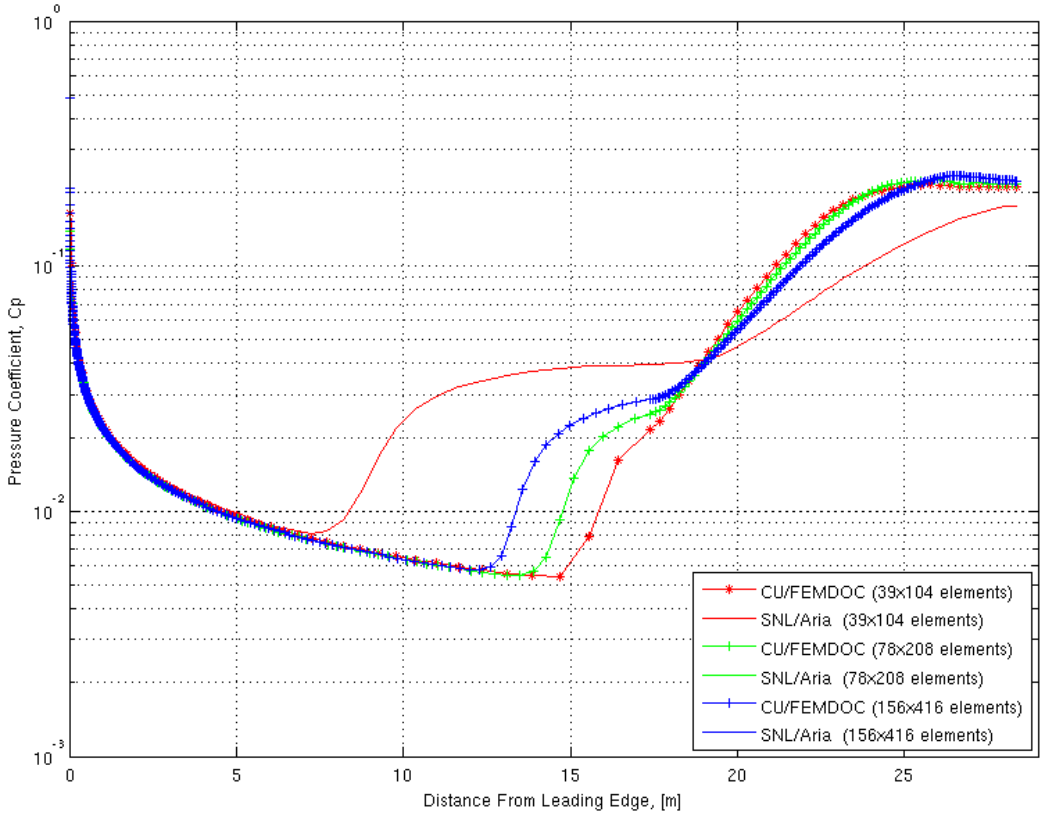



Figure 2.12: Wall pressure coefficients for Holden's compression corner problem.

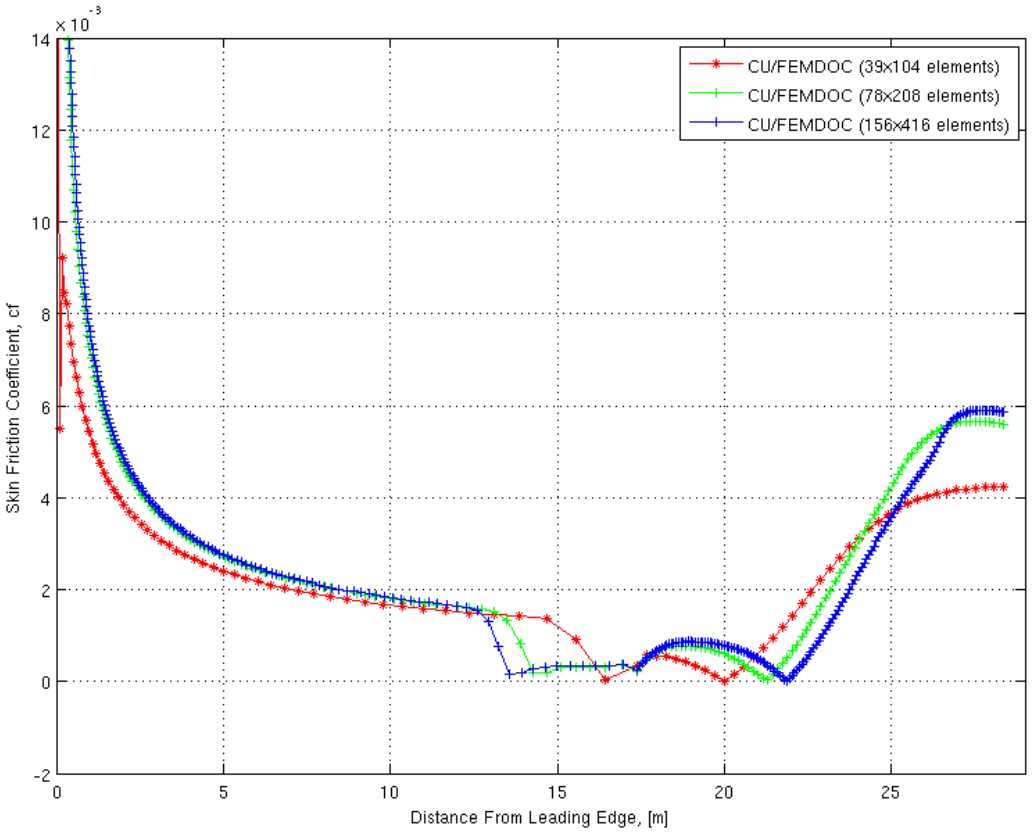



Figure 2.13: Wall skin friction coefficients for Holden's compression corner problem.

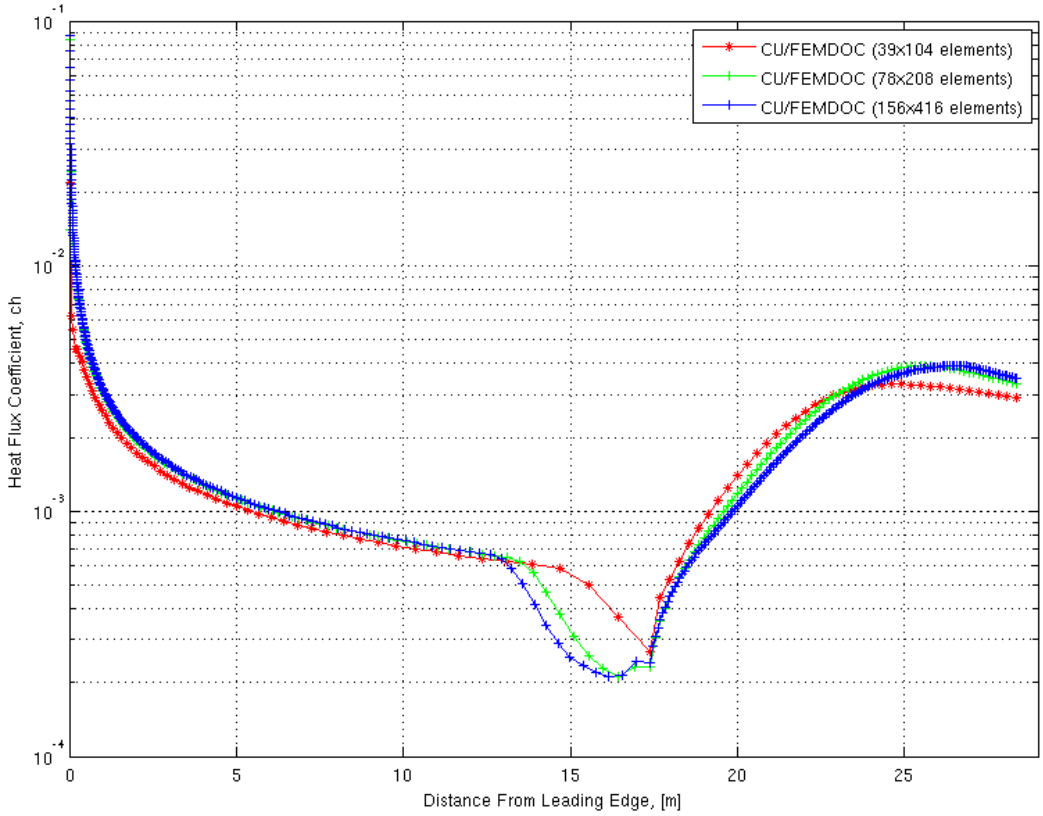



Figure 2.14: Wall heat flux coefficients for Holden's compression corner problem.

#### 2.6.3.1 Problem Description

The flow conditions for this problem are that of a $M_\infty = 3.0$ flow at 40 $km$ altitude where $\rho_\infty = 3.99641 \times 10^{-3}\ kg/m^3$ and $T_\infty = 250.35\ K$. Freestream flow values for $\rho, \rho v_x, \rho v_y$, and $\rho E$ are prescribed on the inlet boundary. Slip boundary condition ($v_y = 0$) is used for the edge lying on the x-axis. A no-slip adiabatic wall boundary condition is used for the surface of the body. No values are prescribed for the outflow edge and the viscous fluxes are integrated to be consistent with the integrated-by-parts weak form of the viscous fluxes.

The gas constants for air are specified as $R = 287.0$ and $\gamma = 1.4$. The two-coefficient Sutherland model for air is used to compute the viscosity ($\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110.4\ K$) and the Prandtl number ($Pr = 0.70$) is used to compute the thermal conductivity of the air.

#### 2.6.3.2 Computational Mesh and Model

Three meshes are used for the purpose of this study. The coarsest mesh, shown in Figure 2.16(a), is a structured 37x40 quadrilateral element domain and contains a total of 1722 nodes. The medium level mesh contains 74x80 quadrilateral elements and has a total of 6723 nodes. This mesh was generated by taking the coarsest mesh and applying a 1-level uniform refinement (i.e. 1 quadrilateral is equally divided into 4 new quadrilaterals). The finest mesh contains 148x160 quadrilateral elements and has a total of 26,565 nodes. This mesh was generated by taking the coarsest mesh and applying a 2-level uniform refinement (i.e. 1 quadrilateral is equally divided into 16 new quadrilaterals).

The backward Euler time integrator is used with adaptive time stepping where an initial time step size of $1 \times 10^{-7}\ s$ and a maximum time step increase of 1.2 are used. Newton's method with an approximate linearization is used to solve the nonlinear problem at each time step. A nonlinear relaxation factor of 0.9 is used with a nonlinear residual drop criteria of $\epsilon = 0.1$. The Trilinos/Aztec GMRES iterative solver is used for solving the linear problem at each nonlinear

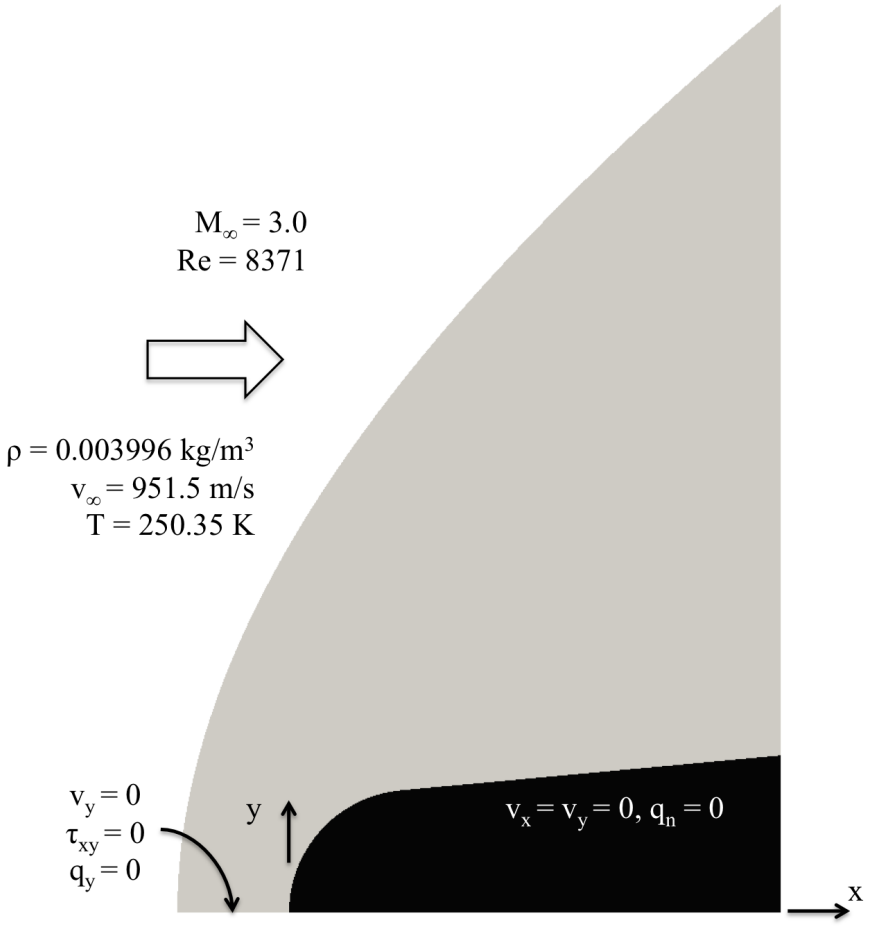



Figure 2.15: Three dimensional supersonic nose tip geometry and boundary conditions.

step. A global ILU(0) preconditioner with a linear residual drop criteria of $\epsilon = 1 \times 10^{-4}$ is used. The SUPG term in the weak residual equation uses node averaged stabilization and $\delta$ discontinuity capturing parameters. The nodally averaged discontinuity parameter in particular is important for computing a smooth shock boundary.

#### 2.6.3.3 Results and Discussion

Nonlinear residual convergence behavior for the three meshes is plotted in Figure 2.17 against the time iteration count. The time step is ramped up exponentially, as shown on this figure. The nonlinear residual norm begins a gradual descent during the small time step iterations early on in the simulation. In all three cases, a breaking point occurs between time iteration 60 - 70 and $\Delta t = 1 \times 10^{-4}$ $s$ where the nonlinear residual norm begins to decrease exponentially. The exact behavior of the residual's descent at this point varies with the refinement of the mesh, with the higher degree of refinement resulting in a slower convergence rate.

Figures 2.18 show contour plots of Mach number, density, pressure, and temperature, respectively, for the finest mesh.

Figure 2.19 shows the static pressure along the surface of the nose tip as computed by the various codes used for this study. The stagnation pressure can be computed analytically from the equation

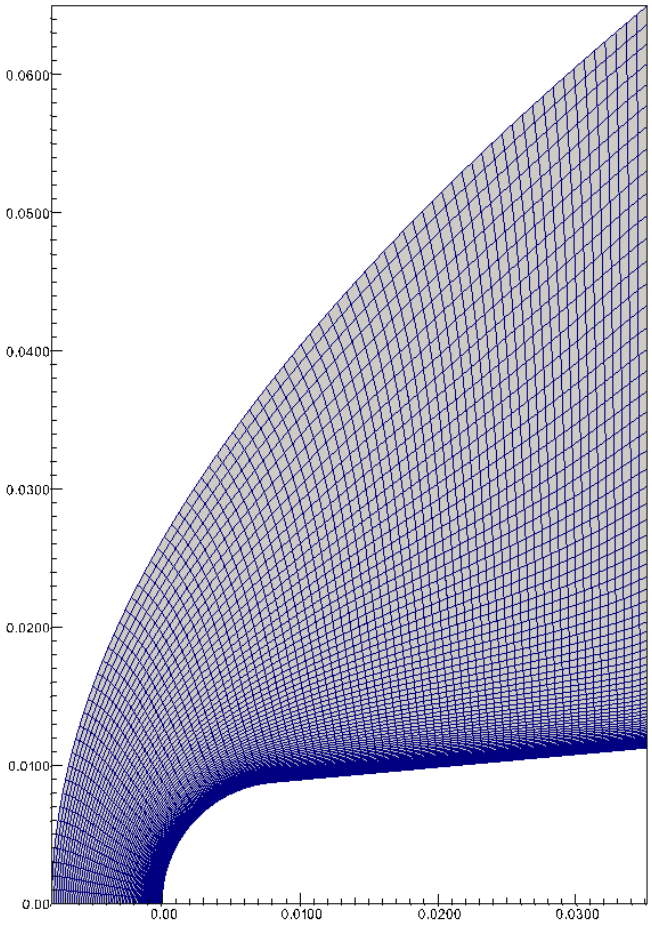
$$P_0 = P_\infty \left( \frac{(\gamma+1)^2 M_\infty^2}{4\gamma M_\infty^2 - 2(\gamma-1)} \right)^{\frac{\gamma}{\gamma-1}} \frac{1-\gamma+2\gamma M_\infty^2}{\gamma+1} \tag{2.154}$$

and substituting the freestream pressure and Mach number values the theoretical perfect gas stagnation pressure should be 3463 $Pa$.

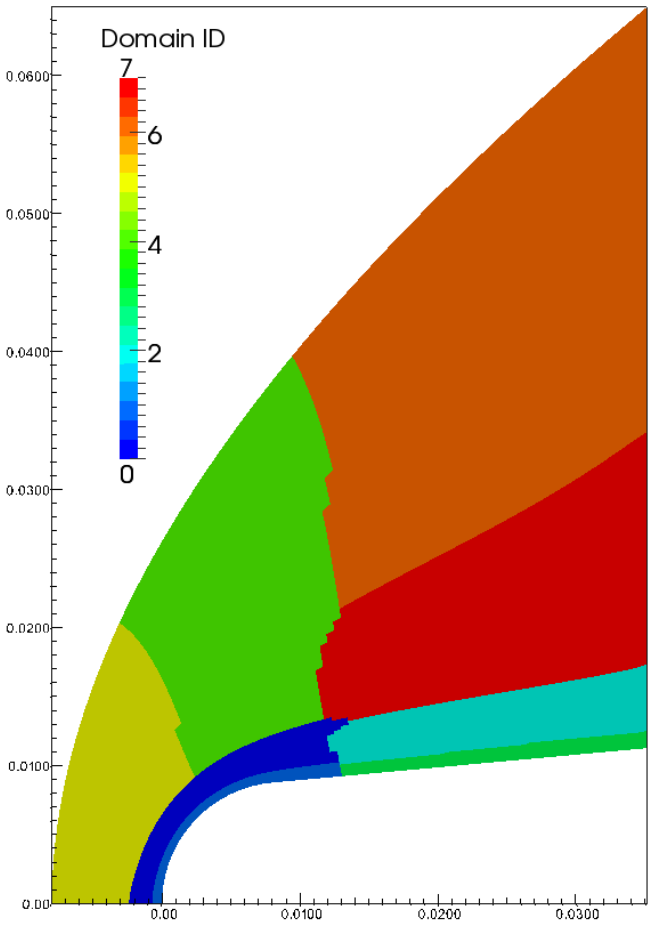
Figure 2.20 shows the temperature along the surface of the nose tip as computed by the various codes used for this study. The stagnation temperature can be computed analytically from the equation

$$T_0 = T_\infty \left( 1 + \frac{\gamma-1}{2} M_\infty^2 \right) \tag{2.155}$$

and substituting the freestream temperature and Mach number values the theoretical perfect gas




(a) 74x80 element mesh.




(b) 74x80 mesh with 8 process domain decomposition.

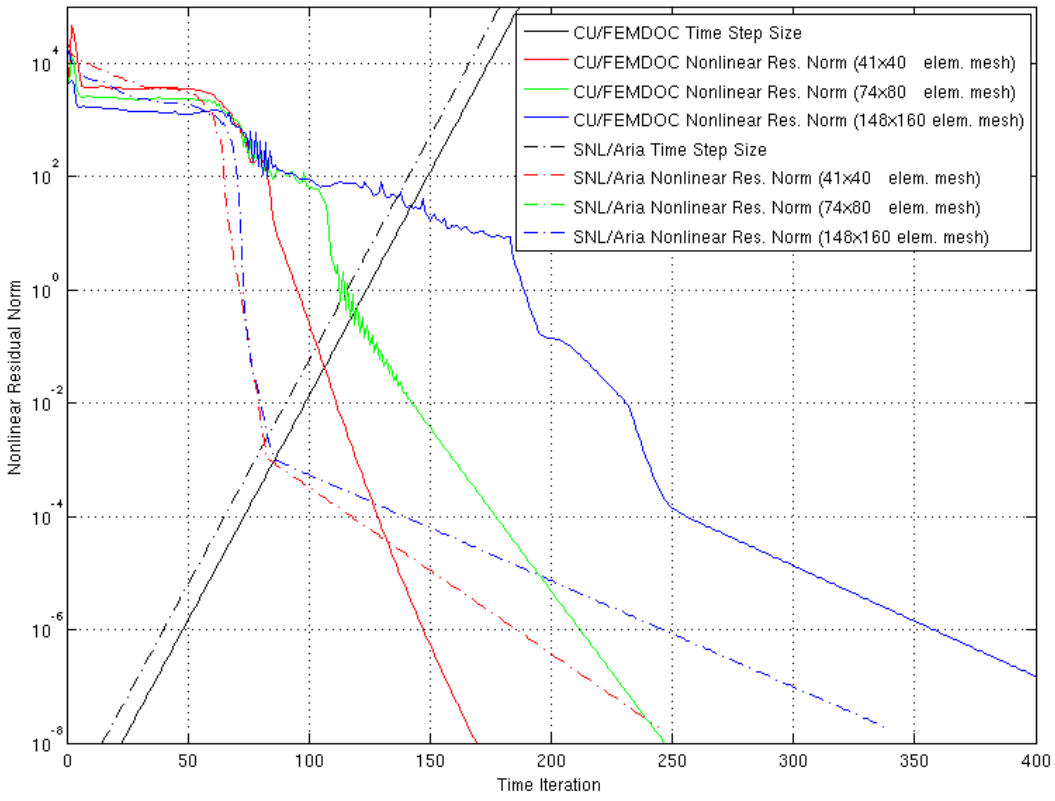
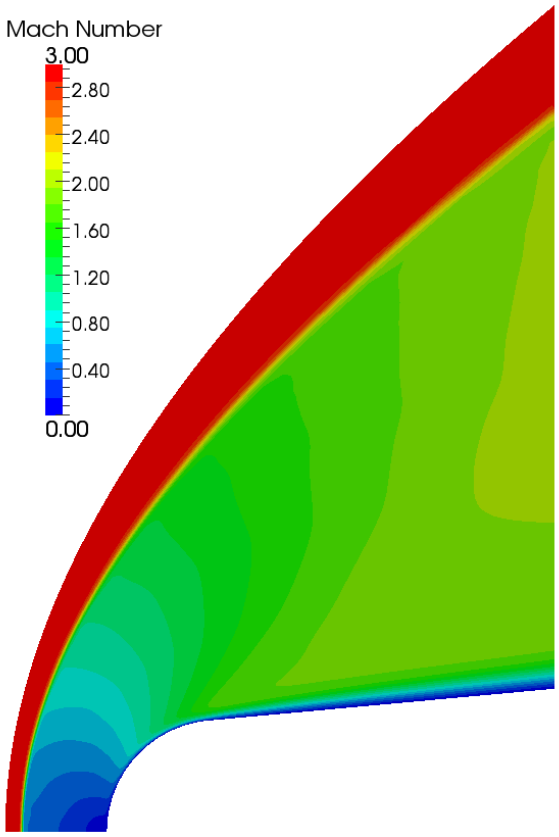
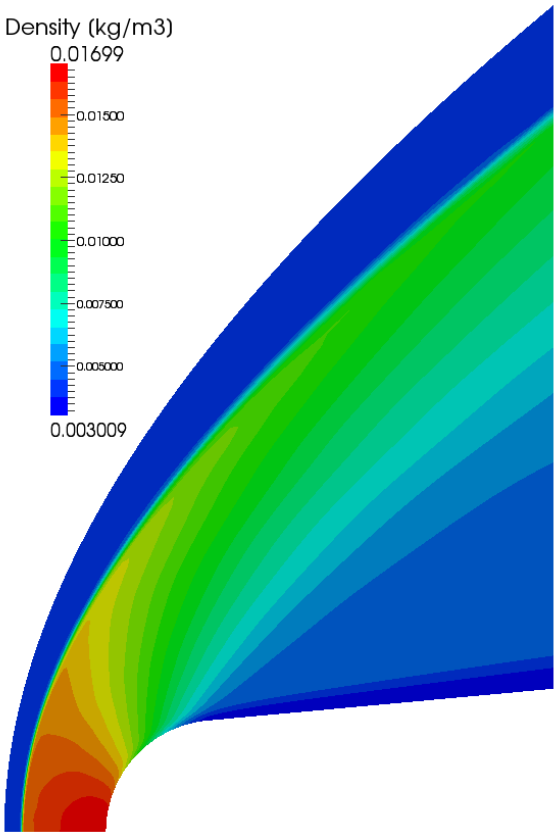


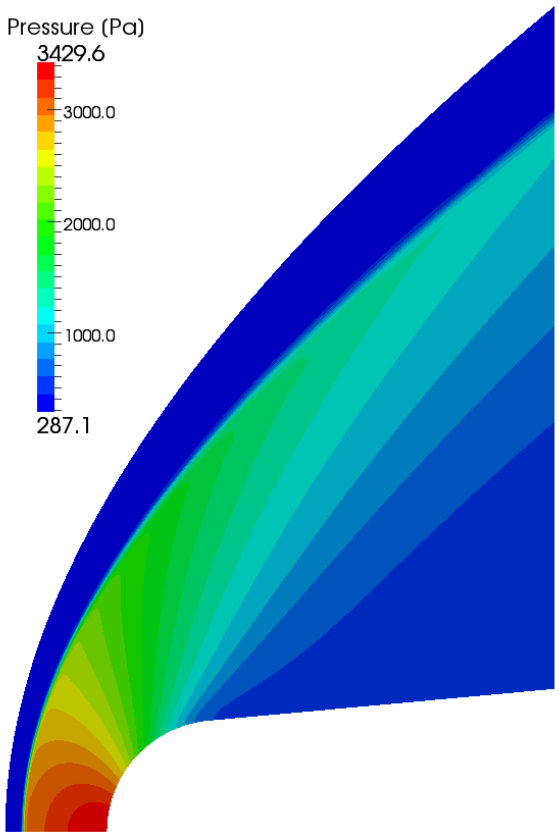

Figure 2.17: Nonlinear residual convergence history for the nose tip problem.



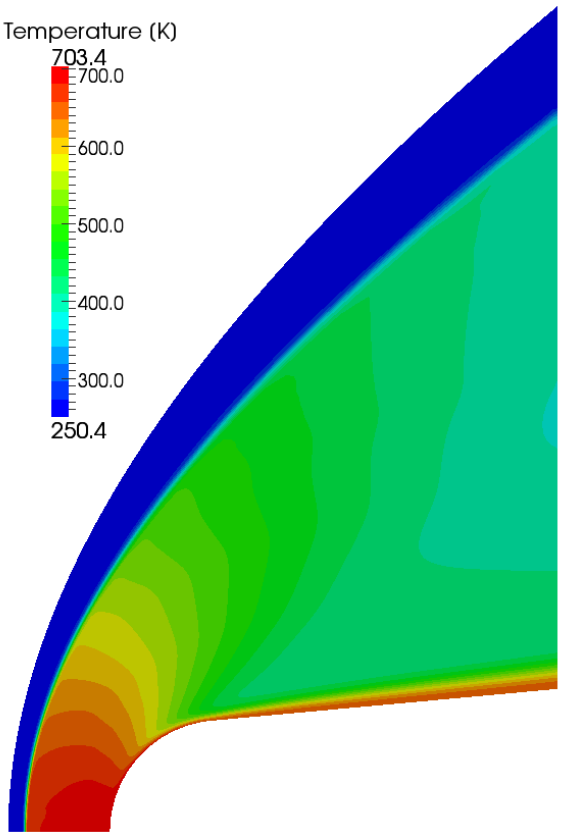

(a) Mach number contours.




(b) Density contours.

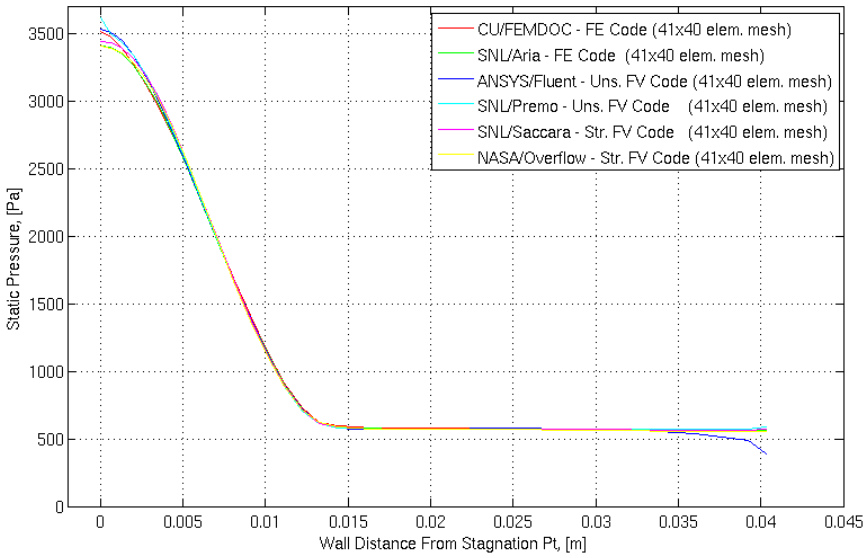



(c) Pressure contours.

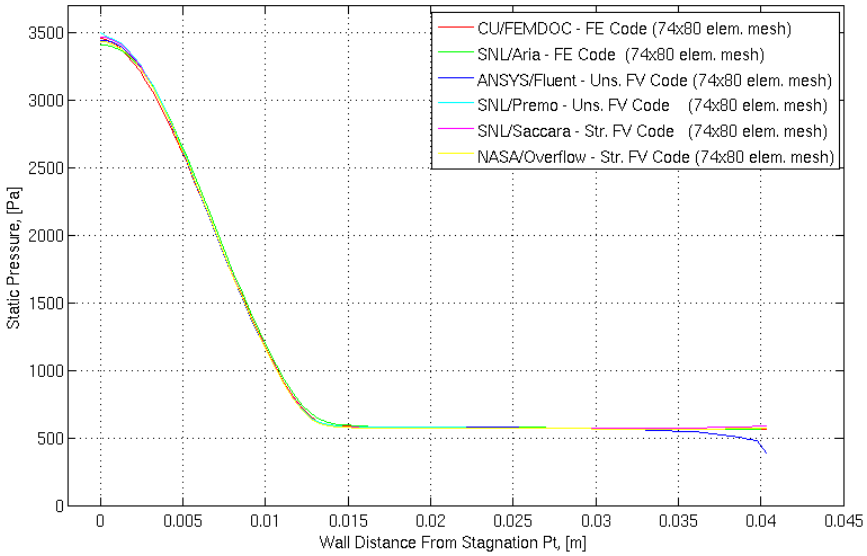



(d) Temperature contours.

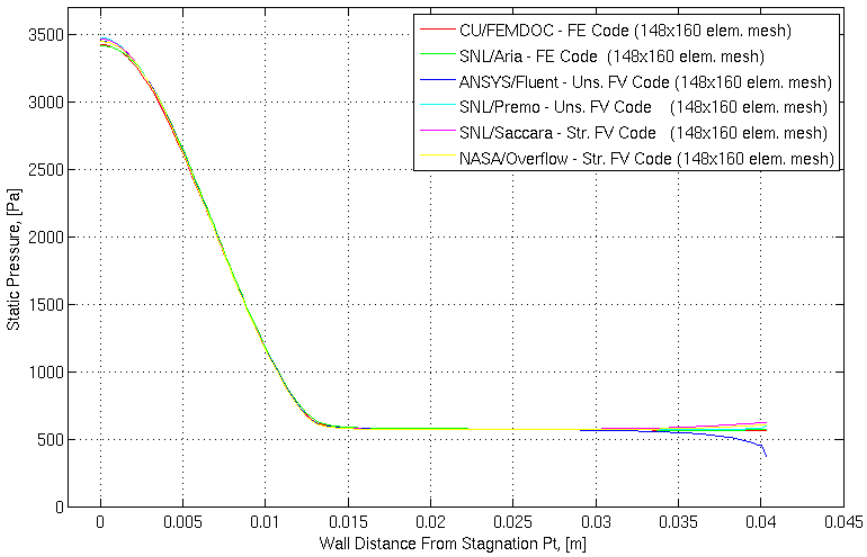
Figure 2.18: Mach 3.0 nose tip solution contours (148x160 element mesh).




(a) 41x40 element mesh.




(b) 74x80 element mesh.



stagnation temperature should be 700.98 $K$.

Figure 2.21 shows the XY shear stress along the surface of the nose tip as computed by the various codes used for this study. Note that these values are the stress values with respect to the Cartesian X & Y coordinates and not the shear stress values tangent to the surface.

Table 2.2 shows the stagnation temperatures and pressures as computed by the analytical equations and as computed the various codes.

### 2.6.4 2-Dimensional Pitching NACA0012 Airfoil

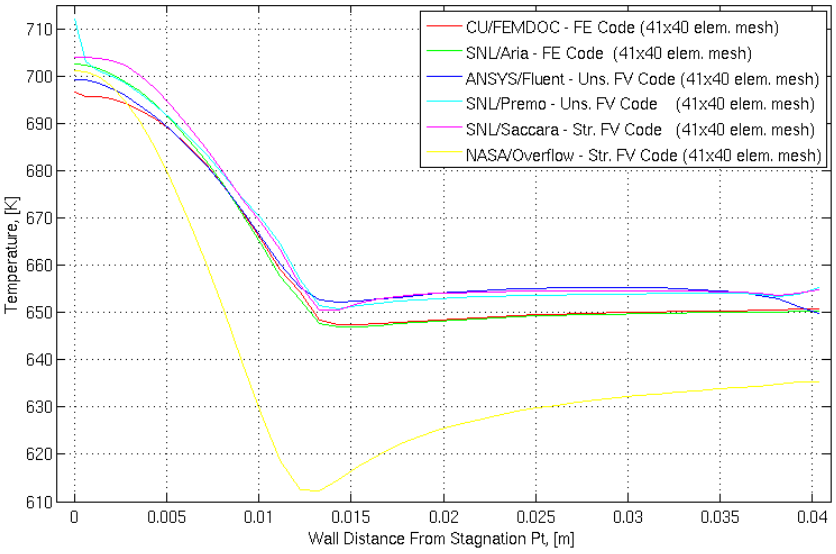
This problem demonstrates the ability of the finite element solver for computing viscous flow solutions on moving meshes. The problem is modeled after one studied by Aliabadi [4] and solves for the flow about a pitching NACA 0012 airfoil.

#### 2.6.4.1 Problem Description

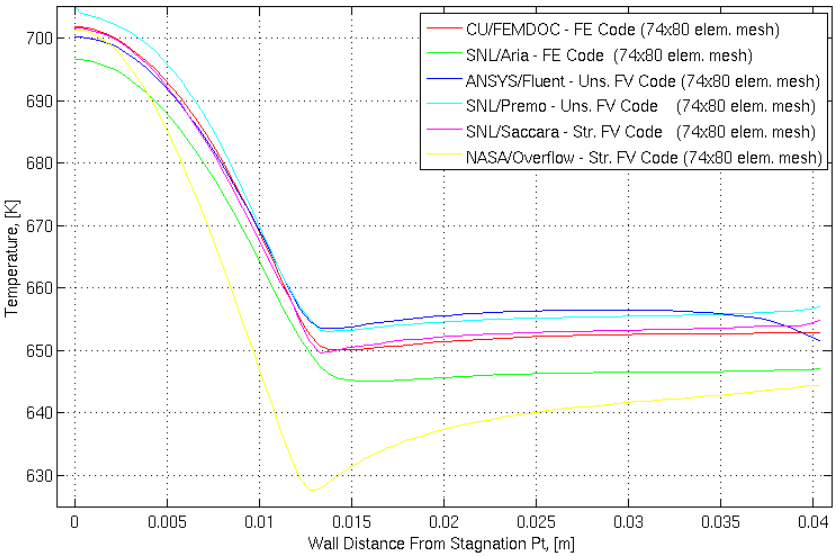
This problem consists of a $M_\infty = 0.2$, $Re = 1000$ flow around a NACA 0012 airfoil. Figure 2.22 shows the physical domain and boundary conditions for the problem. The domain boundaries span the area $-0.0009 \leq x \leq 0.0009\ m$ and $-0.0009 \leq y \leq 0.0009\ m$, and the chord length of the airfoil is $2.117^{-4}\ m$. The properties of the fluid are consistent with air and assumed be $\Gamma = 1.4$, $R = 287$ J/kg-K, $Pr = 0.72$, and the viscosity is modeled by Sutherland's three equation model 2.21 with $\mu_{ref} = 1.8 \times 10^{-5}\ kg/m/s$, $T_{ref} = 300\ K$, and $S = 110\ K$.

The inflow boundary conditions are take sea-level atmospheric values and since the flow is subsonic only the density and momentum terms are specified ($\rho = 1.225\ kg/m^3, V_\infty = 69.4\ m/s, T = 300.0\ K$). The outflow boundary condition requires that the pressure or alternately the temperature is known. For this problem the far-field outflow boundary condition is prescribed as 300 $K$. The wall boundary satisfies the no-slip condition of $u_x = u_y = 0$ with an adiabatic wall. The airfoil is pitched about its mid-chord point according the following equation

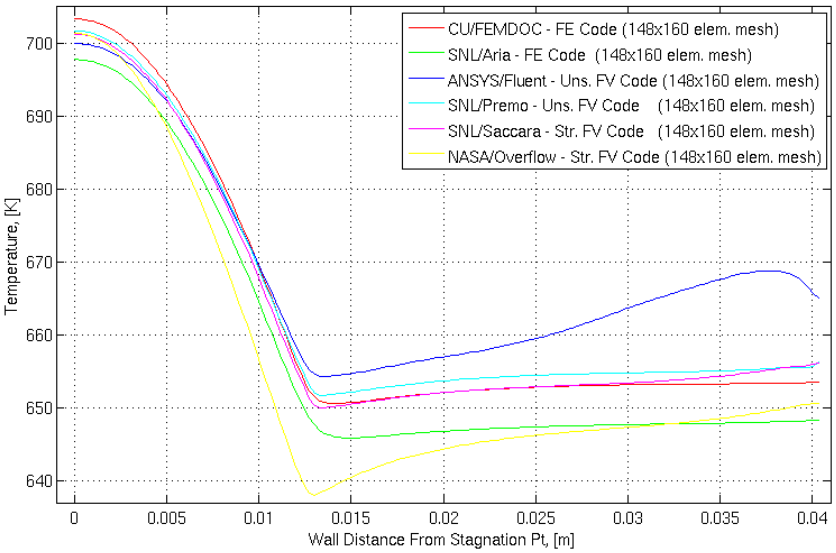
$$\alpha(t) = 10^\circ - 10^\circ cos(2\pi t) \tag{2.156}$$

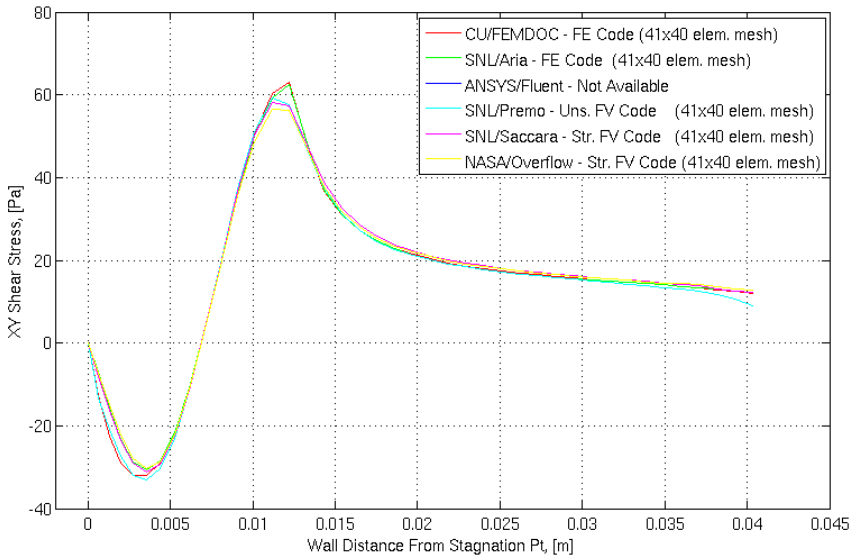



(a) 41x40 element mesh.

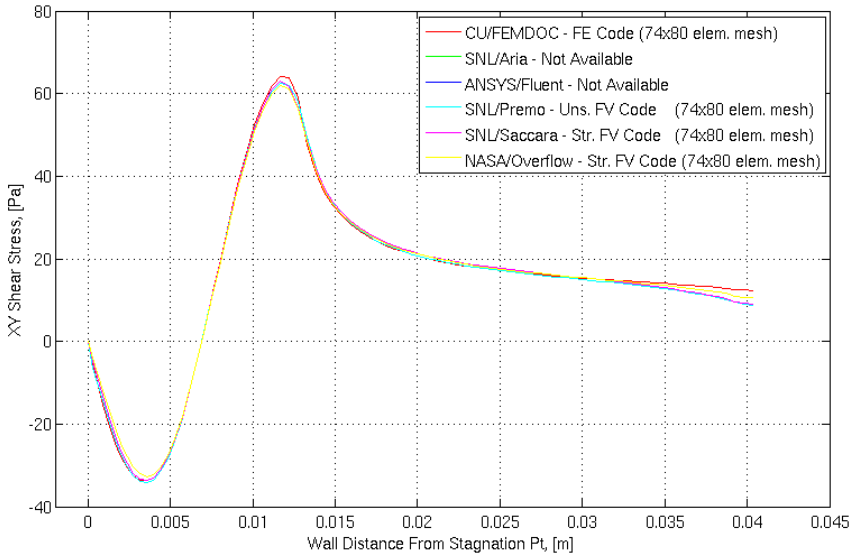



(b) 74x80 element mesh.

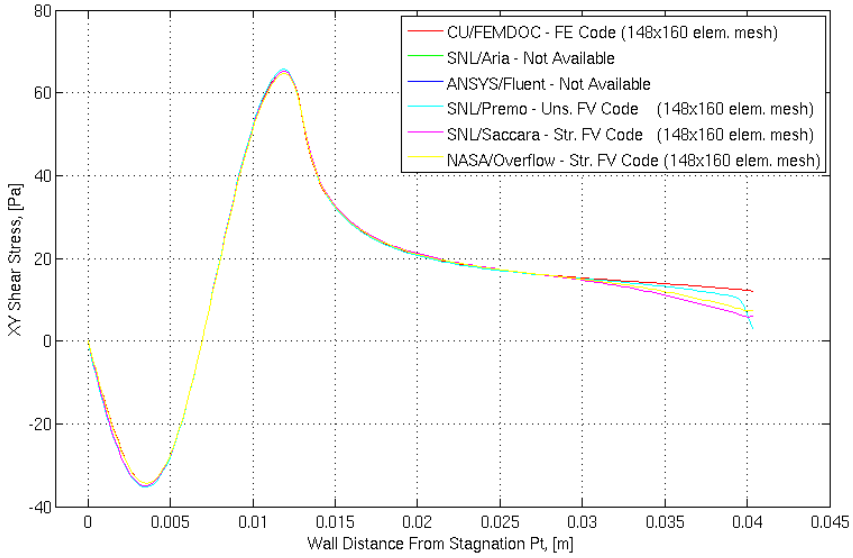





(a) 41x40 element mesh.




(b) 74x80 element mesh.



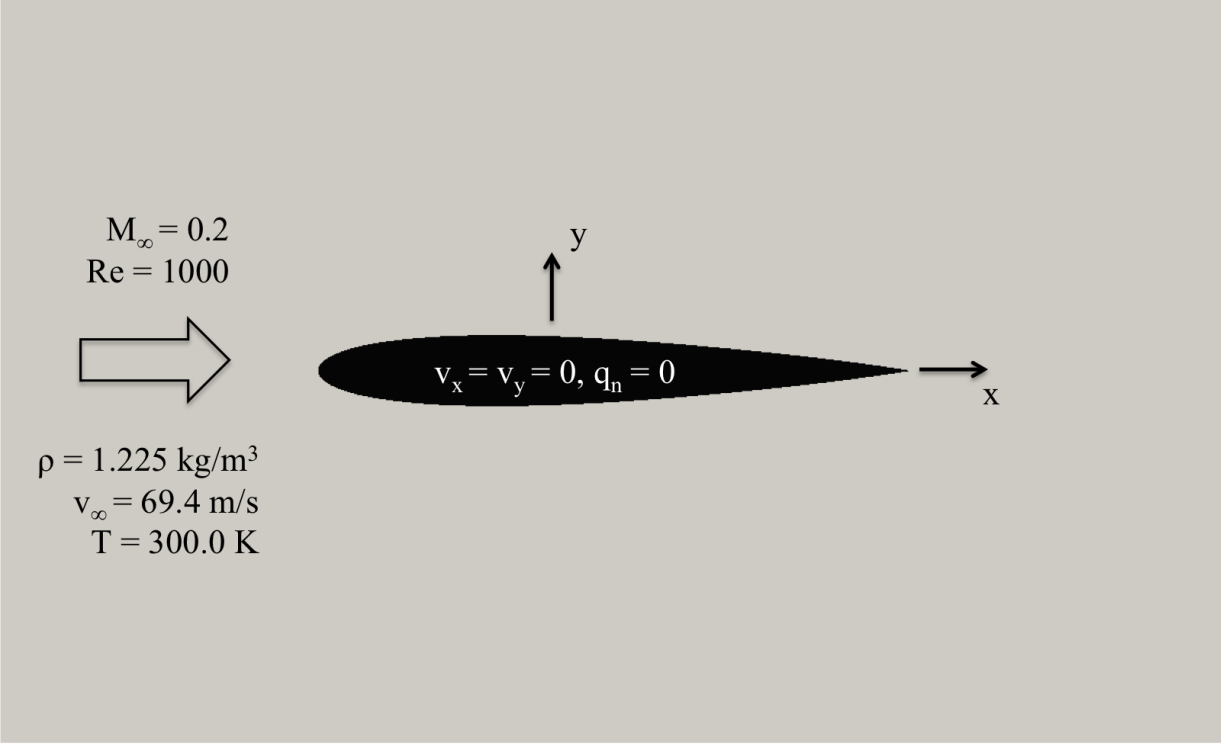



Figure 2.22: NACA 0012 airfoil geometry and boundary conditions.

#### 2.6.4.2 Computational Mesh and Model

The problem is solved using two meshes, one with 60x70 quadrilateral elements and another with 120x140 quadrilateral elements. Figures 2.23-2.25 shows the 60x70 element mesh.

The problem in integrated time to steady-state via a three-point backward Euler (BDF-2) scheme using a constant time step of 0.05 $s$. A Newton scheme is used to solve the nonlinear equations. The nonlinear iteration loop exits when the nonlinear residual drops at least three orders of magnitude or when the initial nonlinear residual norm is less than $1.0 \times 10^{-8}$. This problem utilizes the UMFPACK direct solver for solving the linearized system equations at each Newton step.

Standard element based stabilization is used instead of the nodally average stabilization parameter; given the low Mach number of this problem, both approaches are equally stable. No discontinuity capturing operator is needed or used for these simulations.

The effect of the pitching motion is generated in two different fashions. In the first case, the mesh motion is generated by treating the fluid mesh as an elastic solid and applying rigid body rotation to a block of elements surrounding the airfoil chord center according to equation 2.156. The displacements at the freestream boundaries are then fixed. The mesh deformations then occur in the elements that lay between the rigid body rotation and the fixed freestream boundary. This form of generating the pitching motion requires an appropriate arbitrary Lagrangian-Eulerian treatment due to the motion of the mesh. In the second case, the pitching motion is generated by leaving the airfoil wall fixed and modifying the inflow boundary conditions to simulate the $0^\circ$ to $20^\circ$ angle of attack. These two forms allow for conclusions to be draw about the accuracy of the ALE implementation.

#### 2.6.4.3 Results and Discussion

Figures 2.26 - 2.29 show the Mach number, density, pressure, and temperature contours for the pitching NACA 0012 airfoil at 20 degrees angle of attack when the pitching motion is generated

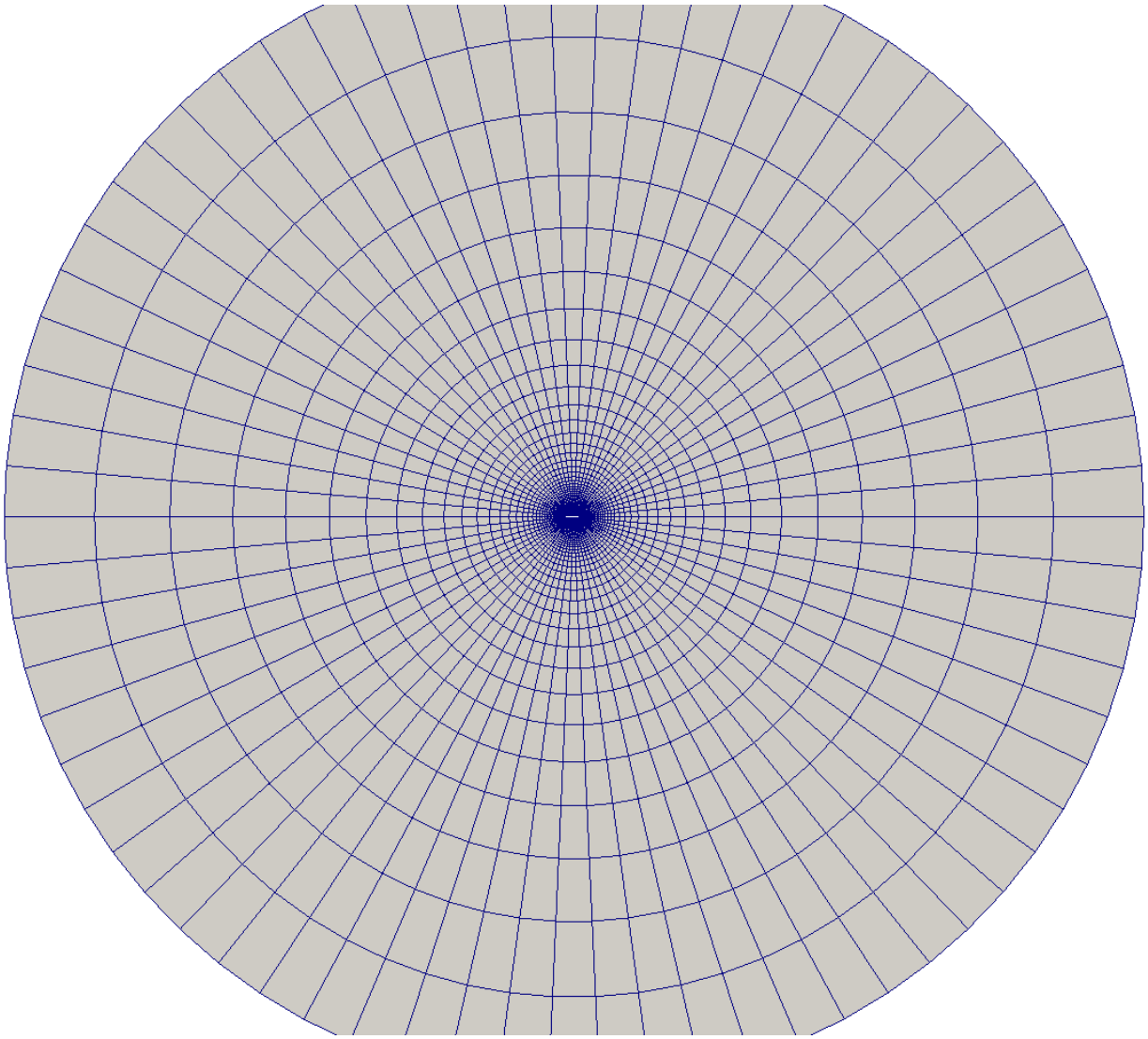


Figure 2.23: NACA 0012 airfoil 60x70 quadrilateral element mesh.

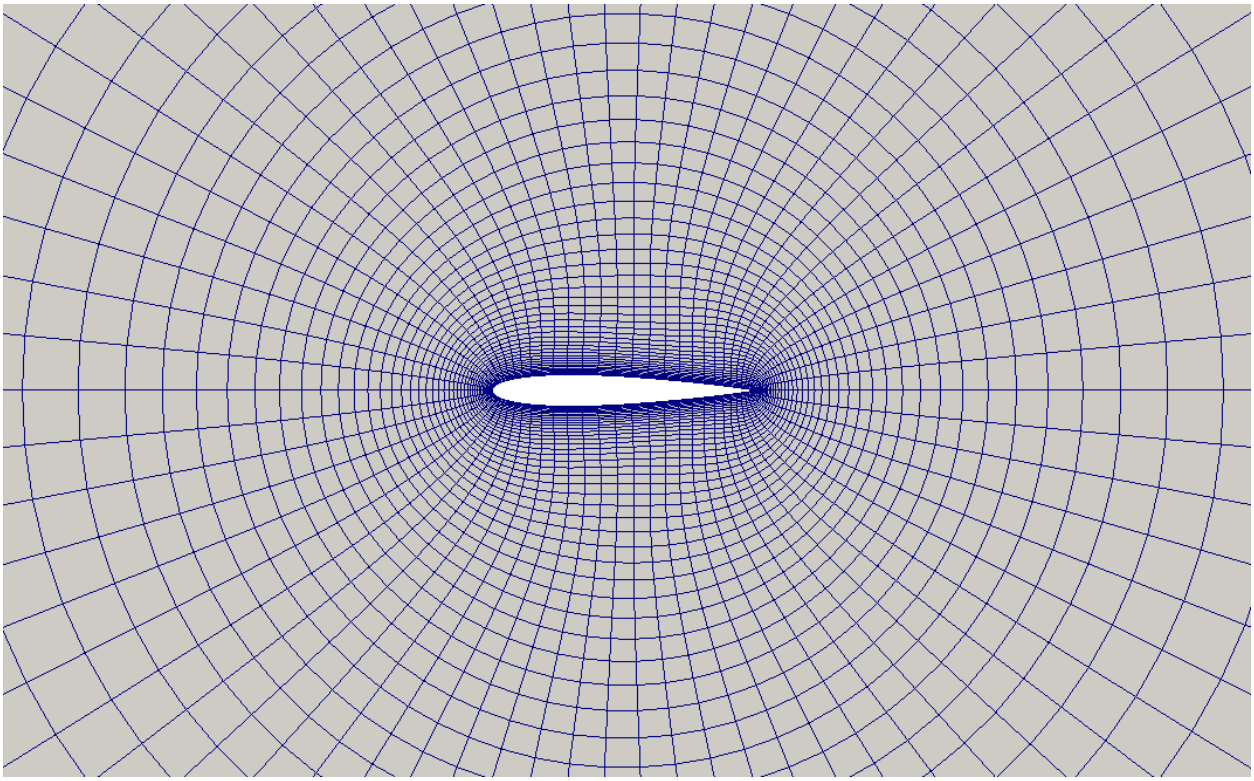


Figure 2.24: NACA 0012 airfoil 60x70 quadrilateral element mesh at 0° angle of attack.

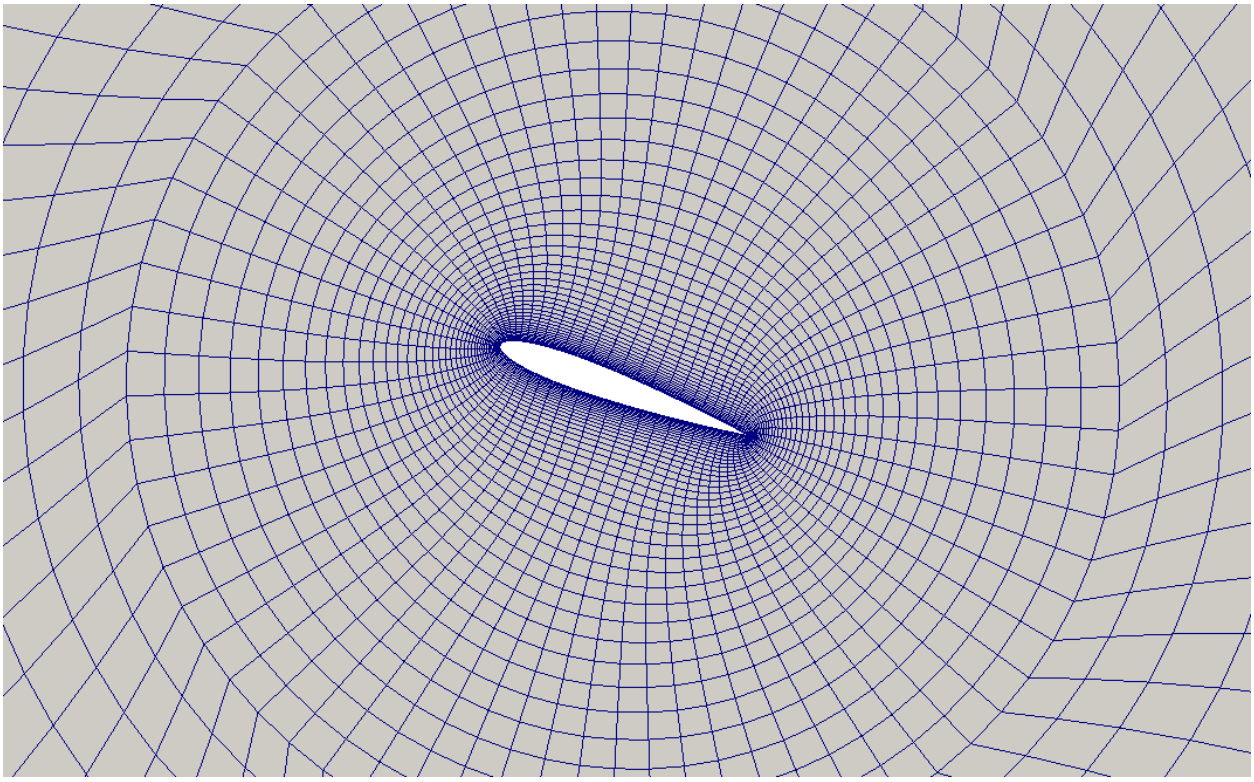


Figure 2.25: NACA 0012 airfoil 60x70 quadrilateral element mesh at 20° angle of attack.

via mesh motion.

Figures 2.30-2.31 show the pressure and skin friction coefficients for the airfoil at 0° angle of attack.

Figures 2.32-2.33 show the drag and lift coefficient history as the airfoil pitches between 10° and 30° angle of attack. The two forms of prescribing the pitching motion of the airfoil (mesh pitching and freestream pitching) show negligible differences in the drag coefficient and lift coefficient through time. As expected, the ALE corrected drag and lift coefficient computed on the moving mesh match the coefficients computed by the non-moving, freestream pitching problem.

### 2.6.5 Axisymmetric 50 Caliber Bullet

This problem demonstrates the ability of the finite element solver from computing viscous axisymmetric solutions about a slender body. This class of problems is important for re-entry type bodies as well munitions design. Similar studies for 50 caliber projectiles have been conducted by other researchers [129] for fully three-dimensional geometry including turbulence effects and body spin. It is important to note that the study contained in this section was run without the effects of turbulence and a spinning body.

#### 2.6.5.1 Problem Description

The problem consists of a $M_\infty = 2.4$, $Re = 3.85 \times 10^6$ flow around a 50 caliber bullet. Figure 2.1 shows the physical domain and boundary conditions for the problem. The domain boundaries span the area $-0.2 \leq x \leq 0.069012\ m$ and $0.0 \leq y \leq 0.8$, and the nose of the bullet is at the origin of the domain.

The inflow boundary conditions are consistent with sea-level standard atmospheric values and the solution is completely prescribed on this surface ($\rho = 1.225\ kg/m^3$, $V_\infty = 816.7\ m/s$, $T = 288.15\ K$). The symmetric boundary lying on the axis just forward of the nose imposes the conditions $u_y = \tau_{xy} = q_y = 0$. The wall boundary satisfies the no-slip condition of $u_x = u_y = 0$ with an adiabatic wall boundary condition. The outflow boundary has no prescribed variables,

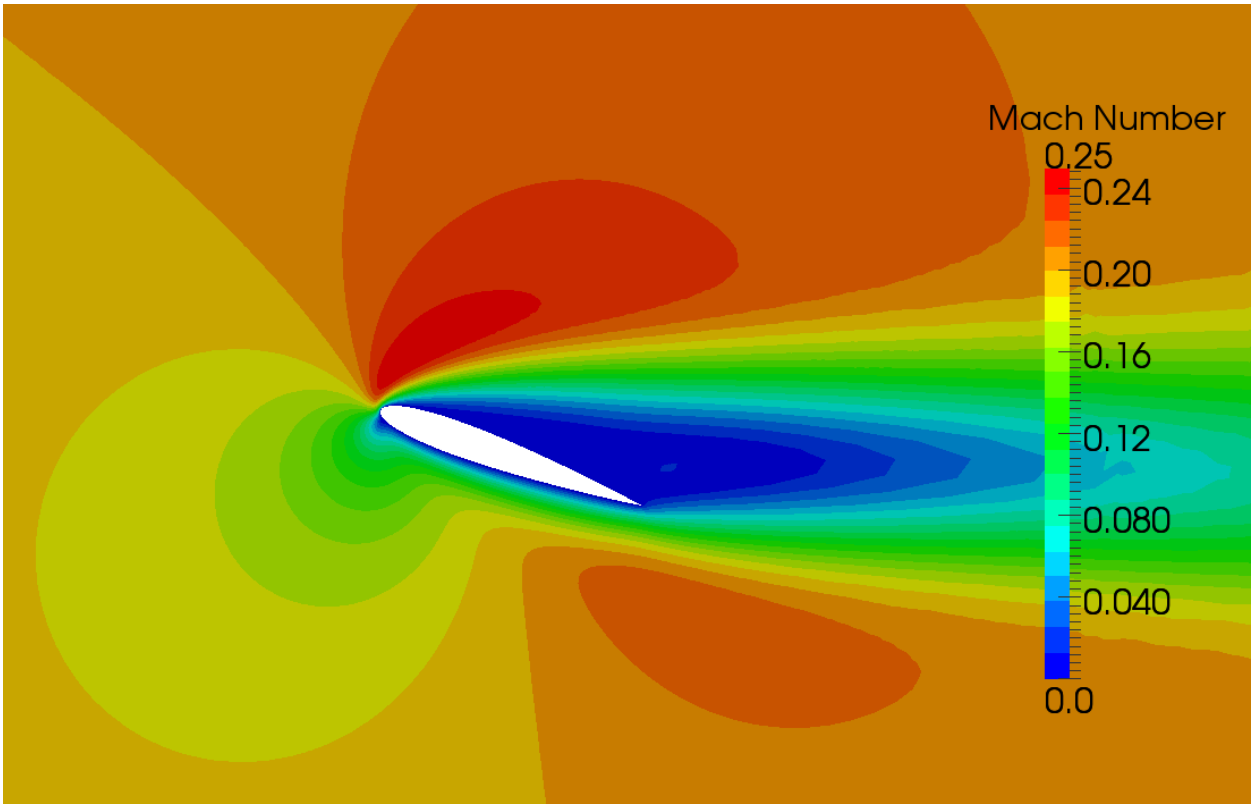



Figure 2.26: 0° to 20° pitching NACA 0012 Mach number contours at 20° angle of attack (120x140 element mesh).

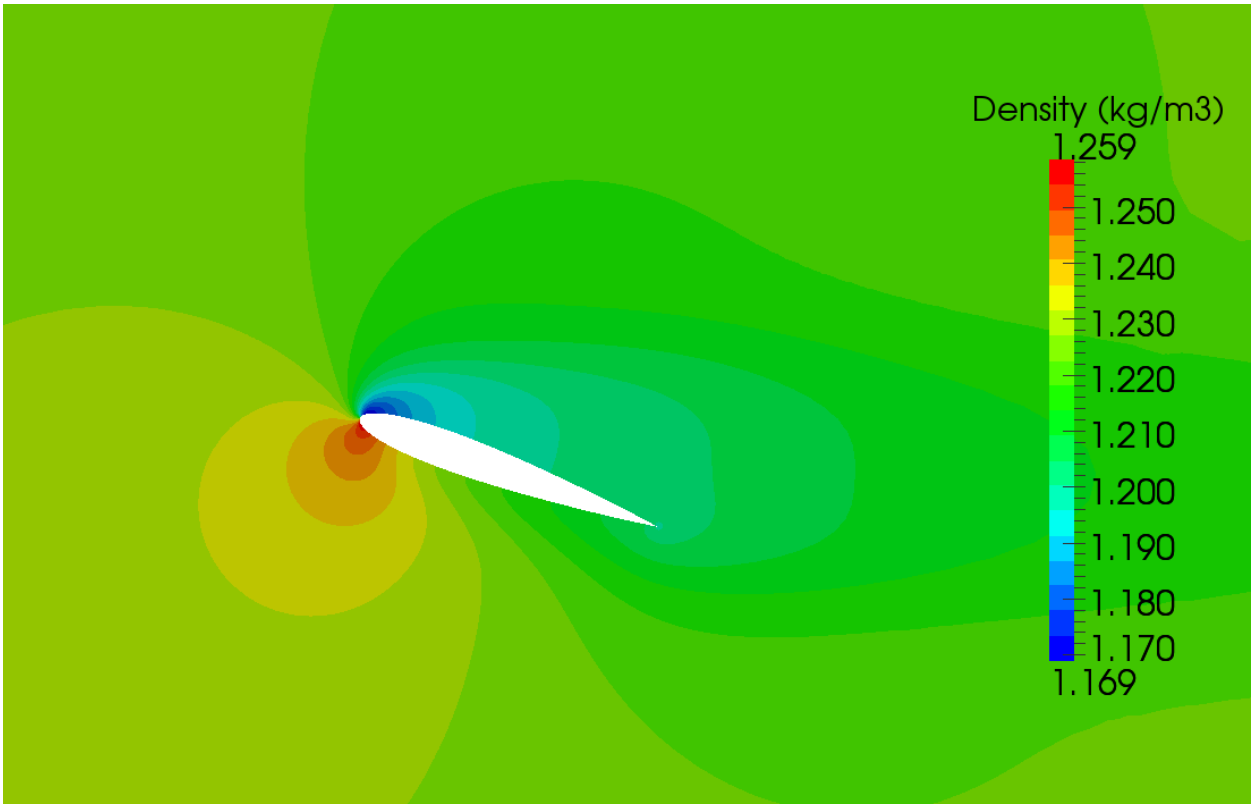



Figure 2.27: 0° to 20° pitching NACA 0012 density contours at 20° angle of attack (120x140 element mesh).

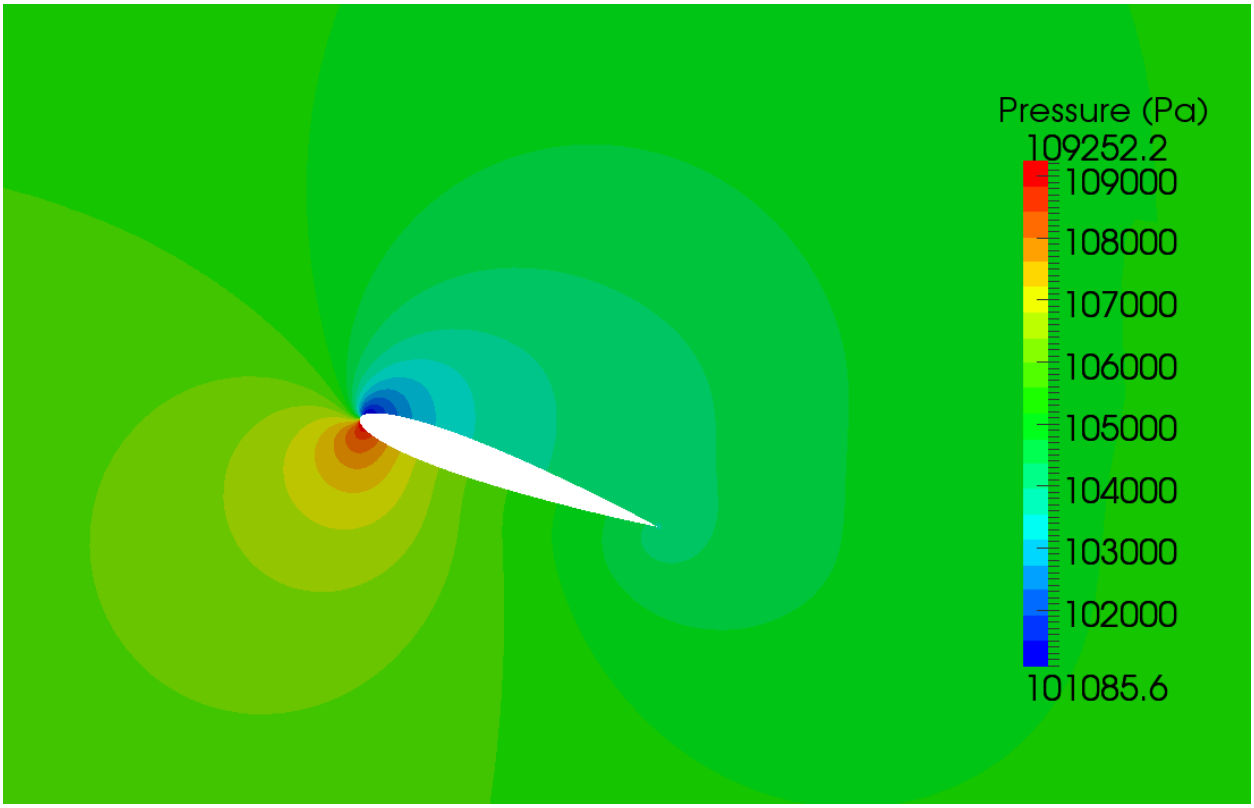



Figure 2.28: 0° to 20° pitching NACA 0012 pressure contours at 20° angle of attack (120x140 element mesh).

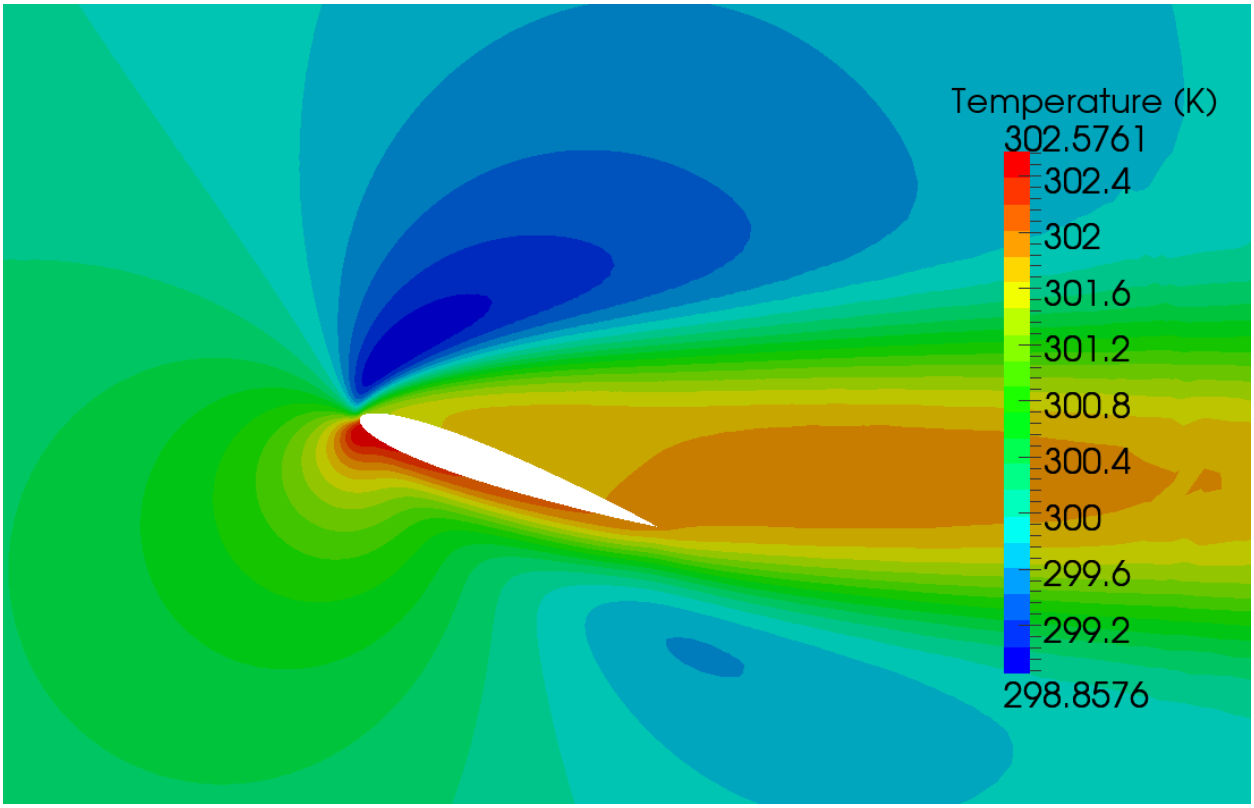



Figure 2.29: 0° to 20° pitching NACA 0012 temperature contours at 20° angle of attack (120x140 element mesh).

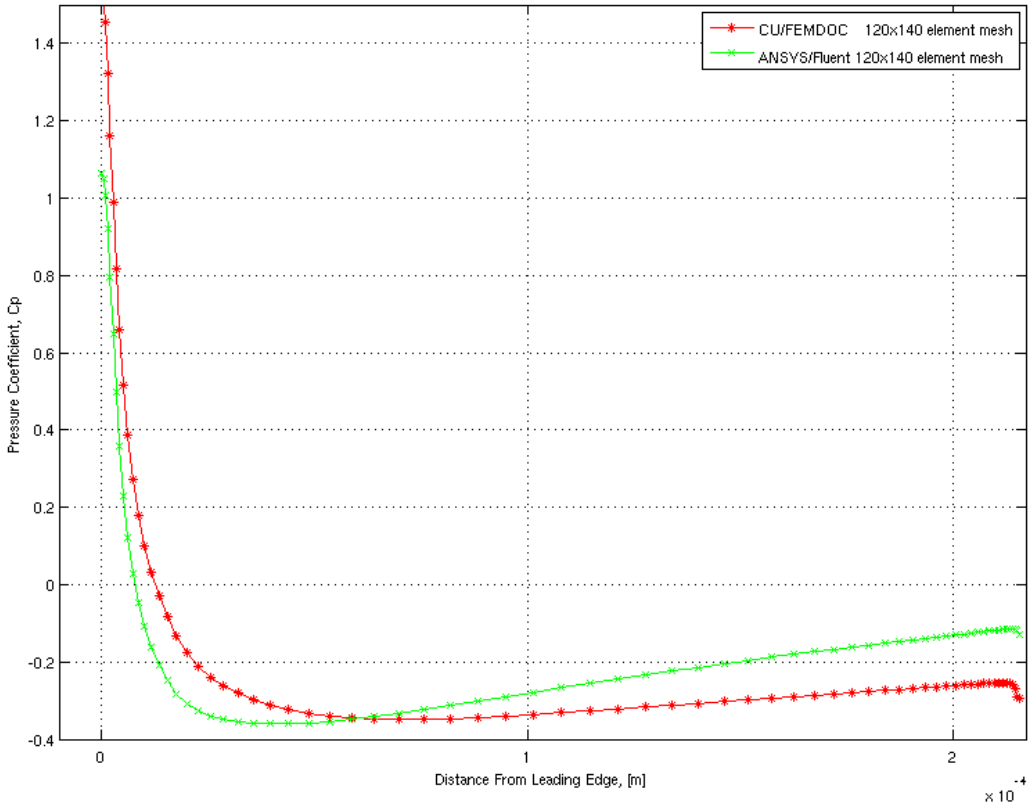



Figure 2.30: 0° AoA NACA 0012 wall pressure coeffient profiles (120x140 element mesh).

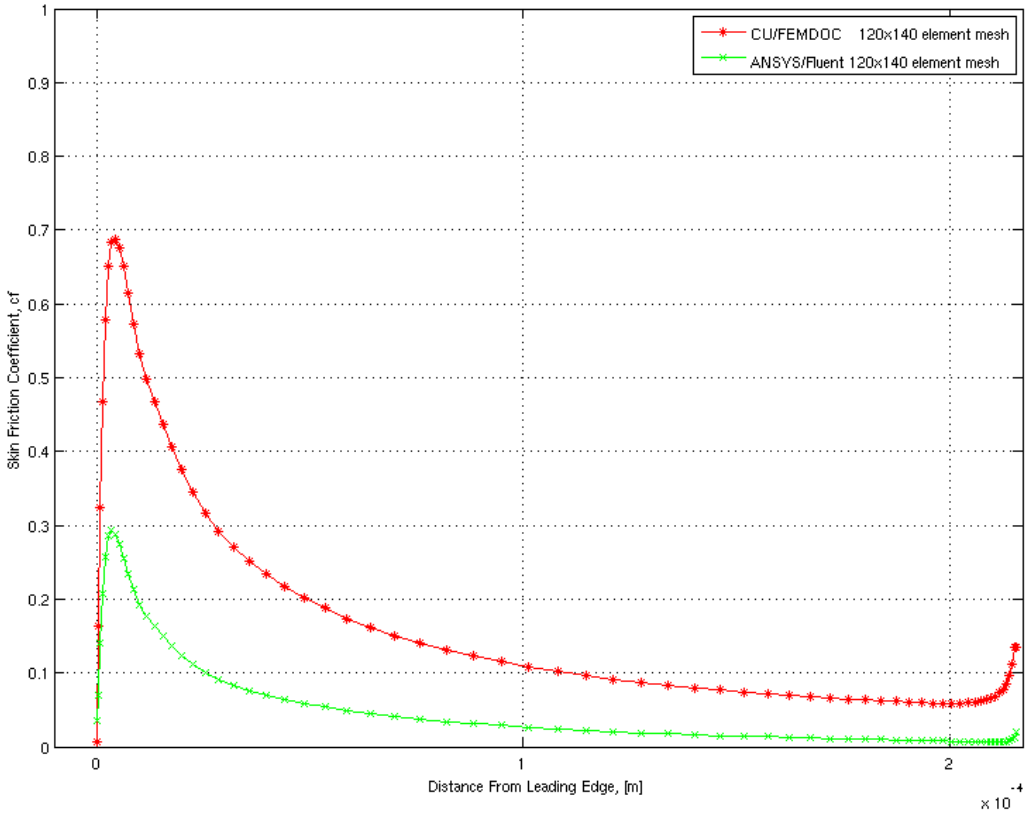



Figure 2.31: 0° AoA NACA 0012 skin friction coeffient profiles (120x140 element mesh).

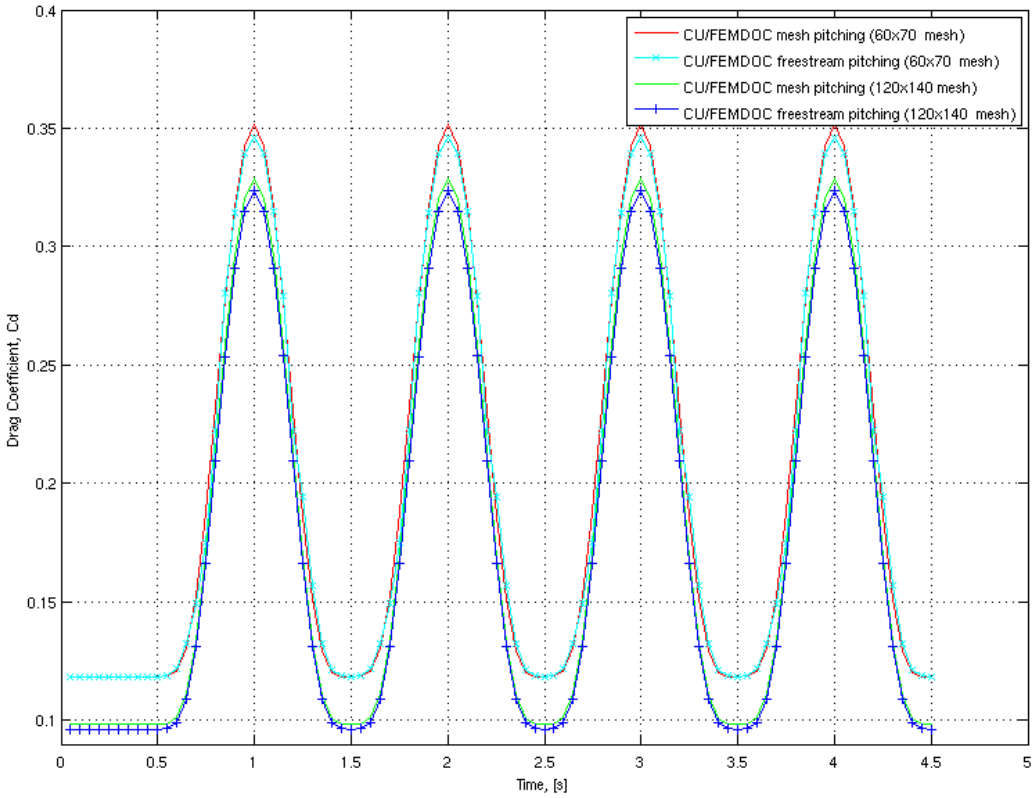



Figure 2.32: 0° to 20° pitching NACA 0012 drag coefficient time history.

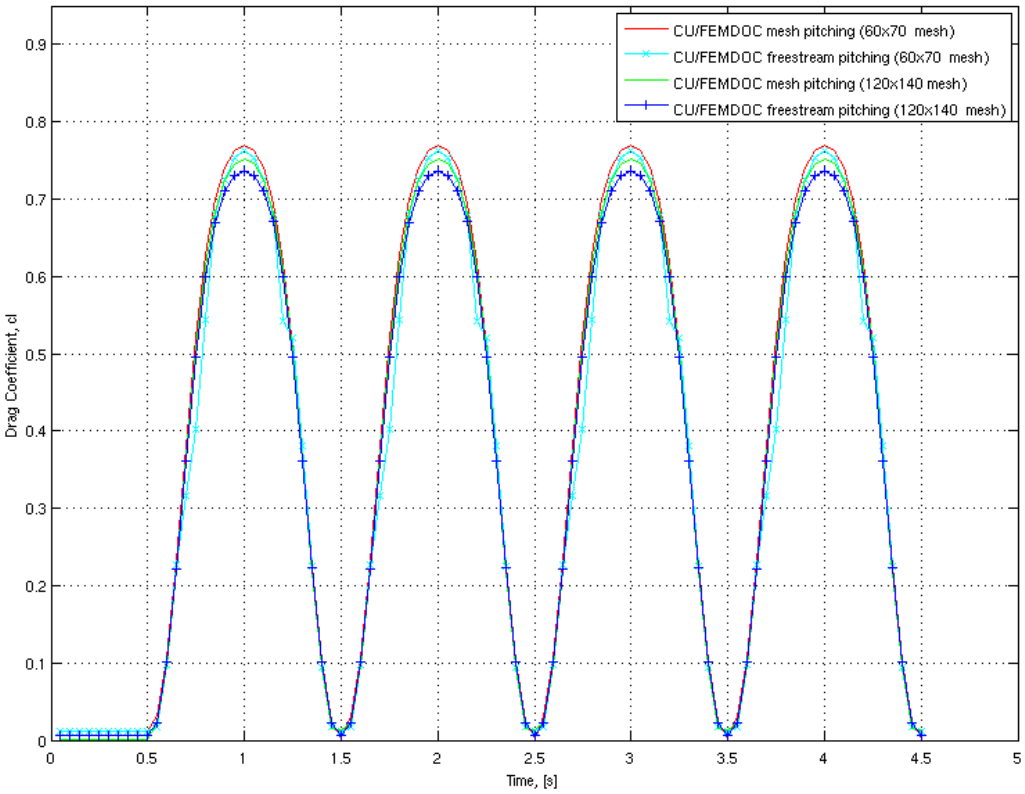



Figure 2.33: 0° to 20° pitching NACA 0012 lift coefficient time history.

however, the boundary term appearing in equation 2.38 is appropriately integrated.

The Sutherland law is used to model the viscosity of the fluid and uses the model coefficients $\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110\ K$. The thermal conductivity of the fluid is modeled by a constant Prandtl number as $\kappa = c_p \mu / Pr$, where $Pr = 0.71$.

#### 2.6.5.2 Computational Mesh and Model

A hierarchy of meshes are used for solving this problem; the meshed domains consist of 50x250, 100x500, and 200x1000 quadrilateral elements. Figure 2.35(a) shows the 50x250 element mesh and the 200x1000 element mesh's 64 process domain decomposition.

The problem is integrated in time to steady-state via a backward Euler (BDF-1) scheme beginning with in an initial time step of $1.0 \times 10^{-4}\ s$. The adaptive time stepping scheme discussed in section 2.4.2 is used with a time step increase limit of 10 %. A Newton scheme is used to solve the nonlinear equations. The nonlinear iteration loop exits when the nonlinear residual drops at least two orders of magnitude or when the initial nonlinear residual norm is less than $1.0 \times 10^{-3}$. The Trilinos/Aztec GMRES iterative solver is used for solving the linear problem and employs and global ILU(1) preconditioning and a linear residual drop criteria of $1.0 \times 10^{-4}$.

#### 2.6.5.3 Results and Discussion

Figure 2.36(a) shows the velocity magnitude and Figure 2.36(b) shows the density contours for the 50 caliber bullet simulation. Figures 2.37(a) and 2.37(b) show the pressure and temperature temperature contours. The stagnation pressure may be computed from the following equation:

$$P_0 = P_\infty \left( \frac{(\gamma - 1)^2 M_\infty^2}{4\gamma M_\infty^2 - 2(\gamma - 1)} \right)^{\frac{\gamma}{\gamma - 1}} \frac{1 - \gamma + 2\gamma M_\infty^2}{\gamma + 1} \tag{2.157}$$

and the stagnation temperature can be computed from the the equation

$$T_0 = T_\infty \left( 1 + \frac{\gamma - 1}{2} M_\infty^2 \right) \tag{2.158}$$

For this problem the theoretical value of the stagnation pressure is $7.8 \times 10^5\ Pa$ and the theoretical value of the stagnation temperature is 620 $K$. The stagnation pressure and temperature computed

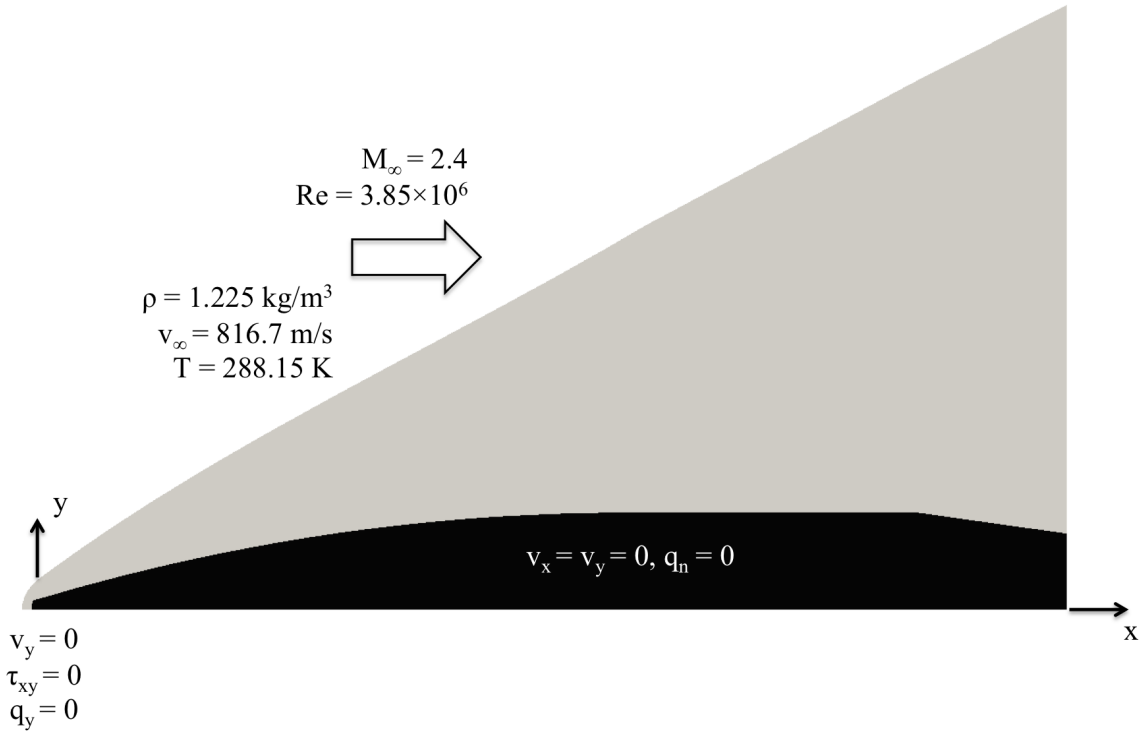
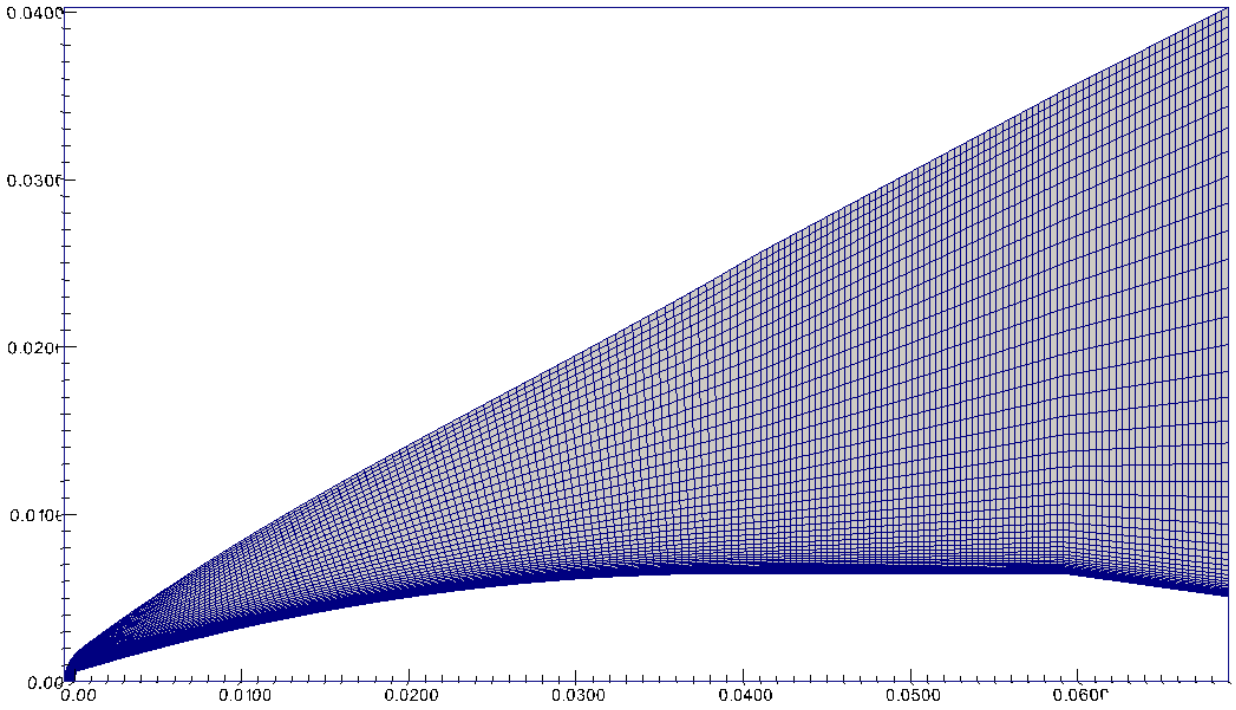
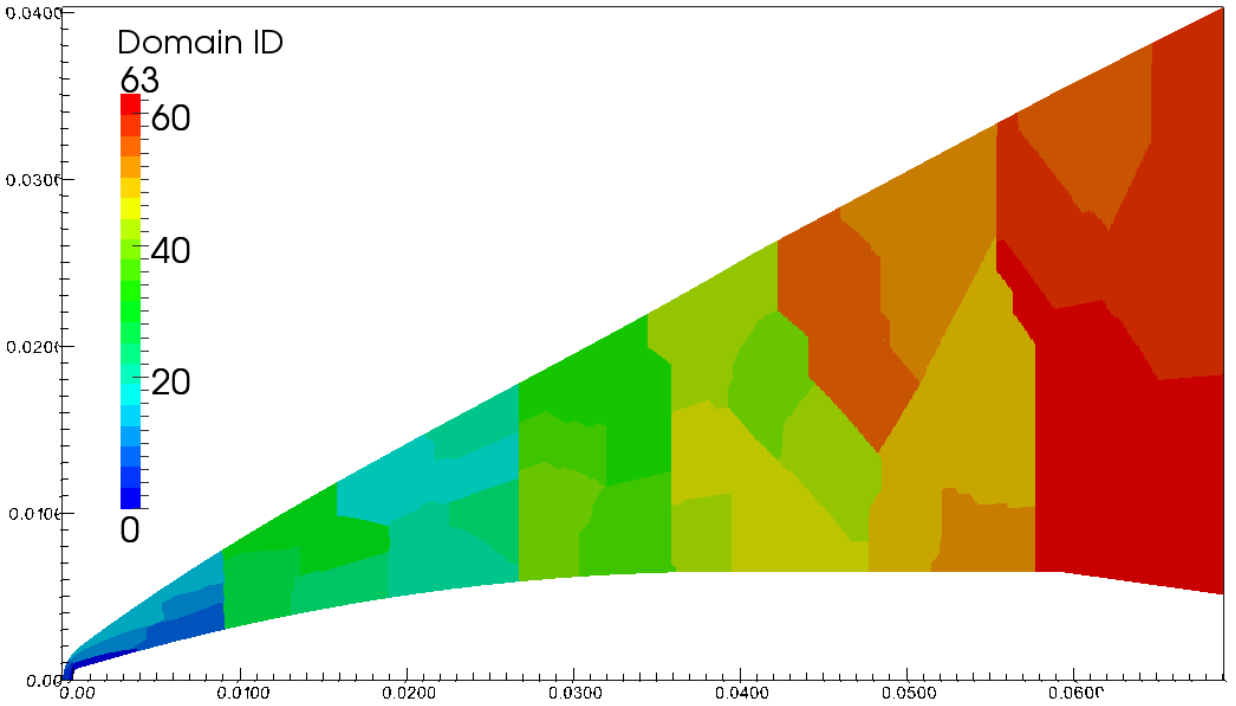



Figure 2.34: 50 caliber bullet geometry and boundary conditions.

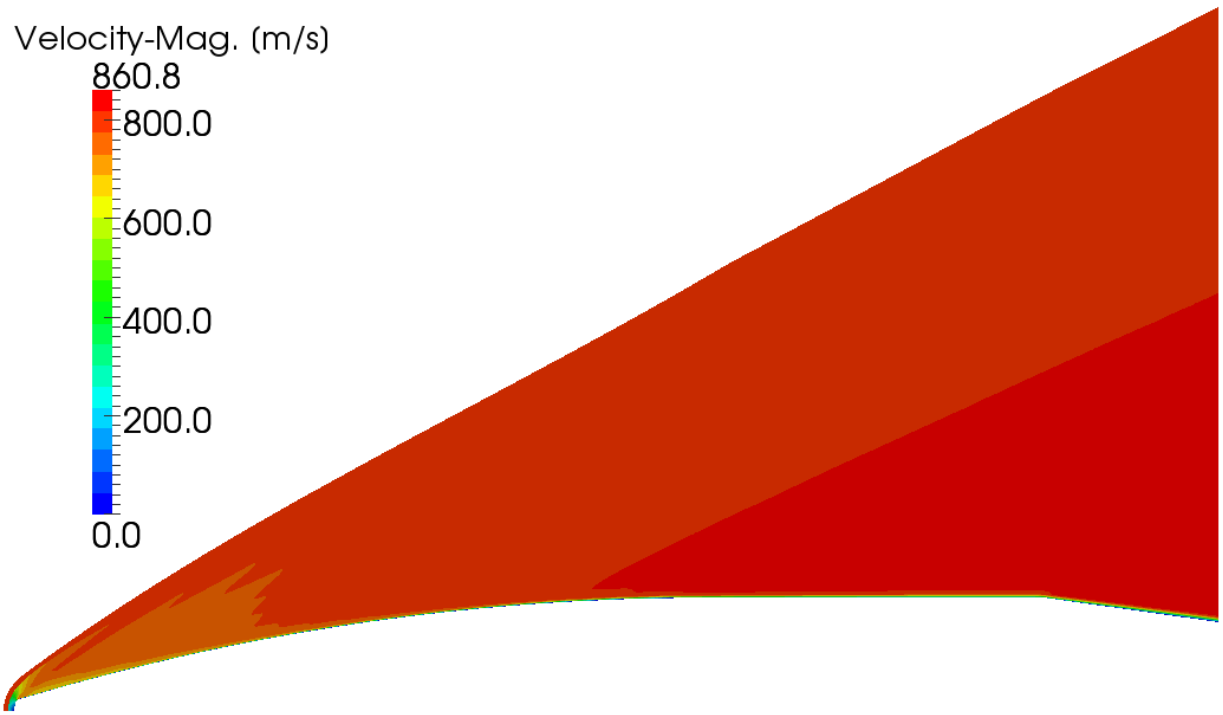



(a) 50x250 element mesh.

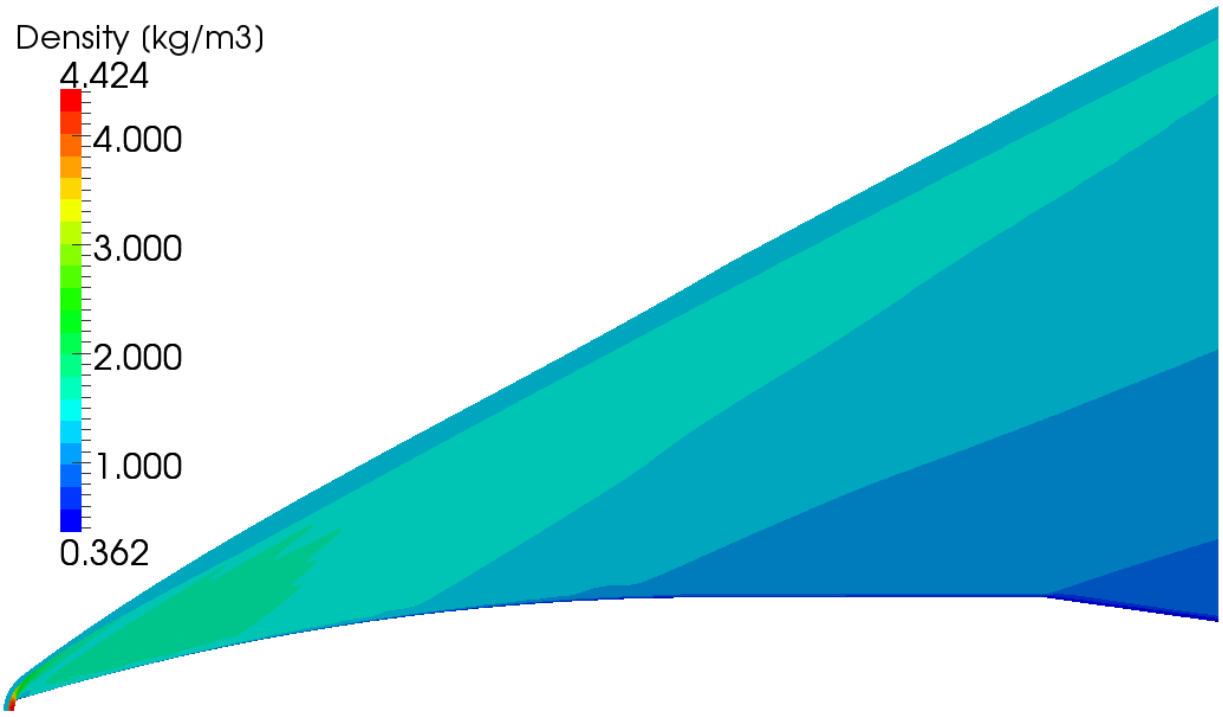



(b) 200x1000 mesh with 64 process domain decomposition.

Figure 2.35: Axisymmetric 50 caliber bullet computational mesh.

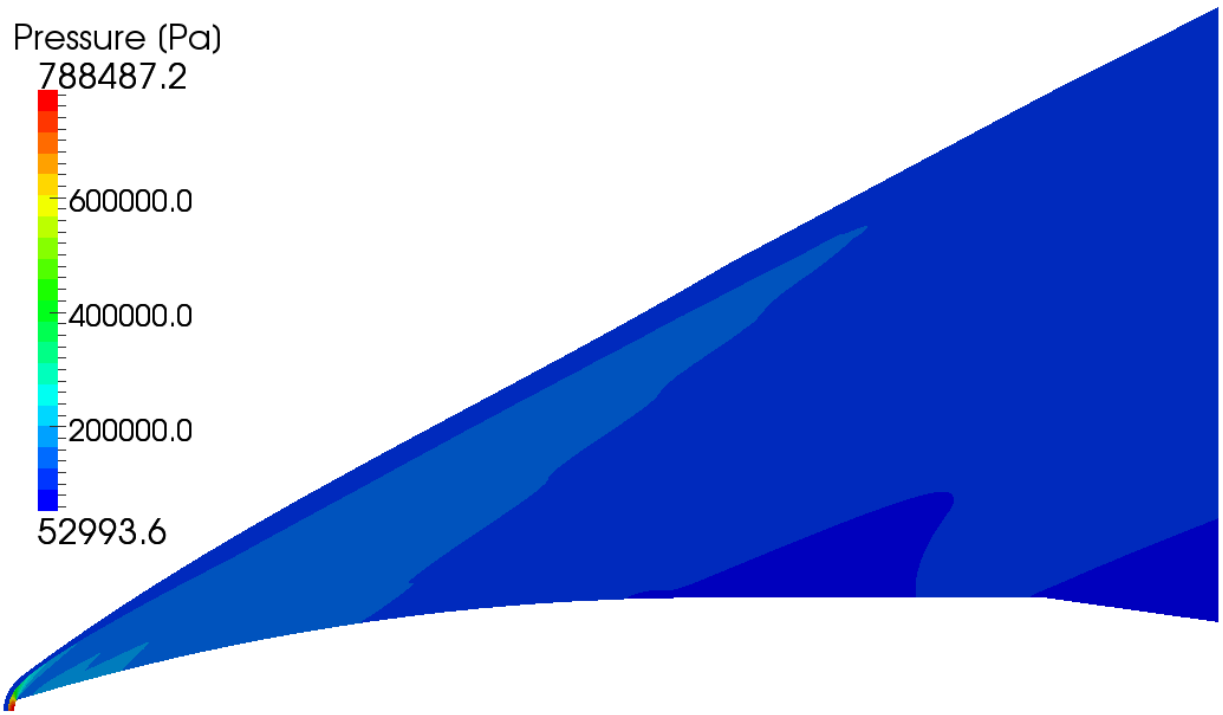



(a) Velocity magnitude contours.

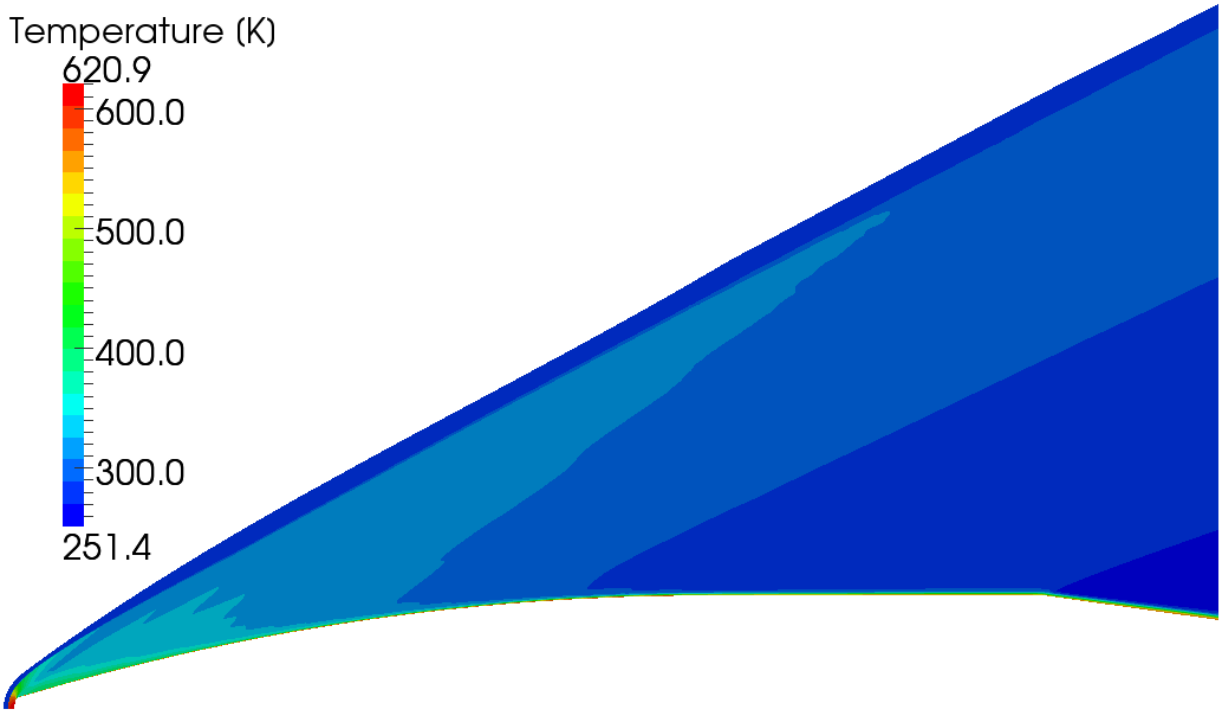



(b) Density contours.

Figure 2.36: Axisymmetric 50 caliber bullet solution contours.




(a) Pressure magnitude contours.




(b) Temperature contours.

Figure 2.37: Axisymmetric 50 caliber bullet solution contours.

by the code and shown figures 2.37(a) and 2.37(b) agree very well with their analytical values.

Figures 2.38 - 2.40 show the pressure, shear stress, and temperature profiles at the wall for the three different meshes. It is believed the oscillations observed in the pressure, shear stress and temperature profiles are caused by compression waves that form and emanate from the bullet's surface. These compression waves form as the cross-sectional area of the body increases and ultimately coalesce to form a shock wave away from the body. If the mesh is too coarse in the presence of the compression waves the oscillations observed here will occur. As shown by these figures, the severity of the oscillations is reduced with mesh refinement.

### 2.6.6 3-Dimensional Nose Tip

This problem is a continuation of the two-dimensional example 2.6.3 as it involves the supersonic flow about a the same nose tip geometry but now in three dimensions. The problem in run with both a pitch and yaw angle; as such the flow is truly three-dimensional and an axisymmetric calculation not appropriate.

#### 2.6.6.1 Problem Description

The flow conditions for this problem are identical to of the example in section 2.6.3, however the pitch and yaw angles are both specified to at 10°. Figure 2.41 shows the problem geometry and boundary conditions. Freestream flow values for $\rho$, $\rho v_x$, $\rho v_y$, $\rho v_z$, and $\rho E$ are prescribed on the inlet boundary and a no-slip adiabatic wall boundary condition is used for the surface of the body. No values are prescribed for the outflow edge and the viscous fluxes are integrated to be consistent with the integrated-by-parts weak form the Galerkin term.

#### 2.6.6.2 Computational Mesh and Model

As with the two-dimensional problem, three meshes of increasing refinement are used for this study. The three-dimensional meshes were generated by revolving the two-dimensional meshes about their x-axes. This produces good quality hexahedral elements throughout the volume, how-

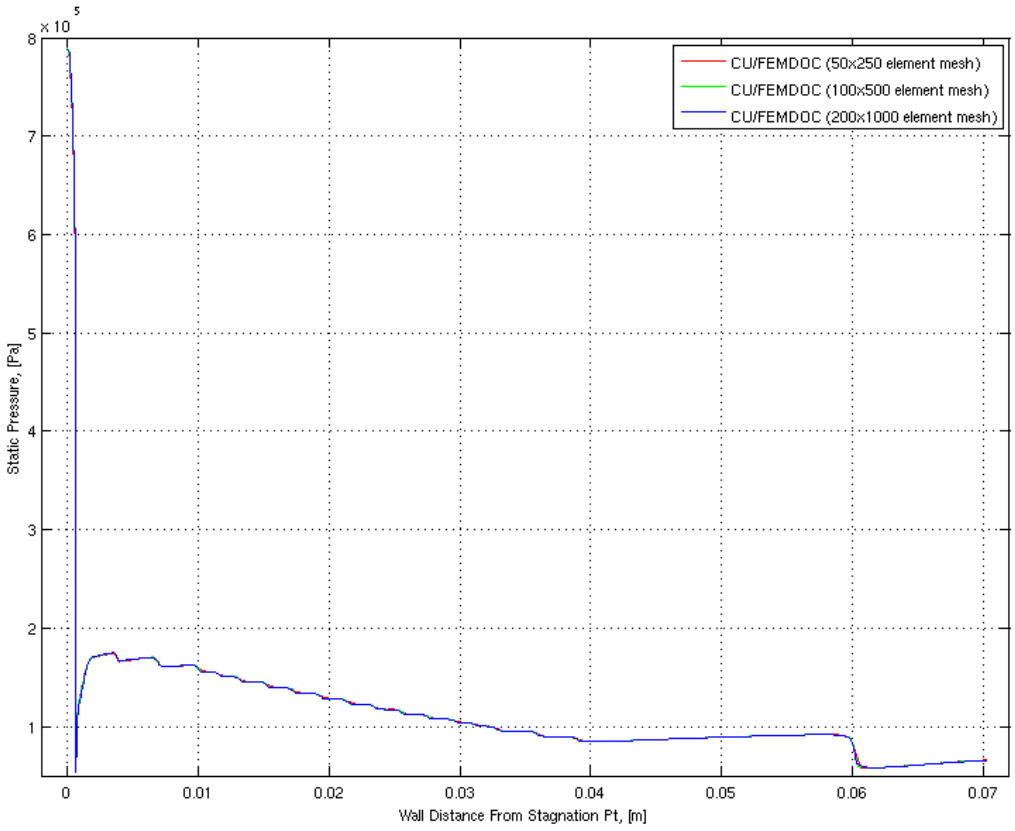



Figure 2.38: 50 caliber bullet wall pressure profiles at steady-state.

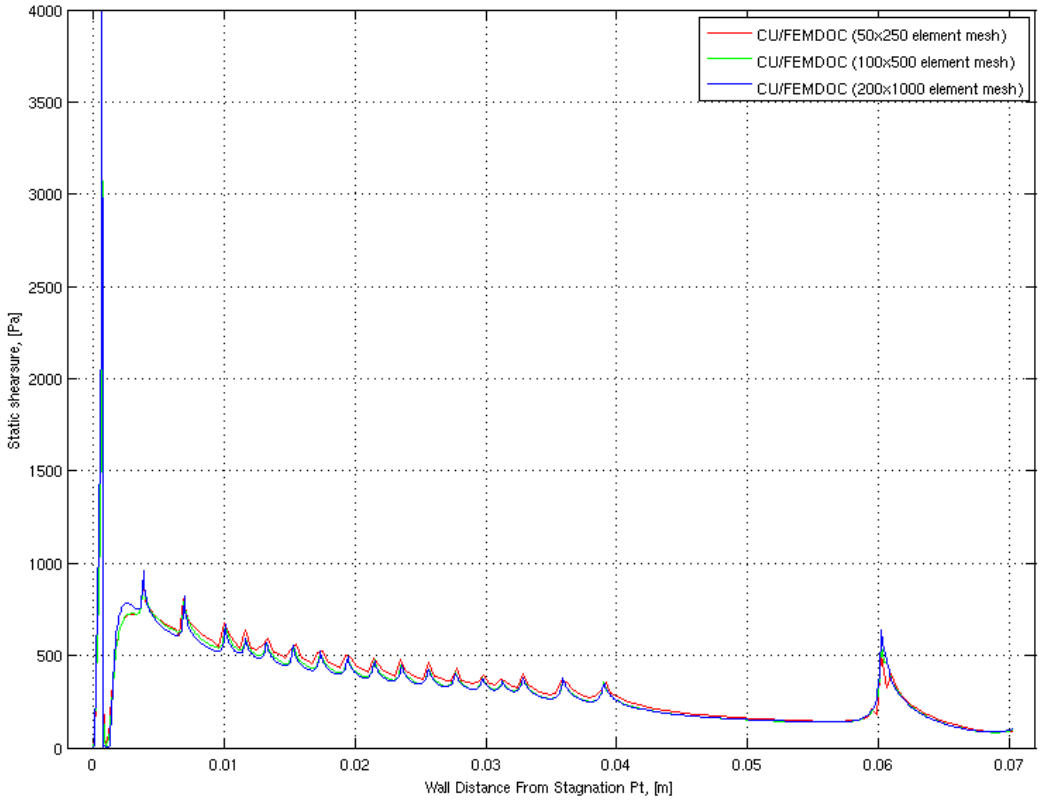



Figure 2.39: 50 caliber bullet wall shear stress profiles at steady-state.

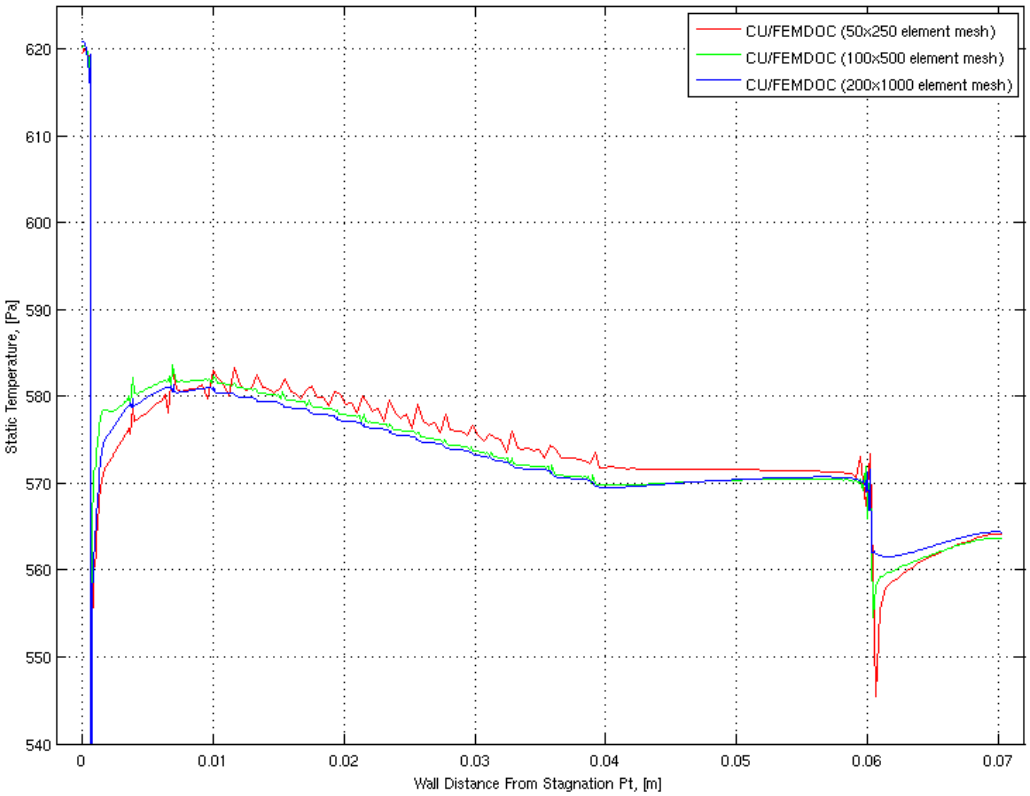



Figure 2.40: 50 caliber bullet wall temperature profiles at steady-state.

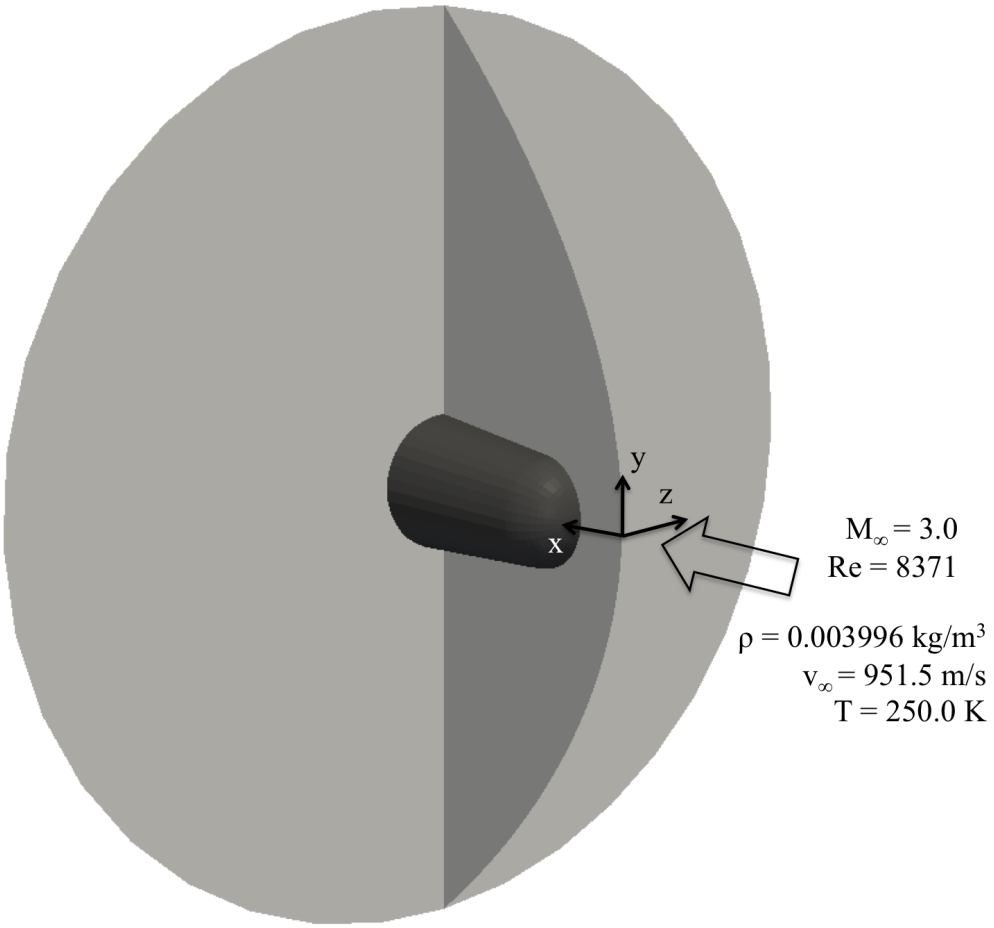



Figure 2.41: Three dimensional supersonic nose tip geometry and boundary conditions.

ever, one drawback is that pentahedral elements that exist on the stagnation line. For each node on the stagnation line, there are at least as many elements connected to it as there are revolution intervals. This leads to large stencil in terms of degree-of-freedom to degree-of-freedom dependencies which has the potential for causing convergence problems and performance loss for linear solvers. However, no such difficulties were experienced in solving these problems.

The backward Euler time integrator is used with adaptive time stepping where an initial time step size of $1 \times 10^{-7}$ $s$ and a maximum time step increase of 1.2 are used. A nonlinear relaxation factor of 0.9 is used with nonlinear convergence criteria of $\epsilon = 0.1$. Node based stabilization and $\delta$ shock capturing parameters are used. As in several the prior examples, the Trilinos/Aztec GMRES is used with a global ILU(0) preconditioner.

The coarsest mesh, shown in Figure 2.42, is a structured 41x40 quadrilateral element domain revolved 360° at 10° revolution intervals and has a total of 9757 nodes. This mesh is decomposed into 8 domains and solved using 8 processors. The medium level mesh is a 74x80x36 element revolved mesh and has a total of 216,075 nodes. This mesh is decomposed into 32 domains and solved using 32 processors. The finest mesh is a 148x160x36 element revolved mesh and has a total of 858,389 nodes. This mesh is decomposed into 128 domains and solved on 128 processors. Figure 2.43 shows this mesh's domain decomposition.

#### 2.6.6.3 Results and Discussion

Figures 2.44 through 2.47 show the Mach, density, pressure and temperature contours for the finest mesh used for this problem. Although not readily apparent from these figures, the non-zero pitch and yaw angles produce a fully three dimensional flow and explain the asymmetries seen on wall and boundaries of the flow domain.s

Figure 2.48 shows the viscous shear stress tensor components plotted on the wall of the body.

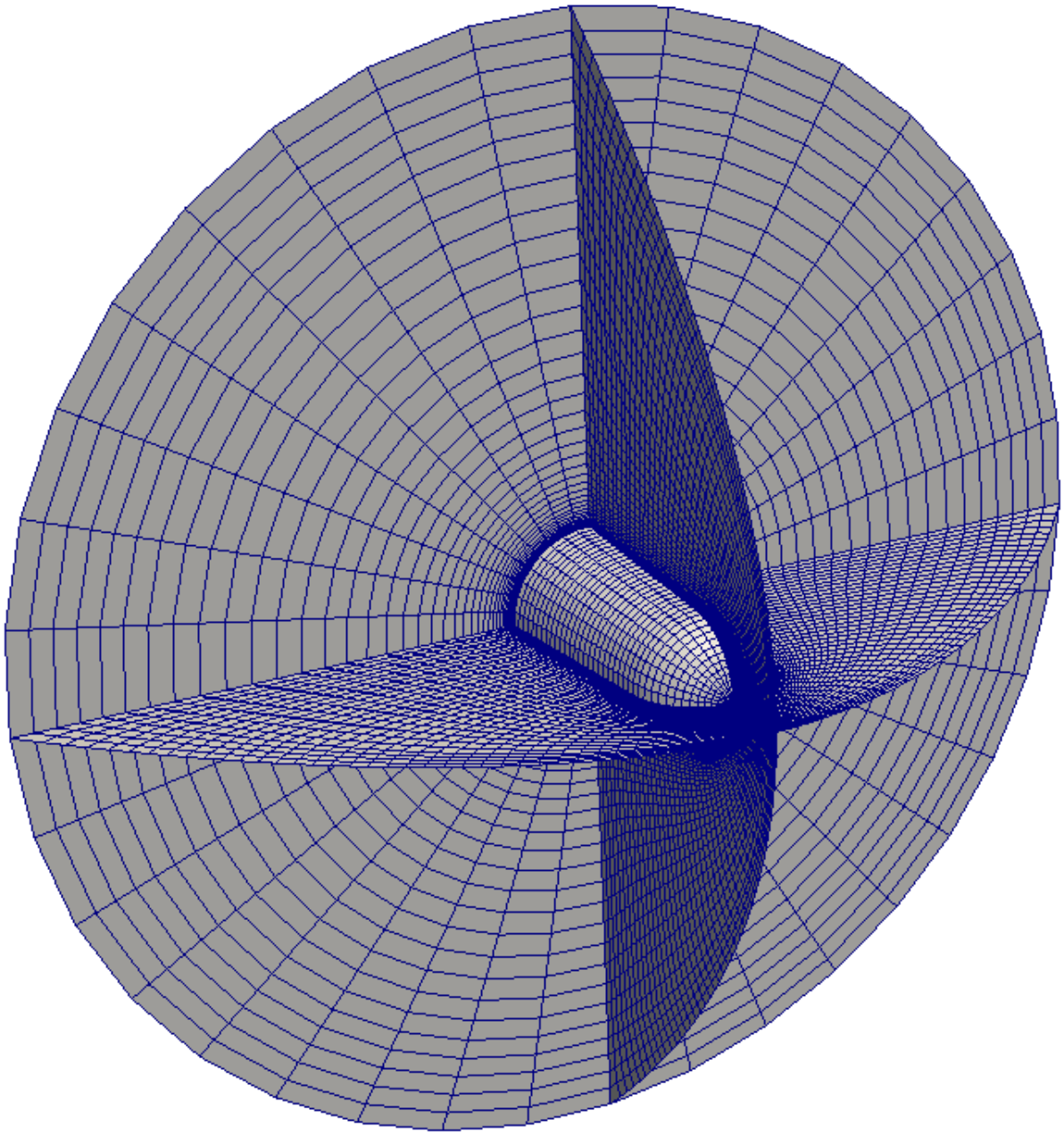


Figure 2.42: Three-dimensional nose tip 41x40x36 element revolved mesh.

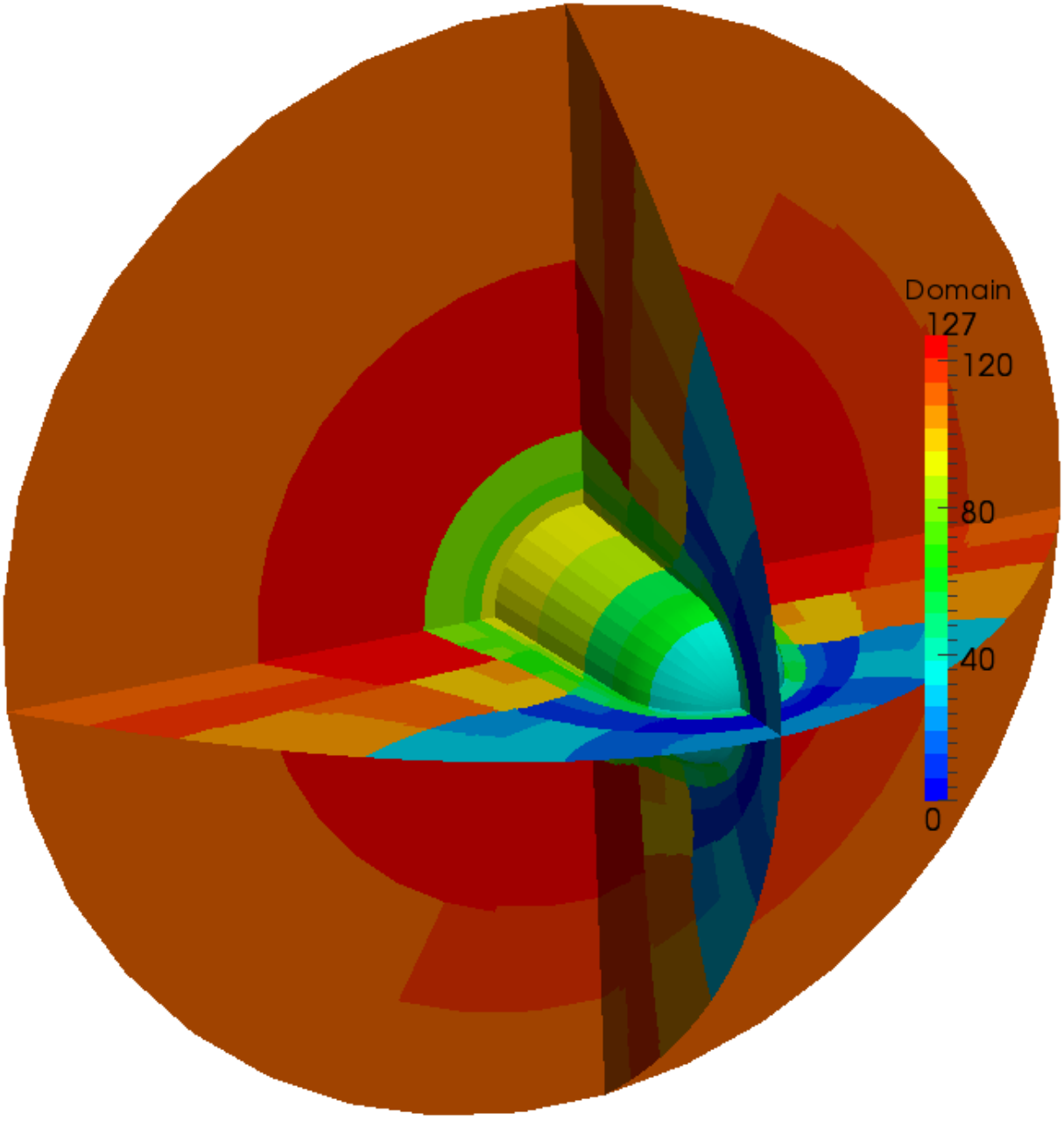



Figure 2.43: Three-dimensional nose tip 148x160x36 element revolved mesh domain decomposition.

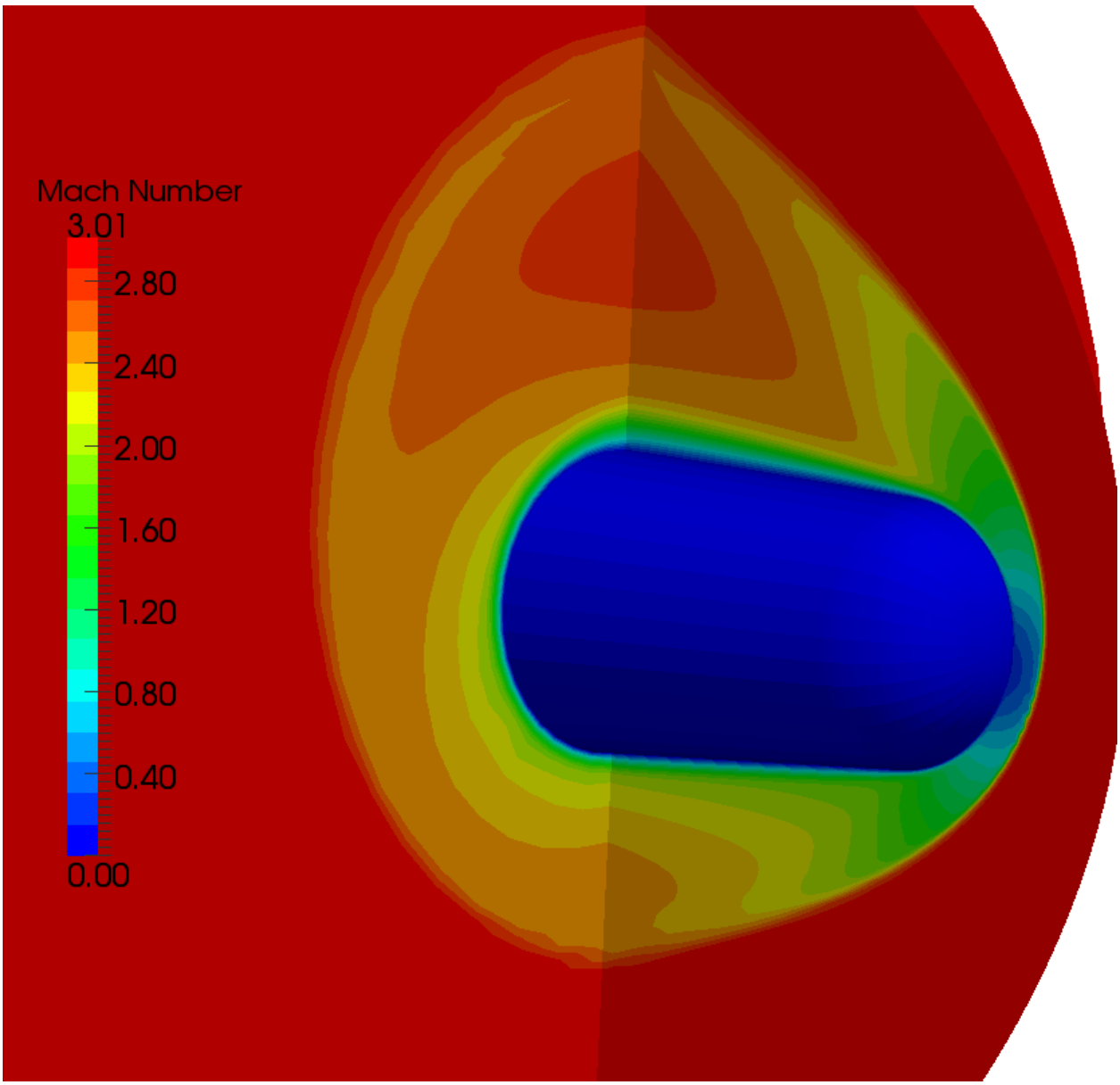



Figure 2.44: Three-dimensional nose tip problem Mach number contours (148x160x36 element mesh).

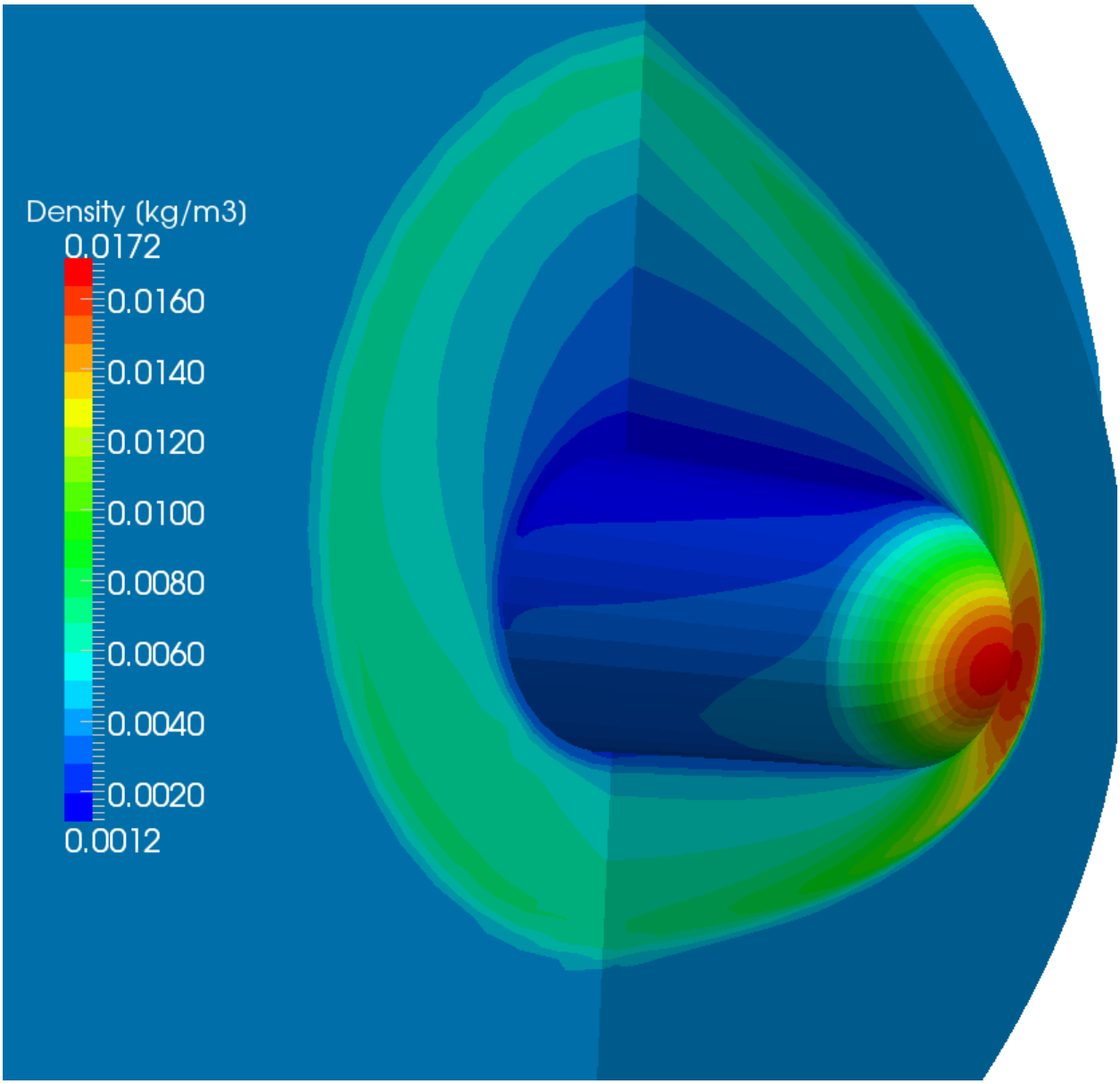



Figure 2.45: Three-dimensional nose tip problem density contours (148x160x36 element mesh).

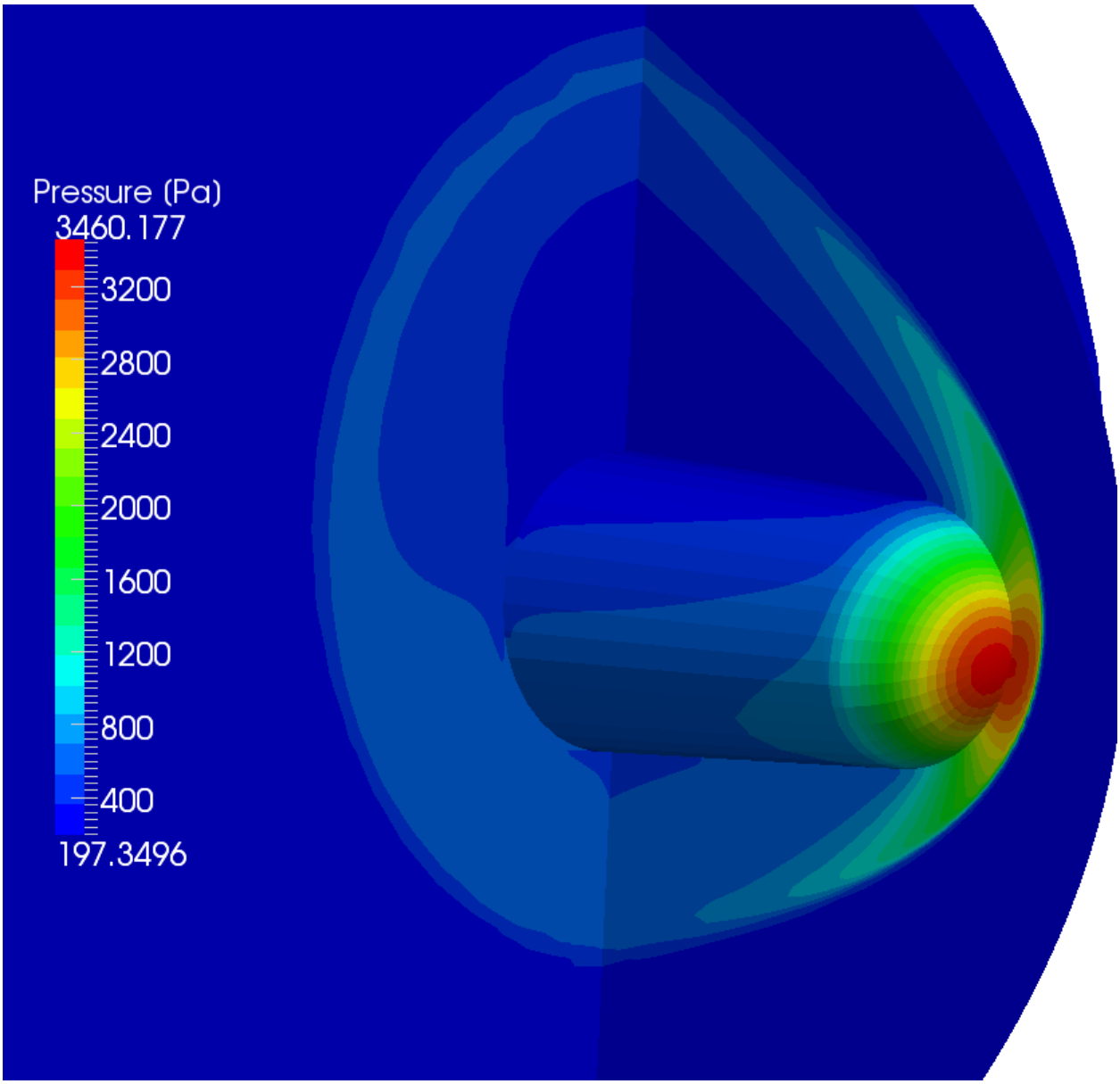



Figure 2.46: Three-dimensional nose tip problem pressure contours (148x160x36 element mesh).

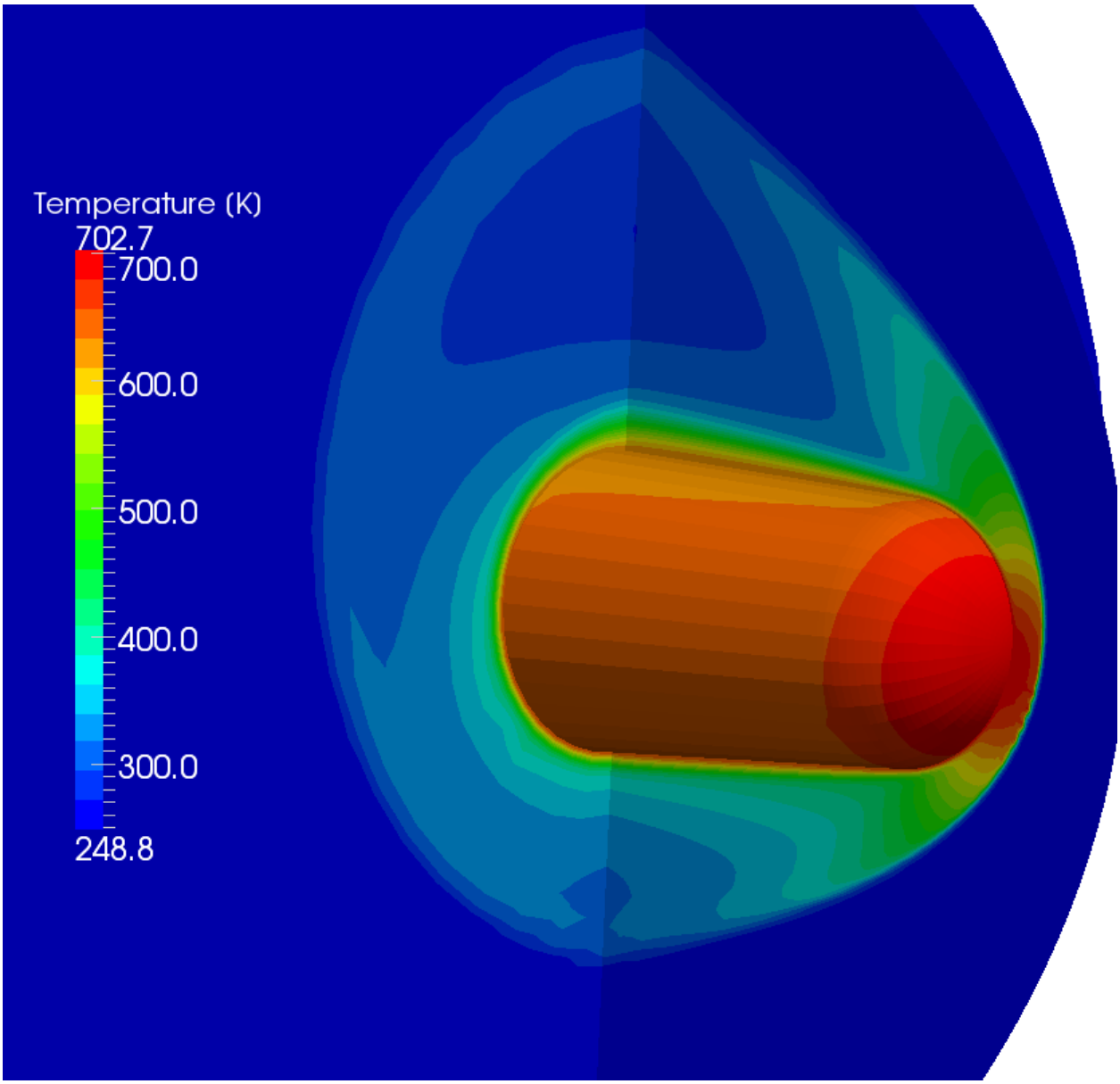
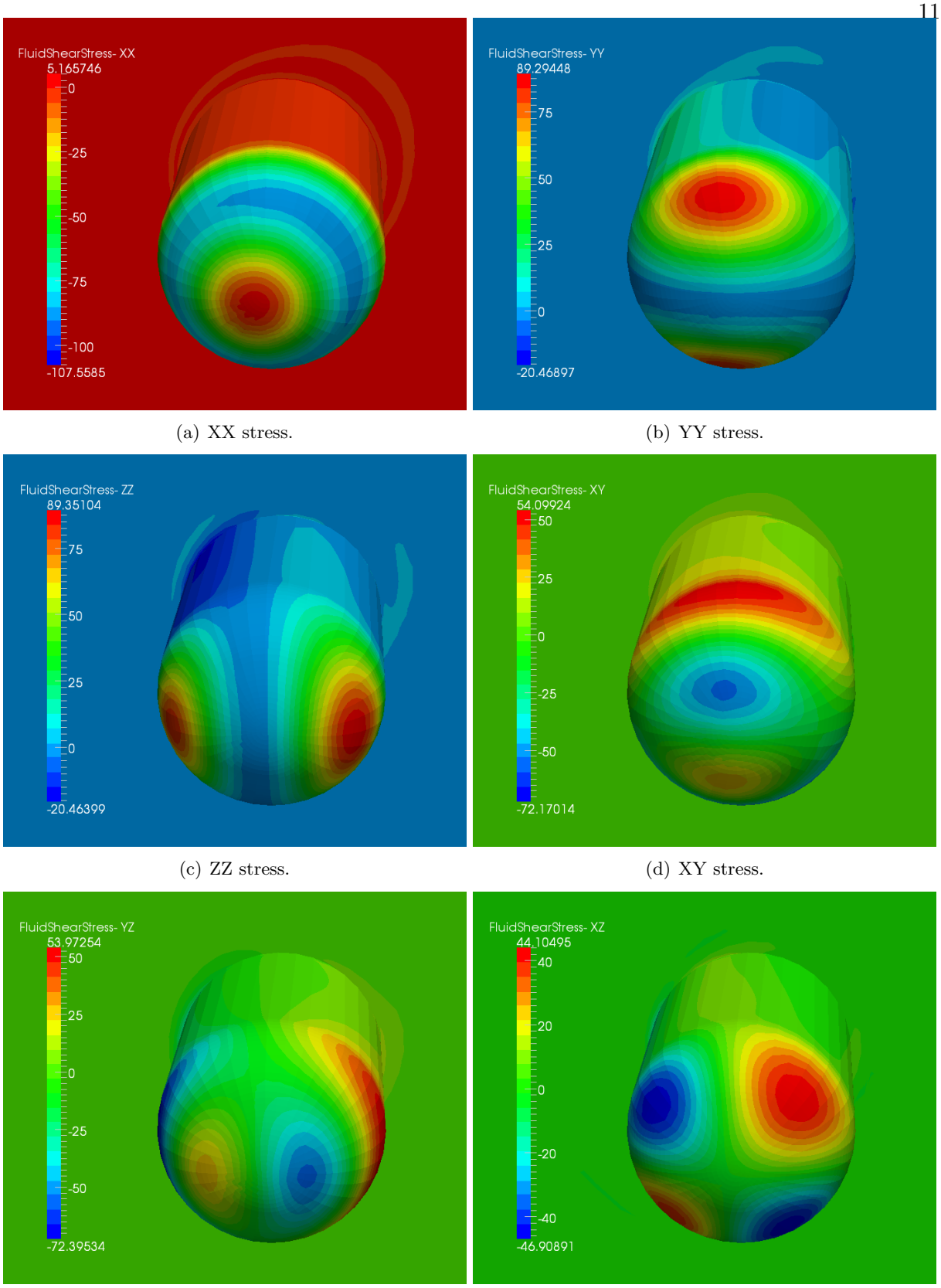



Figure 2.47: Three-dimensional nose tip problem temperature contours (148x160x36 element mesh).




(a) XX stress.

(b) YY stress.

(c) ZZ stress.

(d) XY stress.

(e) YZ stress.

(f) XZ stress.

Figure 2.48: Three-dimensional nose tip deviatoric stress contours (148x160x36 element mesh).

| **Mesh/# of CPUs** | CU/FEMDOC | SNL/Aria | ANSYS/Fluent |
|---|---|---|---|
| 28x16 / 1 CPU | 0.379 | 3.57 | 0.0859 |
| 56x32 / 1 CPU | 1.56 | 11.1 | 0.345 |
| 112x64 / 1 CPU | 5.76 | 31.9 | 1.03 |
| 224x128 / 1 CPU | 26.3 | 100 | 5.44 |
| 224x128 / 2 CPU | 13.3 | 54.0 | 3.10 |
| 224x128 / 4 CPU | 7.61 | 26.5 | 2.48 |
| 224x128 / 8 CPU | 3.98 | 14.0 | – |
| 224x128 / 16 CPU | 2.21 | – | – |
| 448x256 / 4 CPU | 42.4 | 97.4 | 28.0 |
| 448x256 / 8 CPU | 20.4 | 59.2 | – |
| 448x256 / 16 CPU | 11.8 | – | – |
| 448x256 / 32 CPU | 6.02 | – | – |

Table 2.1: Non-dimensional run times for serial & parallel simulations of Carter's flat plate.

| **Code (mesh)** | Stag. Pres. | % diff | Stag. Temp. | % diff. |
|---|---|---|---|---|
| Analytic Value | 3463 Pa | 0.00 % | 701.0 K | 0.00 % |
| CU/FEMDOC (41x40 mesh) | 3522 | 1.70 % | 697.6 | -0.49 % |
| CU/FEMDOC (74x80 mesh) | 3458 | -0.14 % | 701.9 | 0.13 % |
| CU/FEMDOC (148x160 mesh) | 3430 | -0.95 % | 703.4 | 0.34 % |
| SNL/Aria (41x40 mesh) | 3390 | -2.11 % | 697.6 | -0.48 % |
| SNL/Aria (74x80 mesh) | 3418 | -1.30 % | 697.8 | -0.46 % |
| SNL/Aria (148x160 mesh) | 3418 | -1.30 % | 697.8 | -0.46 % |
| ANSYS/Fluent (41x40 mesh) | 3534 | 2.05 % | 699.3 | -0.24 % |
| ANSYS/Fluent (74x80 mesh) | 3405 | -1.67 % | 700.0 | -0.14 % |
| ANSYS/Fluent (148x160 mesh) | 3454 | -0.26 % | 700.0 | -0.14 % |
| SNL/Premo (41x40 mesh) | 3624 | 4.65 % | 712.0 | 1.57 % |
| SNL/Premo (74x80 mesh) | 3494 | 0.90 % | 702.3 | 0.19 % |
| SNL/Premo (148x160 mesh) | 3482 | 0.55 % | 701.8 | 0.11 % |
| SNL/Saccara (41x40 mesh) | 3449 | -0.40 % | 704.0 | 0.43 % |
| SNL/Saccara (74x80 mesh) | 3473 | 0.29 % | 701.7 | 0.10 % |
| SNL/Saccara (148x160 mesh) | 3473 | 0.29 % | 701.3 | 0.04 % |
| NASA/Overflow (41x40 mesh) | 3408 | -1.59 % | 701.1 | 0.01 % |
| NASA/Overflow (74x80 mesh) | 3436 | -0.78 % | 701.3 | 0.04 % |
| NASA/Overflow (148x160 mesh) | 3452 | -0.32 % | 701.4 | 0.06 % |

Table 2.2: Nose tip problem analytic vs. numeric stagnation pressure and temperature comparison for all codes and all meshes.

# Chapter 3

# Computational Analysis of Transient Heat Transfer

## 3.1 Introduction

This chapter discusses the governing equation, discretization, and solution of transient heat transfer problems. The standard Galerkin treatment as well as a stabilized Galerkin discretization will be presented and used in this chapter. An arbitrary Lagrangian-Eulerian form of the parabolic partial differential equation will also be developed for the purpose of solving the equations on a moving mesh. Additionally, formulations for solving volumetric phase change and surface ablation problems will be shown. This chapter will conclude by presenting several numerical example problems that demonstrate the capabilities developed for this thesis to analyze transient heat transfer problems.

## 3.2 Transient Heat Conduction Equation

The transient heat equation for a solid represents an energy conservation equation that governs the time-dependent diffusion of heat energy through a body [119]. The heat equation is a parabolic partial differential equation and in general is much more amenable to solution via numerical methods than the hyperbolic partial differential equations considered in the previous chapter. Nevertheless, instabilities in the numerical solution may arise and these will be addressed in the next section.

### 3.2.1 Differential Form of the Transient Heat Equation

The transient heat conduction equation governs the time-dependent temperature distribution within a body. This equation is written in a generic form as

$$\frac{\partial H}{\partial t} + \frac{\partial q_i}{\partial x_i} - Q = 0 \tag{3.1}$$

where $H$ is the enthalpy of the material, $q_i$ is the heat flux vector, and $Q$ is a volumetric heating or source term. The enthalpy is a measure of the internal energy of the material and is written in this form to account for phase changes that may occur in the material. The volumetric heating term accounts for heat input that may occur from external sources or through chemical reactions. Both of these terms will be discussed in greater detail in the next section. Similar to the viscous terms in the Navier-Stokes energy equation (equation 2.3), the heat flux vector $q_i$ is a measure of the thermal energy flow and is typically written using Fourier's Law

$$q_i = -k_{ij} \frac{\partial T}{\partial x_i} \tag{3.2}$$

where $k_{ij}$ is the symmetric thermal conductivity tensor for the material and $T$ is the temperature of the solid. Using this definition of the heat flux, equation 3.1 is more commonly written as

$$\frac{\partial H}{\partial t} - \frac{\partial}{\partial x_i}\left(k_{ij}\frac{\partial T}{\partial x_j}\right) - Q = 0 \tag{3.3}$$

Equation 3.3 is completed by the Dirichlet boundary conditions

$$T = \bar{T}(\boldsymbol{x}, t) \quad \text{on} \quad \Gamma_T \tag{3.4}$$

which states that the temperature $\bar{T}$ may be prescribed on the Dirichlet boundary surface $\Gamma_T$ and the Neumann boundary conditions

$$q_i \cdot n_i = -\bar{q} - q_c - q_r \quad \text{on} \quad \Gamma_{\bar{q}} \tag{3.5}$$

which states the heat flux transferred normal to the boundary surface $\Gamma_{\bar{q}}$ is balanced by any prescribed heat flux $\bar{q}$, convective heat flux $q_c$, and/or radiative heat flux $q_r$. The convective heat

flux is often represented by Newton's Law of Cooling as

$$q_c = h_c(T - T_{ref}) \tag{3.6}$$

where $h_c$ is the convective heating coefficient and $T_{ref}$ is a reference or ambient temperature. The last term in equation 3.5 is the radiative flux and is represented by the equation

$$q^r = \epsilon^r \sigma(T^4 - T_\infty^4) \tag{3.7}$$

where $\epsilon^r$ is the radiative emissivity, $\sigma$ is the Stefan-Boltzmann constant, and $T_\infty$ is the far-field temperature the body is radiating to. Finally, initial conditions must be given to define the thermal state of the solid on $\Omega_T$

$$T(\boldsymbol{x}, t = 0) = T_0(\boldsymbol{x}) \quad \text{in} \quad \Omega_t \tag{3.8}$$

### 3.2.2 Thermal Material Models

Equation 3.3 is written with the enthalpy term $H$ in order to account for changes in phase of the solid. As a material changes phase, a latent heat effect occurs whereby energy is released or absorbed. This is a highly nonlinear effect and must be handled appropriately by the numerical method. The enthalpy of a solid may be written as a function of temperature according to the equation

$$H(T) = \int_{T_{ref}}^{T} \rho C_p(T) dT + \rho \mathcal{L}_h f(T) \tag{3.9}$$

where $\rho$ is the density of the solid, $C_p(T)$ is the temperature dependent specific heat, $\mathcal{L}_h$ is the latent heat of fusion, and $f$ is a function which specifies the volume fraction of liquid in the material. The "phase function" $f$ may have several forms; the simplest of which is a linear interpolation defined as

$$f(T) = \begin{cases} 0 & \text{if} \quad T < T_{solid} \\ 0 < f*(T) < 1 & \text{if} \quad T_{solid} < T \leq T_{liquid} \\ 1 & \text{if} \quad T > T_{liquid} \end{cases} \tag{3.10}$$

Note that if we assume the material does not change phase and that the specific heat is constant equation 3.3 may be re-written as

$$\rho C_p \frac{\partial T}{\partial t} - \frac{\partial}{\partial x_i}\left(k_{ij}\frac{\partial T}{\partial x_j}\right) - Q = 0 \ . \tag{3.11}$$

The the three-dimensional form of the symmetric thermal conductivity tensor $k_{ij}$ given in equation 3.3 may be written as

$$k_{ij} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \ . \tag{3.12}$$

If the material is assumed to behave in an isotropic fashion then the thermal conductivity tensor becomes

$$k_{ij} = \begin{bmatrix} k_{11} & 0 & 0 \\ 0 & k_{11} & 0 \\ 0 & 0 & k_{11} \end{bmatrix} \ . \tag{3.13}$$

In general, the components of $k_{ij} = k_{ij}\,(T)$ meaning that the thermal conductivity may vary with temperature.

## 3.3 Finite Element Discretization of the Transient Heat Equation

This section discusses the spatial discretization of the transient heat equation via a standard Galerkin finite element method and a stabilized finite element method. The stabilized form is used to suppress solution oscillations that sometimes arise with the Galerkin form, especially when a relatively high heat flux is applied to a low thermal conductivity material.

### 3.3.1 Galerkin Discretization

The standard Galerkin variational statement is obtained by multiplying the governing equation (3.3) by suitably defined test functions $W$

$$\int_{\Omega_t} W \cdot \left[\frac{\partial H}{\partial t} - \frac{\partial}{\partial x_i}\left(k_{ij}\frac{\partial T}{\partial x_j}\right) - Q\right] d\Omega = 0 \tag{3.14}$$

Integrating the second order spatial derivative term in equation 3.14 by parts reduces it to the sum of a first-order derivative volume integral and a surface integral

$$\int_{\Omega_t} W \cdot \left( \frac{\partial H}{\partial t} - Q \right) d\Omega - \left[ - \int_{\Omega_t} \frac{\partial W}{\partial x_i} \cdot \left( k_{ij} \frac{\partial T}{\partial x_j} \right) + \int_{\Gamma_t} W \left( k_{ij} \frac{\partial T}{\partial x_j} \right) n_i d\Gamma \right] = 0 \tag{3.15}$$

After rearranging the previous equation it becomes the familiar Galerkin weak statement for transient heat transfer

$$\int_{\Omega_t} \left[ W \cdot \left( \frac{\partial H}{\partial t} - Q \right) + \frac{\partial W}{\partial x_i} \cdot \left( k_{ij} \frac{\partial T}{\partial x_j} \right) \right] d\Omega - \int_{\Gamma_t} W \; q_i \hat{n_i} \; d\Gamma = 0 \tag{3.16}$$

Recognizing that the boundary flux integral in equation 3.16 is simply the Neumann boundary conditions given in equation 3.5 this may also be expressed as

$$\int_{\Omega_t} \left[ W \cdot \left( \frac{\partial H}{\partial t} - Q \right) + \frac{\partial W}{\partial x_i} \cdot \left( k_{ij} \frac{\partial T}{\partial x_j} \right) \right] d\Omega + \int_{\Gamma_t} W \cdot \left( \bar{q} + q^c + q^r \right) \; d\Gamma = 0 \tag{3.17}$$

This equation is the familiar Galerkin weak statement for the transient heat equation.

### 3.3.2 Galerkin Gradient Least Squares Discretization

The Galerkin weak statement given by equation 3.17 has been known to produce non-physical solution oscillations for certain problems. In particular, problems with high boundary heat fluxes and low thermal conductivity of the solid fall under this category, as described by Fachinotti [37]. Given this deficiency, Ilinca [72] applied the Galerkin gradient least-squares (GGLS) method developed by Franca et al. [46] to stabilize errors arising from the poor spatial approximation in these cases. Given that problems of interest to aerospace applications necessarily involve tremendous heat fluxes and possibly low thermal conductivity materials, the GGLS approach was pursued and implemented.

Similar to the SUPG method shown in the previous chapter, the GGLS formulation is a residual-based method that aims to add dissipation were needed in order to obtain a better behaved numerical scheme. The GGLS analog to the test function perturbations introduced by the SUPG method in equation 2.39 is written as

$$\hat{W} = W + \frac{\partial W}{\partial x_i} k \tau_{ggls} \; . \tag{3.18}$$

For the sake of simplicity, the thermal conductivity tensor $k_{ij}$ is represented here as an isotropic, scalar thermal conductivity $k$. In this equation, a perturbation proportional to the test function derivative and the thermal conductivity $k$ is scaled by the stabilization parameter $\tau_{ggls}$. Computation of $\tau_{ggls}$ is deferred to a later section for the time being. The test function perturbation applies to the strong form of the residual given by equation 3.3, hence the GGLS weak form is written as

$$\int_{\Omega_t} \left[ W \cdot \left( \frac{\partial H}{\partial t} - Q \right) + \frac{\partial W}{\partial x_i} \cdot \left( k_{ij} \frac{\partial T}{\partial x_j} \right) \right] d\Omega + \int_{\Gamma_t} W \cdot (\bar{q} + q^c + q^r) \; d\Gamma + \sum_{e=1}^{n_e} \int_{\Omega_t^e} \frac{\partial W}{\partial x_i} k \tau_{ggls} \cdot \frac{\partial}{\partial x_i} \left[ \frac{\partial H}{\partial t} - \frac{\partial}{\partial x_i} \left( k_{ij} \frac{\partial T}{\partial x_j} \right) - Q \right] = 0 \quad (3.19)$$

The summation symbol in equation 3.19 indicates that the stabilization occurs only over element interiors. Note that stabilization is added via the *gradient* of the strong form of the residual and hence has the greatest influence in the regions where the spatial change in the residual is large.

### 3.3.3 Finite Element Spatial Approximation

The solution of equations 3.17 and 3.19 is obtained by first discretizing the domain $\Omega$ into elements $\Omega^e$ and solving for approximate temperature solutions $T^h$ of the weak statement. The approximate trial solutions $T^h$ are sought by solving the weak statement represented by a linear combination of the test functions $W^h$. The trial solution and test function spaces are defined in a similar manner to equations 2.43 and 2.44 from the previous chapter.

$$\mathcal{S}^h = \left[ T^h \in \left[ C^0(\Omega) \right]^{n_{dof}} \; , \; T^h \mid_{\Omega^e} \in \left[ P^k(\Omega^e) \right]^{n_{dof}} \; , \; T^h = \bar{T} \text{ on } \Gamma_t \right] \quad (3.20)$$

which states that the trial solution must be $C^0$ continuous, representable by an interpolation polynomial $P^k$ of order $k$, and must satisfy the Dirichlet conditions on the boundary. Similarly, the test function space is defined by

$$\mathcal{V}^h = \left[ W^h \in \left[ C^0(\Omega) \right]^{n_{dof}} \; , \; W^h \mid_{\Omega^e} \in \left[ P^k(\Omega^e) \right]^{n_{dof}} \; , \; W^h = 0 \text{ on } \Gamma_t \right] \; . \quad (3.21)$$

The standard approximation for T is represented using element basis functions via a split in the time and spatial dependence of the variable according to the equation

$$T^h(\boldsymbol{x},t) = \sum_{n=1}^{n_n} N^n(\boldsymbol{x}) T^n(t) \ . \tag{3.22}$$

where $n = 1, ..., n_n$ symbolizes the nodes of the element.

### 3.3.4 GGLS Stabilization Parameter

The stabilization parameter used here for the GGLS formulation is the same one defined by Franca [46]. $\tau_{ggls}$ is written as

$$\tau_{ggls} = \frac{h^2}{6k}\bar{\xi} \tag{3.23}$$

where

$$\bar{\xi} = \frac{\cosh\left(\sqrt{6\alpha}\right) + 2}{\cosh\left(\sqrt{6\alpha}\right) - 1} - \frac{1}{\alpha} \tag{3.24}$$

$$\alpha = \frac{\left(\rho C_p/\Delta t\right) h^2}{6k} \tag{3.25}$$

and $h$ is the element size which is defined here to be

$$h_T = 2\left(\sum_{n=1}^{n_n} \mid \boldsymbol{r}_e \cdot \nabla N^n \mid\right)^{-1} \quad \text{where} \quad \boldsymbol{r}_e = \frac{\nabla T}{\|\nabla T\|} \ , \tag{3.26}$$

The stabilization parameter used here was derived for one-dimensional GGLS problems, however as reported by reference [72] and as observed here the stabilization parameter also performs reasonably well for multi-dimensional problems.

## 3.4 Ablation Modeling

Hypersonic re-entry vehicles often need some form of thermal protection system in order to mitigate the extremely high heat fluxes and resulting thermal conduction into the body while passing through the atmosphere. Perhaps the most commonly used method of reducing the heating of the structure is through the use of an actively ablating material. The intent of an ablator is to absorb and remove as much of the heat energy as possible through mass loss via chemical

decomposition of the material. The intent of this section is to overview two types of surface ablation models and discuss the implementation of one in terms of solving the heat conduction equation on a moving boundary.

The physics of the hypersonic flow environment was discussed in the previous chapter yet it is important to reiterate that chemical reactions often come into play along with viscous effects of the body. A chemically reacting ablator will inject species into the flow field which may in turn react with the fluid species. Unfortunately, it is outside the scope of this thesis to incorporate thermochemical flow effects thus accounting for the chemical interactions of the ablator and fluid. For the purpose of demonstrating the monolithic coupling approach used in this dissertation the simpler Q* discussed here was chosen. For more details on sophisticated thermochemical ablation modeling techniques see, for instance, the work of Amar and Blackwell [7, 8] and Chen and Milos [29, 107].

### 3.4.1 Heat of Ablation (Q*) Model

The so-called "Heat of Ablation" model accounts for the amount of energy absorbed by the material as it changes phase from a solid to a liquid or gaseous state. The Q* model uses a latent heat to approximate the energy consumed at the ablating surface and makes no attempt to model in detail what may be occurring in terms of chemical reactions. For many problems this model is able to give a rough approximation of the surface energy balance and recession rate. However, its use is often entirely insufficient for accurate prediction of many modern ablators that require the in-depth modeling of the chemical decomposition process.

The flux boundary condition as written by equation 3.5 is now written including a latent heat term that accounts for recession of boundary as the material at the surface changes phase

$$q_i \cdot n_i = -\bar{q} - q^c - q^r + q^* \quad \text{on} \quad \Gamma_q \tag{3.27}$$

where $q*$ is written as

$$q^* = \rho \mathcal{L}_h \dot{s} \quad \text{on} \quad \Gamma_q \tag{3.28}$$

where $\mathcal{L}_h$ is the latent heat due to phase change and $\dot{s}$ is the surface recession velocity.

### 3.4.2 Thermochemical Equilibrium Ablation Model

For a thermochemical ablation model the surface energy balance is written as

$$q_i \cdot n_i = -\bar{q} - q^c - q^r + q^{tc} \quad \text{on} \quad \Gamma_q \tag{3.29}$$

where $q^{tc}$ is written as

$$q^{tc} = \rho\dot{s}\,(h_w - h_c) \quad \text{on} \quad \Gamma_q \tag{3.30}$$

In the previous equation, $\dot{s}$ is the surface recession velocity due to thermochemical ablation and $h_w$ and $h_c$ are the specific enthalpies of the gas at the wall and the enthalpy of the charred ablator. The char enthalpy is defined as

$$h_c = h_{ref} + \int_{T_{ref}}^{T} C_p(T)\; dT \tag{3.31}$$

and the product $\rho\dot{s}$ is defined as

$$\rho\dot{s} = C_m B' \tag{3.32}$$

where $C_m$ is a mass transfer coefficient and $B'$ represents a dimensionless ablation mass transfer rate. It is typically assumed that the mass transfer coefficient is equal to the convective heat transfer coefficient $h_c$ in equation 3.6. The $B'$ values must be computed for a specific material given a wall temperature and pressure. The equilibrium chemistry code ACE [118] is a commonly used tool for performing this task. The thermochemical equilibrium ablation development shown here is similar to that shown by Kuntz et al. [86]; the reader is referred to this reference for further information.

## 3.5 Solution Strategies for the Transient Heat Conduction Equation

### 3.5.1 Nonlinear Solution via Newton's Method

Newton's method is used to solve the nonlinear equations arising from the weak formulation of equations 3.17 and 3.19, the basics of which were previously discussed in section 2.4.1.

### 3.5.2 Semi-Discrete Residual Equation

The Galerkin gradient least-squares weak residual equation is given by

$$\mathfrak{R}_t(\dot{T},T) \equiv \int_{\Omega_t} \left[ W \cdot \left( \rho C_p(T^h)\dot{T}^h + \rho \mathcal{L}_h \dot{f}(T) - Q \right) + \frac{\partial W}{\partial x_i} \cdot \left( k_{ij}(T^h) \frac{\partial T}{\partial x_j} \right) \right] d\Omega + \int_{\Gamma_t} W \cdot (\bar{q} + q_c + q_r) \; d\Gamma + \sum_{e=1}^{n_e} \int_{\Omega_t^e} \frac{\partial W}{\partial x_i} k\tau_{ggls} \cdot \frac{\partial}{\partial x_i} \left[ \rho C_p(T^h)\dot{T} + \rho \mathcal{L}_h \dot{f}(T) - \frac{\partial}{\partial x_i} \left( k_{ij}(T^h) \frac{\partial T}{\partial x_j} \right) - Q \right] = 0 \quad (3.33)$$

The weight functions used to interpolate the solution are defined by the vector

$$\boldsymbol{W}^e = [W_1 \quad W_2 \quad ... \quad W_n] \quad (3.34)$$

Using the weight function vector $\boldsymbol{W}$ the approximate solution defined in equation 3.22 and its time derivative may be written simply as the dot product with the elements nodal state variables

$$T^h(\boldsymbol{x},t) = \boldsymbol{W}^T \boldsymbol{T}^e \quad (3.35)$$

$$\dot{T}^h(\boldsymbol{x},t) = \boldsymbol{W}^T \dot{\boldsymbol{T}}^e \quad (3.36)$$

The previous equation can now be cast into a semi-discrete matrix-vector form

$$\mathfrak{R}_t(\dot{\boldsymbol{T}},\boldsymbol{T}) \equiv \boldsymbol{\mathcal{R}}_t(\dot{\boldsymbol{T}},\boldsymbol{T}) + \boldsymbol{R}_t(\boldsymbol{T}) \quad (3.37)$$

where the dynamic residual is computed as

$$\boldsymbol{\mathcal{R}}_t(\dot{\boldsymbol{T}},\boldsymbol{T}) = \sum_{e=1}^{n_e} \boldsymbol{\mathcal{R}}_t^e(\dot{\boldsymbol{T}}^e,\boldsymbol{T}^e) \quad (3.38)$$

The element-level nonlinear dynamic residual contribution is expressed by

$$\boldsymbol{\mathcal{R}}_t^e(\dot{\boldsymbol{T}},\boldsymbol{T}) \equiv \boldsymbol{\mathcal{R}}_{t,galerkin}^e + \boldsymbol{\mathcal{R}}_{t,ggls}^e \; . \quad (3.39)$$

$\boldsymbol{\mathcal{R}}_{t,galerkin}^e$ is the dynamic portion of the standard Galerkin residual in which any material nonlinearities may exist

$$\boldsymbol{\mathcal{R}}_{t,galerkin}^e = \int_{\Omega^e} W^T \, \rho \, C_p(T^h)\dot{T}^h \; d\Omega + \int_{\Omega^e} W^T \rho \mathcal{L}_h \dot{f}(T^h) \; d\Omega \quad (3.40)$$

and $\dot{f}$ is the time derivative of the phase function field. $\boldsymbol{\mathcal{R}}^e_{t,ggls}$ accounts for the GGLS contribution to the time dependent part of the residual equation

$$\boldsymbol{\mathcal{R}}^e_{t,ggls} = \int_{\Omega^e_t} \frac{\partial \boldsymbol{W}^e{}^T}{\partial x_i} k_{ij}(T^h)\, \tau_{ggls}(T^h) \cdot \frac{\partial}{\partial x_i}\left[\rho C_p(T^h)\dot{T}^h + \rho \mathcal{L}_h \dot{f}\right]\, d\Omega \tag{3.41}$$

The nonlinear static residual $\boldsymbol{R}_t(T)$ is assembled from element residual contributions in the same way the dynamic residual is built. The element-level static residual is defined according to the equation

$$\boldsymbol{R}^e_t(T) \equiv \boldsymbol{R}^e_{t,galerkin} + \boldsymbol{R}^e_{t,ggls}\ . \tag{3.42}$$

$\boldsymbol{R}^e_{t,galerkin}$ is the steady portion of the standard Galerkin treatment of the heat equation and is expressed as

$$\boldsymbol{R}^e_{t,galerkin} = \int_{\Omega^e_t} \frac{\partial W^T}{\partial x_i} \cdot \left[k_{ij}(T^h)\frac{\partial T^h}{\partial x_j}\right]\, d\Omega - \int_{\Omega^e_t} W^T \cdot Q(T^h)\, d\Omega + \int_{\Gamma^e_t} W^T \cdot (\bar{q} + q_c + q_r)\, d\Gamma \tag{3.43}$$

If the GGLS formulation is included, the element static residual $\boldsymbol{R}^e_t$ is augmented by the GGLS residual for the heat equation

$$\boldsymbol{R}^e_{t,ggls} = \int_{\Omega^e} \frac{\partial W^T}{\partial x_i} k_{ij}(T^h)\, \tau_{ggls}(T^h) \cdot \frac{\partial}{\partial x_i}\left[-\frac{\partial}{\partial x_i}\left(k_{ij}(T^h)\frac{\partial T^h}{\partial x_j}\right) - Q(T^h)\right]\, d\Omega \tag{3.44}$$

It is important to note that the third-order spatial derivatives arising from this equation vanish for linear or quadratic element basis functions. In many cases this term is dropped from consideration.

### 3.5.3 Jacobian Calculation

Differentiating the residual statement (equation 3.33) with respect to $T$ we obtain the Jacobian expression

$$\begin{aligned}\frac{\partial \boldsymbol{\mathfrak{R}}(\dot{T},T)}{\partial T} = \boldsymbol{\mathfrak{J}}(\dot{T},T) \equiv \\ \int_{\Omega_t} W \cdot \left[\rho\frac{\partial C_p(T)}{\partial T}\dot{T} + \rho C_p(T)\frac{\partial \dot{T}}{\partial T} + \rho\mathcal{L}_h\frac{\partial \dot{f}(T)}{\partial T} - \frac{\partial Q(T)}{\partial T}\right]\, d\Omega + \\ \int_{\Omega_t} \frac{\partial W}{\partial x_i} \cdot \left(\frac{\partial k_{ij}(T)}{\partial T}\frac{\partial T}{\partial x_j} + k_{ij}(T)\frac{\partial W}{\partial x_j}\right) d\Omega\ + \int_{\Gamma} W \cdot \left(\bar{q} + \frac{\partial q_c}{\partial T} + \frac{\partial q_r}{\partial T}\right)\, d\Gamma\ + \\ \sum_{e=1}^{n_e} \int_{\Omega^e_t} \frac{\partial W}{\partial x_i}\, k_{ij}(T)\, \tau_{ggls}(T) \cdot \frac{\partial}{\partial x_i}\left[\frac{\partial}{\partial T}\left(\mathcal{L}T - Q(T)\right)\right]\end{aligned} \tag{3.45}$$

The previous equation may be written compactly as

$$\mathfrak{J}(\dot{T},T) \equiv \boldsymbol{\mathcal{J}}_t(\dot{T},T) + \boldsymbol{J}_t(T) \tag{3.46}$$

where the dynamic Jacobian is computed as

$$\boldsymbol{\mathcal{J}}(\dot{T},T) = \sum_{e=1}^{n_e} \boldsymbol{\mathcal{J}}_t^e(\dot{T}^e,T^e) \tag{3.47}$$

The element-level nonlinear dynamic Jacobian contribution is expressed by

$$\boldsymbol{\mathcal{R}}_t^e(\dot{T},T) \equiv \boldsymbol{\mathcal{J}}_{t,galkerin}^e + \boldsymbol{\mathcal{J}}_{t,ggls}^e \tag{3.48}$$

where $\boldsymbol{R}_{t,galerkin}^e$ is the standard Galerkin Jacobian in which any material nonlinearities may be accounted for

$$\boldsymbol{J}_{t,galerkin}^e = \int_{\Omega^e} W^T \cdot \left[\rho \frac{\partial C_p(T^h)}{\partial T}\dot{T}^h + \rho C_p(T^h)\frac{\partial \dot{T}^h}{\partial T}\right] \, d\Omega \tag{3.49}$$

and $\dot{f}$ is the time derivative of the phase function field. $\boldsymbol{J}_{t,ggls}^e$ accounts for the derivatives of the GGLS contribution to the time dependent part of the residual equation

$$\boldsymbol{J}_{t,ggls}^e = \int_{\Omega_t^e} \frac{\partial W}{\partial x_i}^T k_{ij}(T^h)\,\tau_{ggls}(T^h) \cdot \frac{\partial}{\partial x_i}\left[\rho C_p(T^h)\dot{T}^h + \rho \mathcal{L}_h \dot{f}\right] \, d\Omega \tag{3.50}$$

The static Jacobian $\boldsymbol{J}_t$ is assembled from element Jacobian contributions in the same way the dynamic Jacobian is built and is defined according to the equation

$$\boldsymbol{J}_t^e(T) \equiv \boldsymbol{J}_{t,galerkin}^e + \boldsymbol{J}_{t,ggls}^e \tag{3.51}$$

where

$$\boldsymbol{J}_Q^{e^s} = \tag{3.52}$$

$$\begin{aligned}\boldsymbol{J}_{t,galerkin}^e = &\int_{\Omega_t^e} \frac{\partial W}{\partial x_i}^T \cdot \left(\frac{\partial k_{ij}(T^h)}{\partial T}\frac{\partial T^h}{\partial x_j} + k_{ij}(T^h)\frac{\partial W}{\partial x_j}\right) d\Omega - \\ &\int_{\Omega_t^e} W^T \cdot \left[\rho \mathcal{L}_h \frac{\partial \dot{f}(T^h)}{\partial T} - \frac{\partial Q(T^h)}{\partial T}\right] d\Omega + \int_{\Gamma_t^e} W^T \cdot \left(\bar{q} + \frac{\partial q_c}{\partial T} + \frac{\partial q_r}{\partial T}\right) d\Gamma\end{aligned} \tag{3.53}$$

and if the GGLS formulation is included its static residual contribution is defined by

$$\boldsymbol{J}_{t,ggls}^e = \int_{\Omega_t^e} \frac{\partial W}{\partial x_i}^T k_{ij}(T^h)\,\tau_{ggls}(T^h) \cdot \frac{\partial}{\partial x_i}\left[\frac{\partial}{\partial T}\left(\rho C_p(T^h)\frac{\partial T}{\partial t} - \frac{\partial}{\partial x_i}\left(k_{ij}(T^h)\frac{\partial T}{\partial x_j}\right) - Q(T^h)\right)\right] d\Omega \tag{3.54}$$

where the derivatives of $k_{ij}(T^h)$ and $\tau_{ggls}(T^h)$ have been ignored for the sake of simplicity.

### 3.5.4 Time Integration via the Θ-Scheme

The semi-discrete problem posed by equation 3.37 is solved here via the Θ-scheme time integrator. As an alternative to semi-discretization, the space-time finite element methods, which discretize both the temporal and spatial parts of the PDE via finite element methods, have been successfully applied to solving first-order in time PDEs [10].

The Θ integrator defines an intermediate state at which the residual is built. The intermediate state is defined by

$$\tilde{\boldsymbol{T}} = \Theta \boldsymbol{T}^{n+1} + (1-\Theta)\boldsymbol{T}^{n} \tag{3.55}$$

and the time derivative of the state is computed by the backward difference

$$\dot{\boldsymbol{T}} = \frac{\boldsymbol{T}^{n+1} - \boldsymbol{T}^{n}}{\Delta t} \tag{3.56}$$

The residual and Jacobian given by equations 3.33 and 3.45 are now computed as functions of $\tilde{\boldsymbol{T}}$ and $\dot{\boldsymbol{T}}$. Setting $\Theta = 1$ yields the backward Euler integrator and $\Theta = 0.5$ produces a second-order accurate mid-point rule integrator.

The adaptive time step selection process given by equation 2.112 also applies to this integrator. Since the Θ-scheme is a two-point method (using only $\boldsymbol{T}^{n+1}$ and $\boldsymbol{T}^{n}$) no special form of the integrator accounting for previous time step sizes is needed, as was the case with the BDF scheme presented in the last chapter. It is also possible to use the BDF integrator to solve the heat transfer equations presented here as well as use the Θ-scheme to solve the Navier-Stokes equations of the last chapter.

## 3.6 Arbitrary Lagrangian-Eulerian Form of the Heat Equation

### 3.6.1 ALE Form of the Governing Equation

In the event the mesh is moving while solving the transient heat equation (as may be the case in ablation of the solid), an arbitrary Lagrangian-Eulerian (ALE) frame of reference is adopted in order correct for advection of the material introduced by the mesh motion. Consequently, equation

3.3 must be re-written as

$$\frac{\partial H}{\partial t} + v_i^m \frac{\partial H}{\partial x_i} + \frac{\partial}{\partial x_i}\left(k_{ij}\frac{\partial T}{\partial x_j}\right) - Q = 0 \tag{3.57}$$

where $v_i^m$ is the mesh motion velocity. This is consistent with replacing the partial time derivative $\partial/\partial t$ with the total derivative

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + v_i \frac{\partial}{\partial x_i} \tag{3.58}$$

which describes the rate of change of a quantity that is both changing in space and time.

### 3.6.2 Time Accurate ALE Integration

Time accuracy of an integrator solving the ALE form of the governing equation 3.57 is automatically satisfied on a moving grid due to the nature of the equations. The energy equation solved for transient heat transfer is a pure diffusion process, with no advective terms inherent to the equation. As noted by Geuzaine [47], diffusive fluxes do not affect the DGCL, only advective fluxes. Since the only advection in equation 3.57 is introduced by the non physical and arbitrary mesh motion the DGCL is not a consideration. Simple numerical experiments can be done to prove this statement.

However, there is another issue related to the time accuracy which arises in the case of excessive mesh motion. If the non-dimensional Peclet number, defined as

$$Pe = \frac{\rho\, v^m\, h\, C_p}{2k} \tag{3.59}$$

which is a measure of the advective and diffusive scales, is greater than one then the spatial approximation may exhibit oscillations in the solution. This is related to the same cause of instability of a finite element scheme for an advection dominated flow. Hence, stabilization techniques such as an SUPG, GLS, or GGLS method may be used to stabilize these effects.

## 3.7 Numerical Example Problems

### 3.7.1 2-Dimensional Heat Transfer on a Moving Mesh

This example solves the steady state heat transfer on a moving mesh and is based on a problem from the ANSYS Verification Manual [75].

#### 3.7.1.1 Problem Description

Figure 3.2 shows the geometry and boundary conditions used for this problem. The solid material has a thermal conductivity of $k = 54\ W/m \cdot K$, density of $\rho = 7833\ kg/m^3$, and thermal capacitance of $C_p = 465\ J/kg \cdot K$. The top surface is heated by a convective heat flux where $h = 50\ W/m^2 \cdot K$ and a ambient reference temperature of $T_a = 1000\ K$. The bottom of the domain has an imposed Dirichlet boundary condition of $T = 0\ K$. A sinusoidal forcing function is applied to an elastic system governing the motion of the mesh, which generates the mesh deformations.

#### 3.7.1.2 Computational Mesh and Model

Figure 3.2 shows the 11 element mesh used for this problem. The problem is integrated to steady state via the $\Theta$-scheme time integrator using an adaptive time stepping method. The SuperLU direct linear solver is used for computing the nonlinear update within the Newton solver. Note this problem is linear in nature, hence the nonlinear solver computes the exact solution in one Newton step for every time iteration.

#### 3.7.1.3 Results and Discussion

Figures 3.3(a) - 3.3(e) show the temperature contour plots for several positions of the mesh during the simulation. It is interesting to note that although the mesh is moving, the steady-state response remains unaltered by the mesh motion.

Figure 3.4 shows the temperature profile vertically along the edge of the domain at the same points in time shown in figures 3.3(a) and 3.3(d).

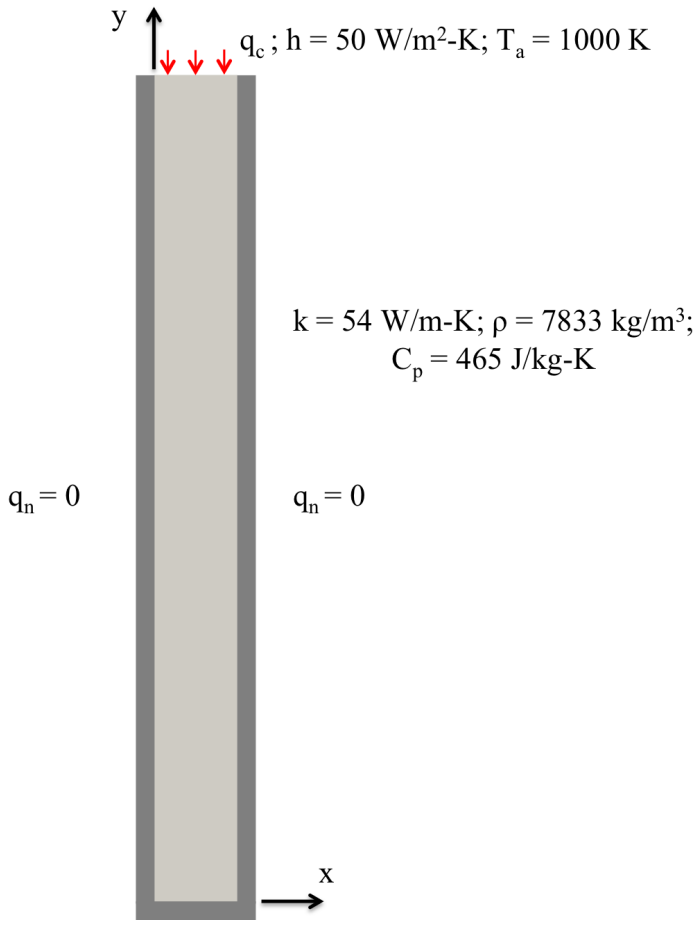



Figure 3.1: Geometry and boundary conditions for heat transfer on a moving grid.

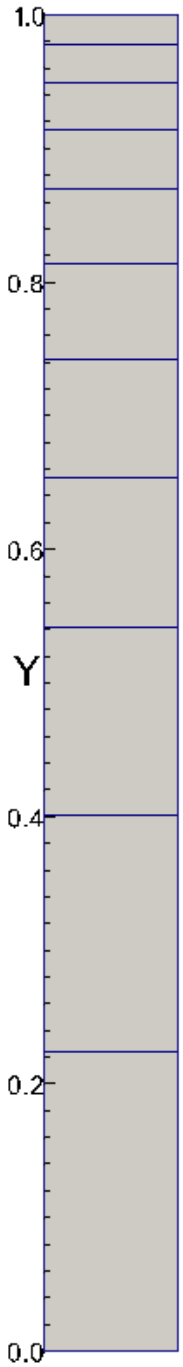
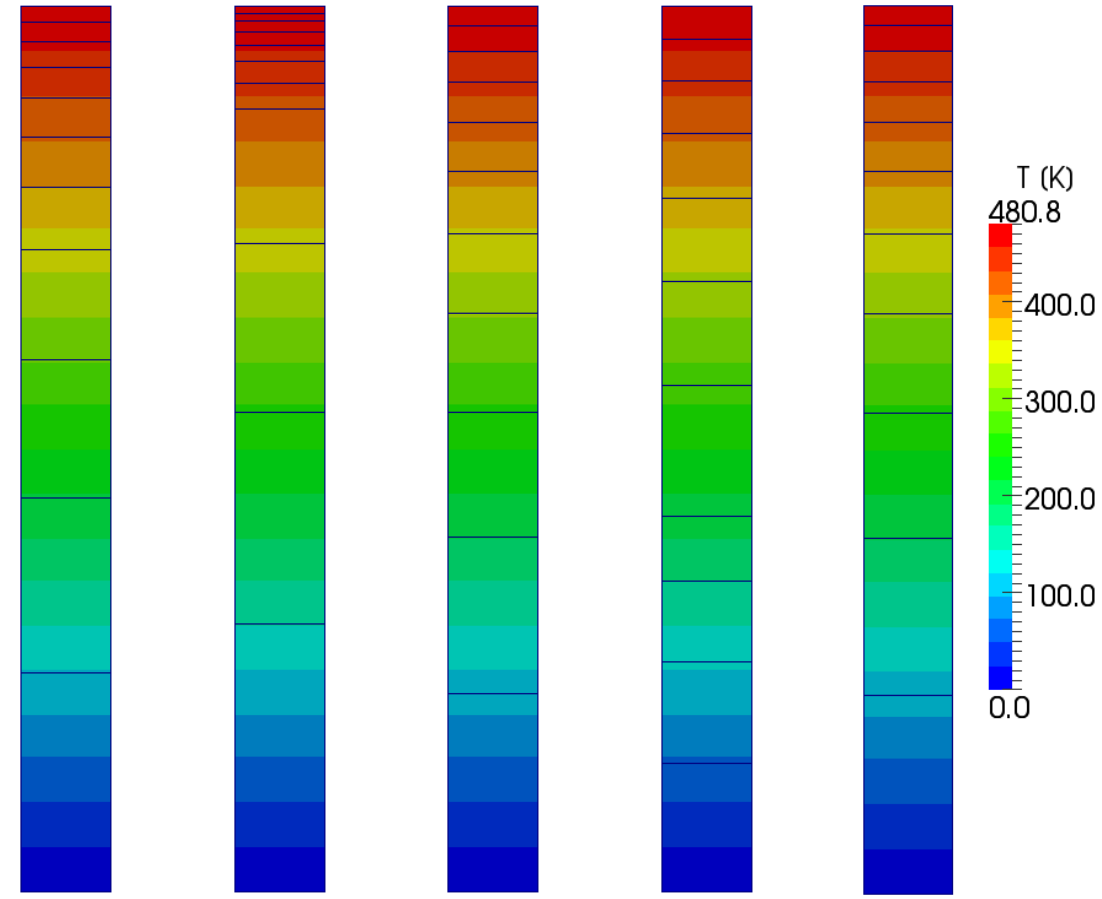



Figure 3.2: 11 element computational mesh used for the moving grid heat transfer problem.




(a) $t = 1000\ s$ (b) $t = 5000\ s$ (c) $t = 10000\ s$ (d) $t = 15000\ s$ (e) $t = 20000\ s$

Figure 3.3: Temperature contours at various mesh positions in time.

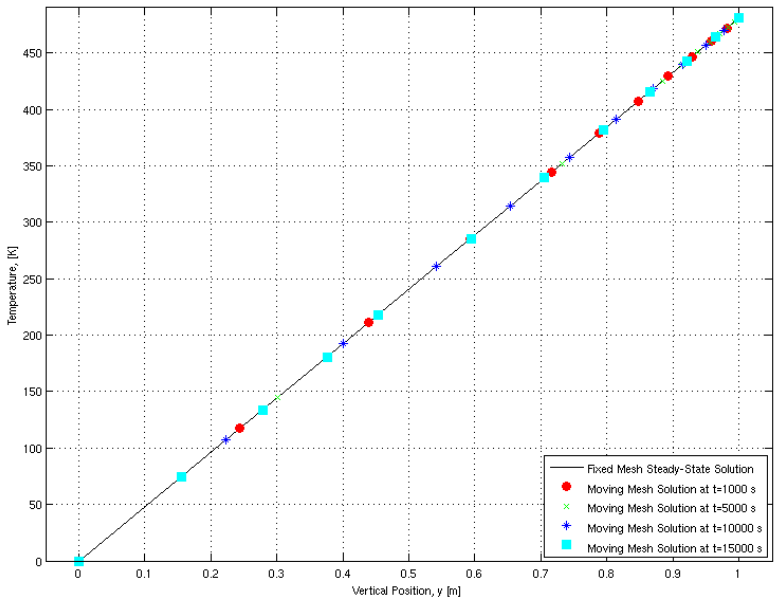



Figure 3.4: Temperatures along the vertical edge of the domain for the undeformed steady-state configuration and two other mesh positions and instances in time.

### 3.7.2 2-Dimensional Phase Change Example

In this example the phase change formulation presented in section 3.2.2 is used to demonstrate its capability. This problem illustrates the benefit of utilizing phase changing materials with high latent heat as a means of altering internal heat flow for design purposes. This concept will be further explored later in this thesis; at present we simply consider the phase change formulation as an analysis capability.

#### 3.7.2.1 Problem Description

The geometry and boundary conditions are shown in Figure 3.9. The geometry is divided into two distinct material zones; the inner and outer ring consist of a tungsten-like material with $k = 200.0\ W/m \cdot K$, $\rho = 19,300\ kg/m^3$, $c = 134\ J/kg \cdot K$, and the middle ring of material is an aluminum alloy with $k = 210.0\ W/m \cdot K$, $\rho = 2700\ kg/m^3$, $c = 950\ J/kg \cdot K$. The tungsten material changes phase between $T_s = 3300\ K$ and $T_l = 3305\ K$ with a latent heat of $1.84 \times 10^5\ J/K$, while the aluminum alloy changes phase between $T_s = 900\ K$ and $T_l = 925\ K$ with a latent heat of $3.80 \times 10^5\ J/K$. The relatively high latent heat and low density of the aluminum alloy makes it an intriguing material in terms of thermal energy absorption.

The outer surface of the cylinder is heated by a constant $2.0 \times 10^6\ W/m^2$ heat flux and the remaining boundaries are treated as adiabatic surfaces. The entire structure is at an initial uniform temperature of $800.0\ K$ when the simulation begins.

#### 3.7.2.2 Computational Mesh and Model

The problem is solved using a mesh with 922 quadrilateral elements and is shown in figure 3.6.

The problem is solved via the theta-scheme time integrator scheme using $\Theta = 0.5$ and a constant time step of $0.01\ s$. A Newton scheme is used to solve the nonlinear residual equations. The nonlinear iteration loop exits when the nonlinear residual drops at least four orders of magnitude. Before the temperature of the structure reaches the melting temperature of either material, the

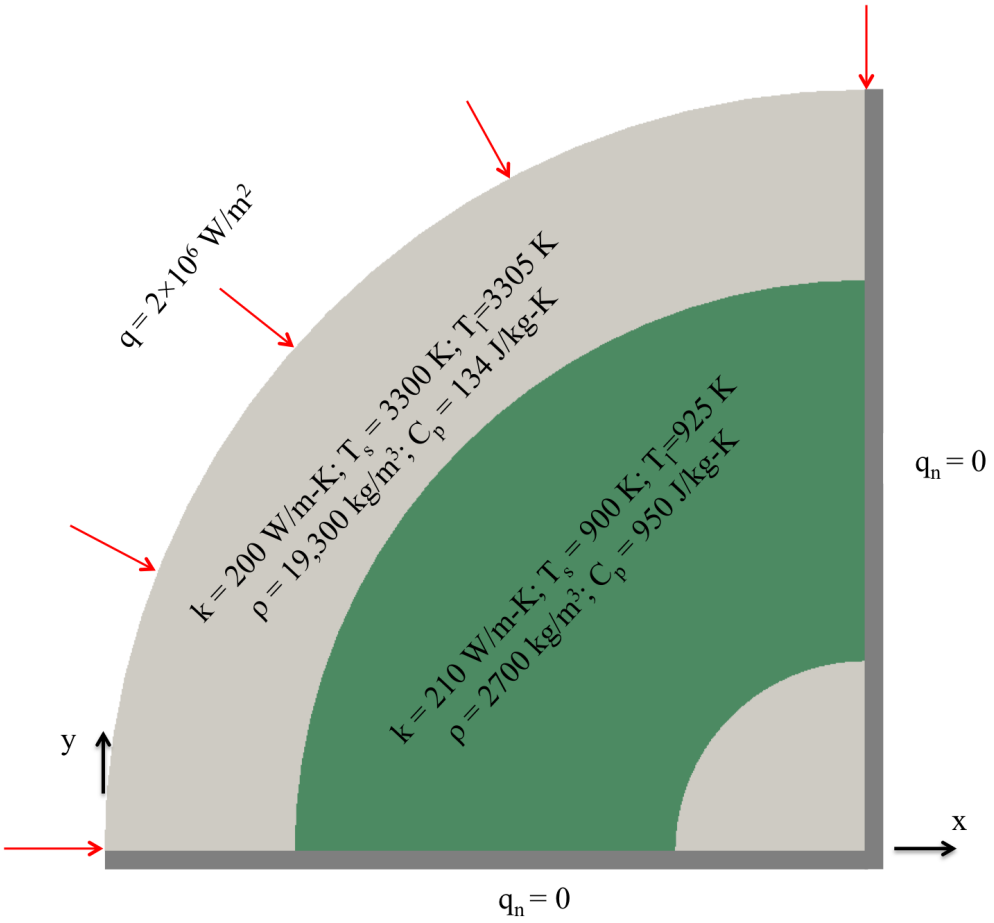



Figure 3.5: Geometry and boundary conditions for the quarter cylinder phase change problem.

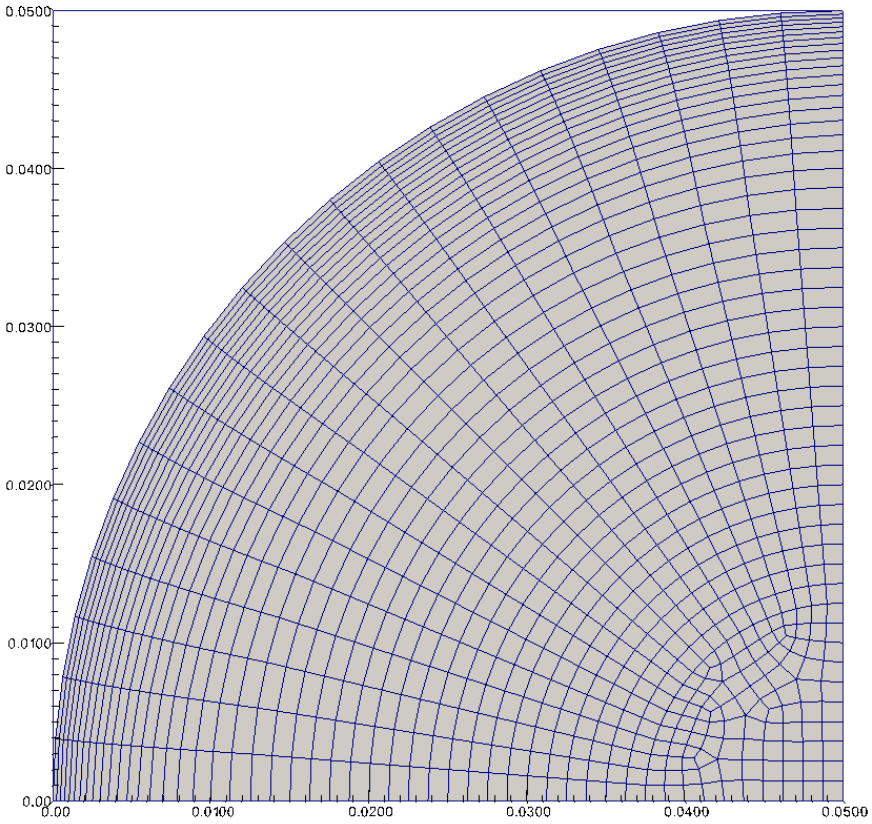



Figure 3.6: 922 quadrilateral element mesh for the quarter cylinder phase change problem

problem behaves linearly and only a single nonlinear solve is required to converge within the Newton loop. However, due to the nonlinear nature of the phase change process once a material begins to change phase the problem suddenly becomes a nonlinear one. In many situations some amount of under-relaxation of the nonlinear update computed by the Newton solve can actually speed convergence over no relaxation. In order to accommodate these two situations, a multi-level Newton solver is used. The first level specifies a maximum of 2 Newton solves and no relaxation of the Newton update. The solver enters the second level if the first level fails to reduce the residual to the tolerance specified above, as is the case when the problem becomes nonlinear. The second level specifies a maximum of 20 Newton solves and an update relaxation of 0.8. This setup has proven to noticeably accelerate the overall time integration over a single-level Newton solver. Trilinos' Aztec/GMRES iterative solver is used to solve the linearized equations generated by the Newton solver. The ML multi-level preconditioner available within Trilinos is used in conjunction with the GMRES. The GMRES/ML combination provides a very fast linear solver capability and for many problems of this type is over two times faster than some of the fastest direct solvers available.

#### 3.7.2.3 Results and Discussion

Figure 3.7 shows the temperature and phase functions fields at the end of the simulation for the problems with and without phase change. It is qualitatively clear from this figure that the problem with phase change leads to a much lower temperature field than the non-phase changing problem.

Figure 3.8 shows the temperature histories of the center node for an all tungsten cylinder and for one in which the middle material zone is aluminum. This plot makes it very apparent that the phase changing material "shields" the inner material by absorbing energy through the latent heat effect. As one can see, from roughly 6.0 $s$ to 16.0 $s$ the temperature rise at this node levels off. Once the aluminum has completely changed phase after this time, the zone continues heating at roughly the same rate as the all tungsten problem. The end result, however, is that at the end of the simulation (20.0 $s$ in this case), the difference in temperature between the all tungsten and

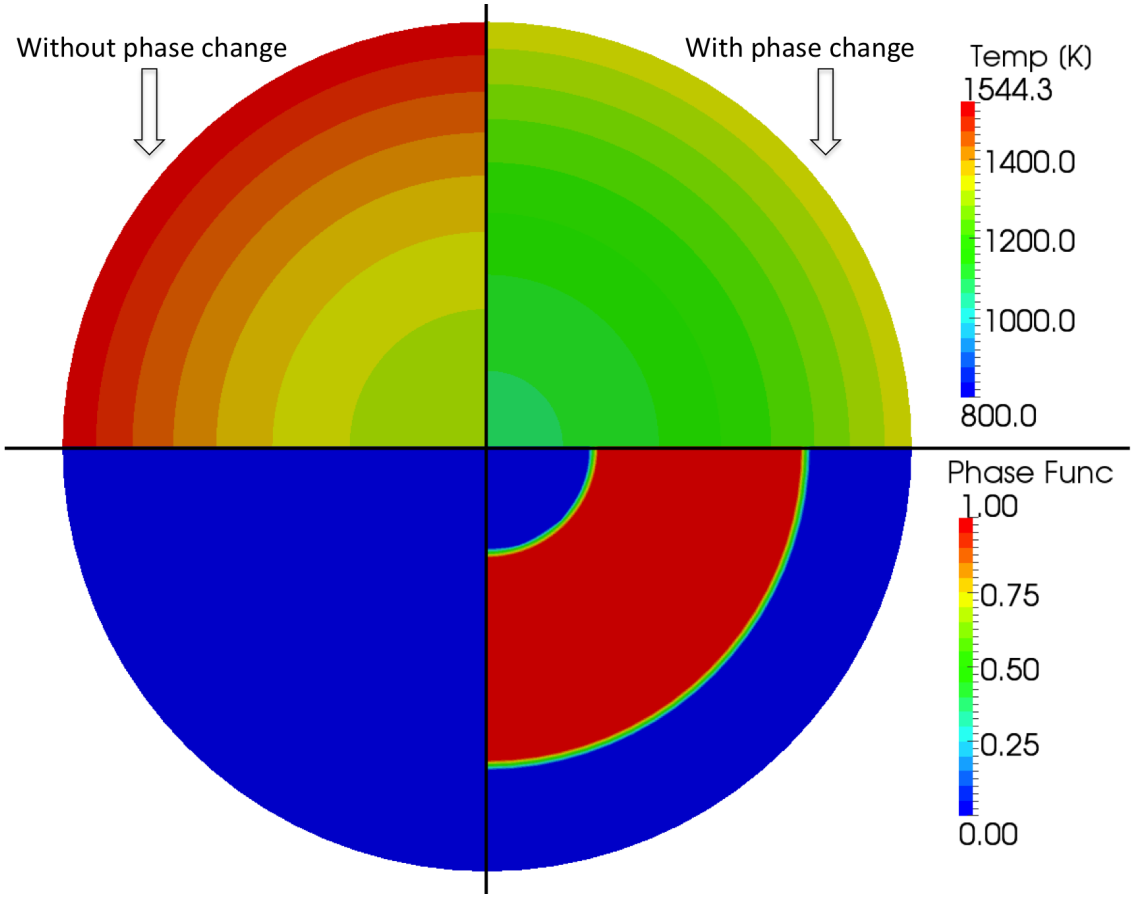



Figure 3.7: Temperature and phase function contour plot comparisons. The top half of the figure displays temperature contours and the bottom half shows the phase function value at the end of the simulation (20 $s$). The left half shows the non-phase changing problem, and the right shows the phase changing problem.

tungsten/aluminum problems is roughly 200 $K$. This is an observation we aim to exploit in a later chapter on topology optimization.

### 3.7.3 Axisymmetric GGLS Example

This example problem highlights the use of the GGLS formulation in a situation where the standard Galerkin method exhibits wild undershoots in the solution.

#### 3.7.3.1 Problem Description

The geometry is a half-sphere modeled by an axisymmetric quarter-circle of radius 0.02 $m$ and material properties of $\rho = 2700\ kg/m^3$, $C_p = 896\ J/(kg \cdot K)$ and $k = 17.0\ W/(m \cdot K)$. A heat flux of $\bar{q} = 1.0 \times 10^9$ is applied to the curved boundary of the quarter-circle, all other edges are adiabatic (i.e. $\bar{q} = 0$). The initial temperature of the solid is set to 300 $K$.

#### 3.7.3.2 Computational Mesh and Model

This problem is solved using two different meshes. The first mesh is relatively coarse (considering the high heating rate it experiences), consisting of only 601 element, and is shown in figure 3.10(a). The second mesh is significantly more refined and contains 40,409 elements and is shown in figure 3.10(b).

This problem is solved via five backward Euler time steps that are taken with a fixed time step size of $\Delta t = 0.001\ s$. The UMFPACK serial direct solver is used to solve the linearized problem within Newton's method.

#### 3.7.3.3 Results and Discussion

Figures 3.11(a) and 3.11(a) show the solutions as computed by the standard Galerkin formulation and the Galerkin gradient least-squares formulation for the coarse mesh. It is apparent from these figures that the standard Galerkin method undershoots the solution tremendously, computing the clearly non-physical temperature of roughly $-1167\ K$ at the row of nodes one element depth in

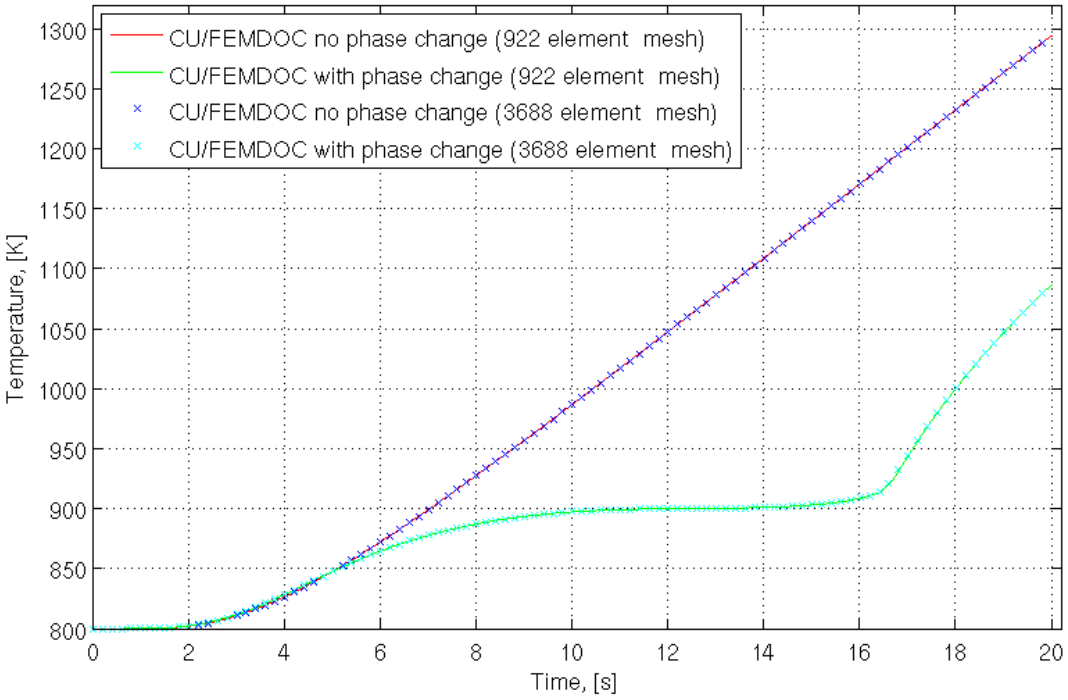



Figure 3.8: Center node temperature time history for both phase changing and non-phase changing simulations.

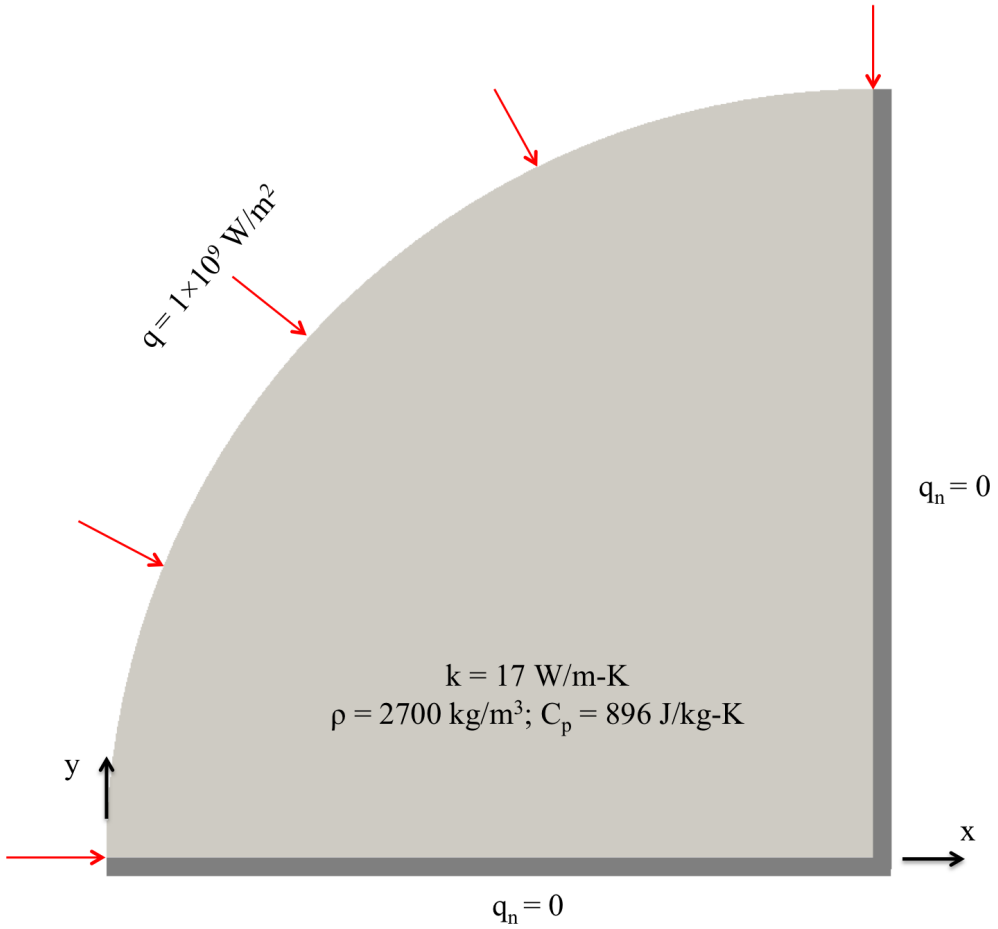
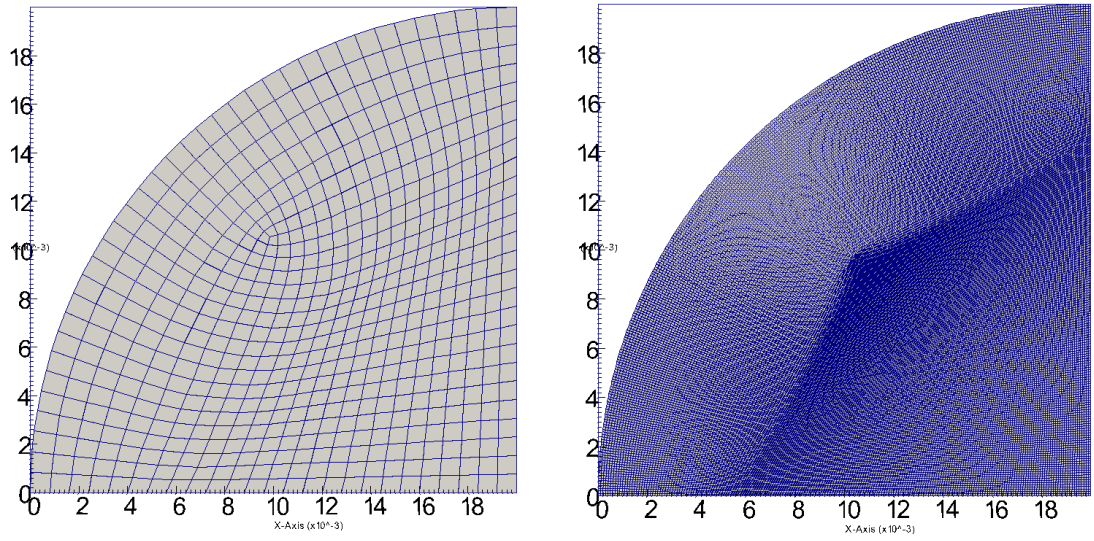



Figure 3.9: Quarter cylinder phase change problem geometry and boundary conditions.

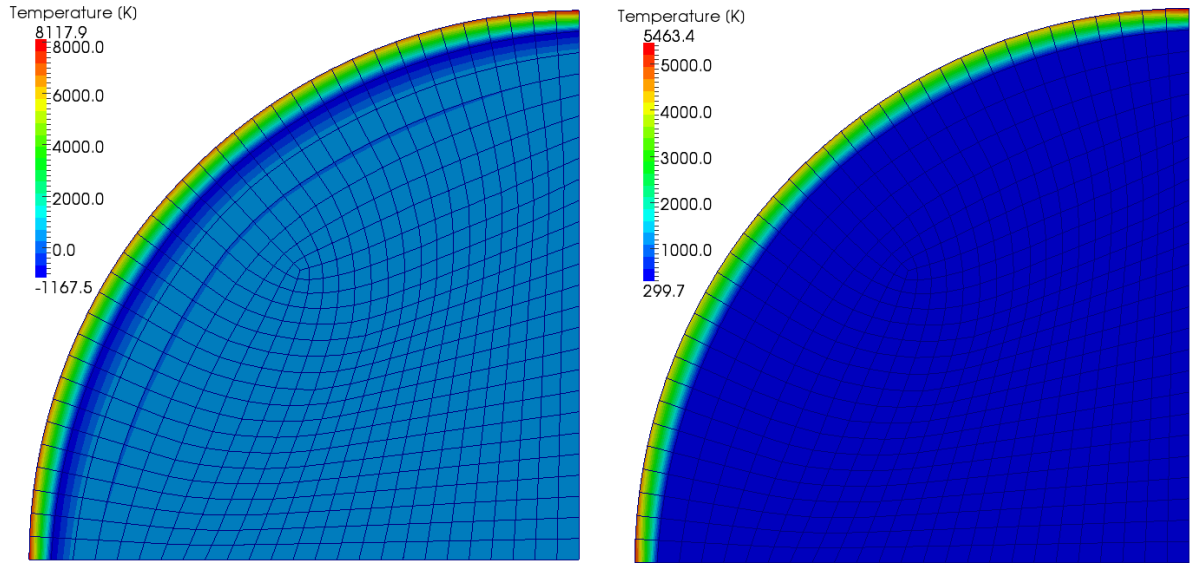



(a) 601 element coarse mesh.

(b) 40,409 element fine mesh.

Figure 3.10: Coarse and fine meshes used for the GGLS example.

from the outer boundary. This undershoot is due to the fact that the elements are too coarse to accurately capture the temperature gradient induced by the high heat flux. Using the GGLS method for the same problem, however, proves that the stabilization term controls the strong gradient and prevents the temperature field from taking on negative values.




(a) Standard Galerkin solution.

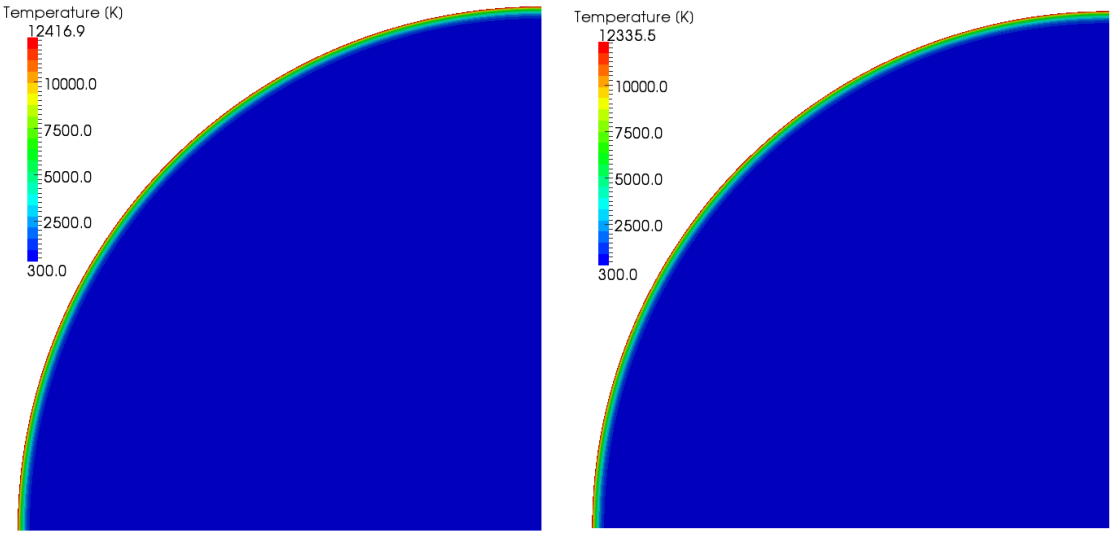
(b) GGLS stabilized solution.

Figure 3.11: Coarse mesh temperature contours computed using a standard Galerkin and the GGLS formulation.

The standard approach to avoiding over/undershoots in the solution when using a Galerkin formulation for heat transfer is to refine the mesh. The next two figures show that by refining the mesh the Galerkin solution no longer under-predicts the solution and the GGLS solution is almost exactly identical to Galerkin result. With this observation we conclude that it is possible to avoid oscillatory solutions with coarser meshes via the GGLS formulation.

### 3.7.4 Axisymmetric Q* Ablation

This problem tests the Q* ablation formulation presented earlier in this chapter and was inspired the problem presented by Hogge [57].




(a) Standard Galerkin solution.

(b) GGLS stabilization solution.

Figure 3.12: Fine mesh temperature contour computed using a standard Galerkin and the GGLS formulation

#### 3.7.4.1 Problem Description

Figure 3.13 shows the geometry and boundary conditions for this problem. The geometry is a half-sphere modeled by an axisymmetric quarter-circle of radius 0.05 $m$ and material properties of $\rho = 1925\ kg/m^3$, $C_p = 2200\ J/(kg \cdot K)$ and $k = 25.0\ W/(m \cdot K)$. The material begins to ablate at 3800 $K$ with a latent heat release of $1.925 \times 10^7\ J/kg \cdot K$. A convective heat flux is applied to the outer surface of the sphere with a convection coefficient that varies continuously between $5,000$ and $15,000\ W/m^2 \cdot K$ with an ambient temperature of $T_a = 5000\ K$. All other edges are adiabatic (i.e. $\bar{q} = 0$), and the initial temperature is 300 $K$ .

#### 3.7.4.2 Computational Mesh and Model

This problem is solved using three meshes with varying refinement in the circumferential direction. A 30x5, 30x10, and 30x20 quadrilateral element mesh are employed to assess the quality of the ablating surface recession as the level of refinement at the surface can affect the smoothness of the ablation response. A $\Theta$-scheme time integrator is used with $\Theta = 1.0$ and a constant time step of $\Delta\ t = 0.1\ s$ is used for 200 time iterations. The nonlinear Newton solver has an initial relaxation of 1.0. If the first Newton solve fails to converge, the Newton solve begins again with a relaxation of 2/3. This has proven to be an efficient and effective strategy for solving ablation type problems.

Figure 3.14 shows the 30x5 element mesh. Five elements in the circumferential direction is clearly a coarse approximation of the sphere's curved boundary. Refinements in this direction are used to yield an ablation response with better resolution.

#### 3.7.4.3 Results and Discussion

Figure 3.15 shows the temperature contours and final ablated shape at the 20.0 $s$ simulation end time. It is noted that the coarsest mesh roughly approximates the ablation boundary but its final shape is very similar in nature to the more refined meshes. Table 3.1 shows surface temperature at the stagnation point at the end of the simulation and the percentage it deviates from the 3800 $K$

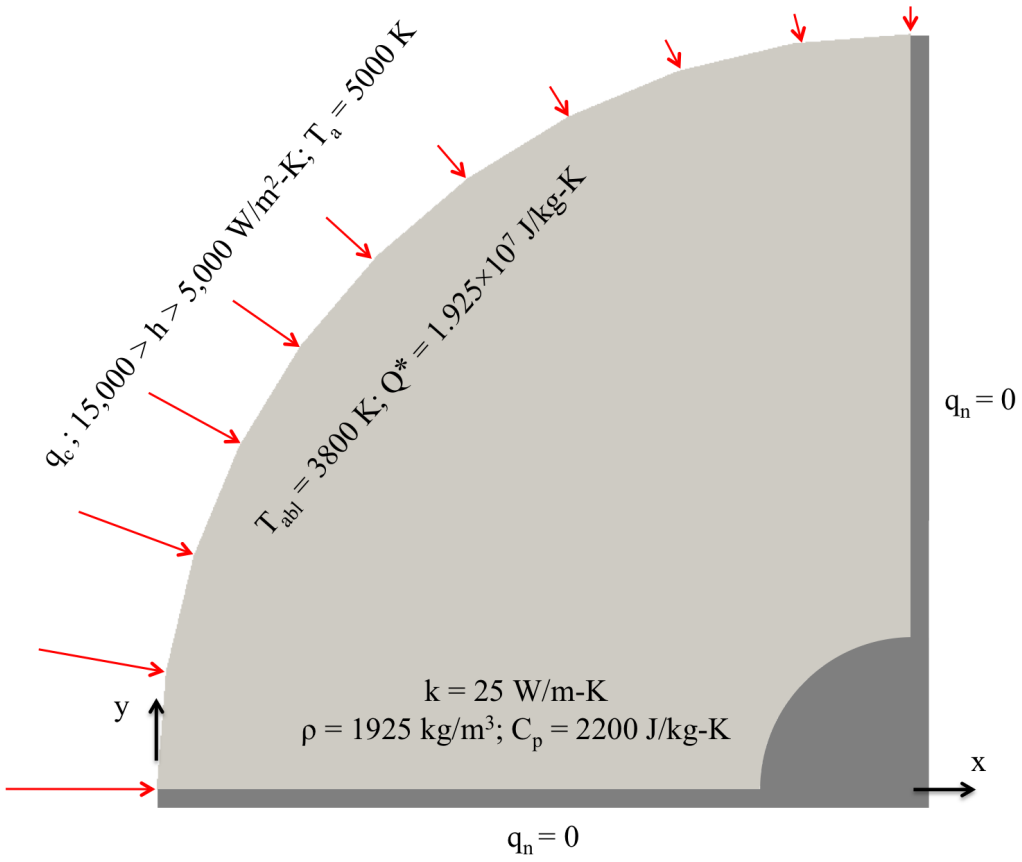



Figure 3.13: Axisymmetric sphere ablation geometry and boundary conditions.

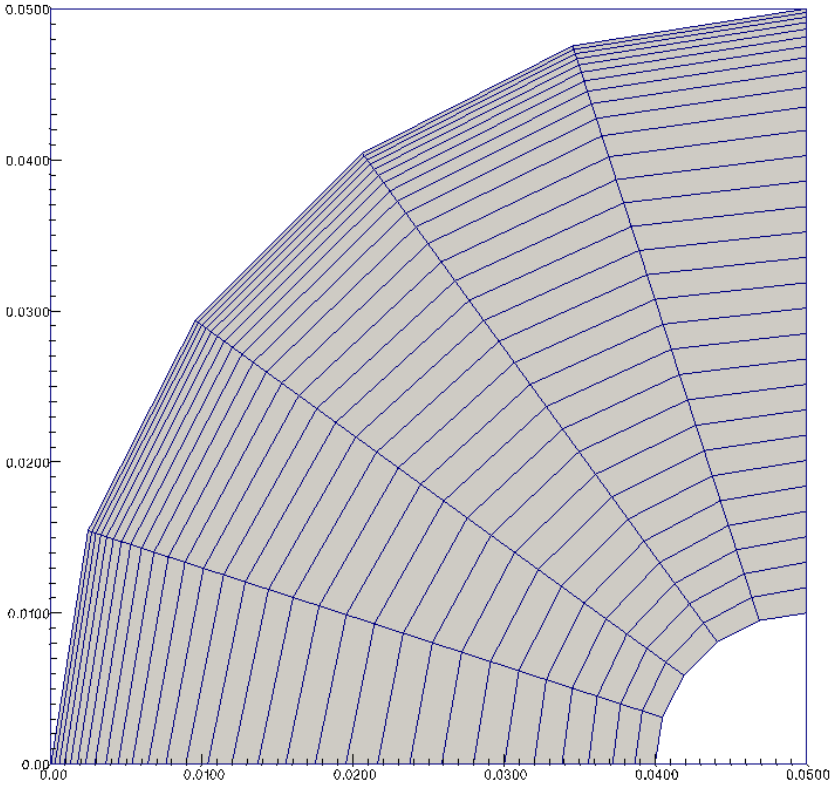
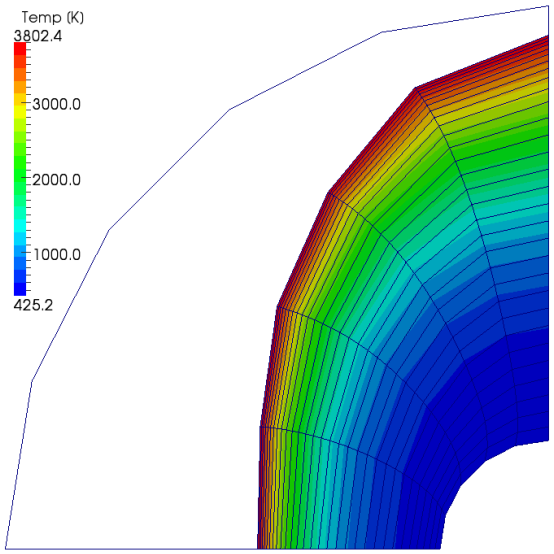
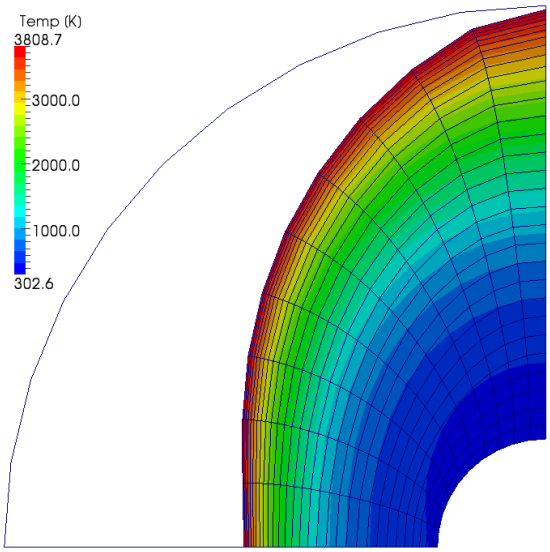



Figure 3.14: Axisymmetric sphere ablation 30x5 quadrilateral element mesh.

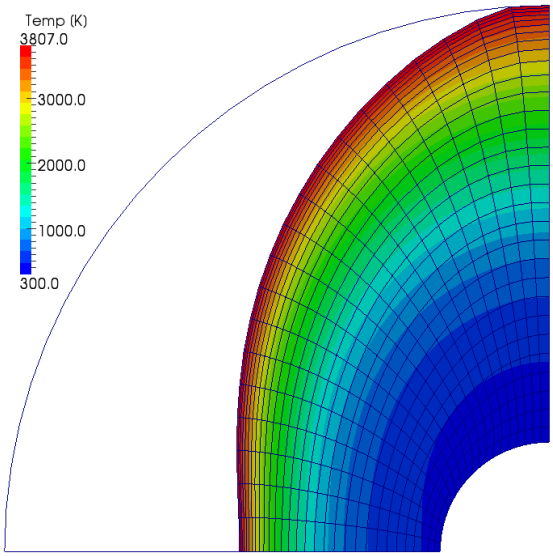
ablation temperature. From this we conclude that the penalty formulation used to enforce the ablation temperature at the boundary does an adequate job of holding this value.




(a) 30x5 element mesh.




(b) 30x10 element mesh.




(c) 30x20 element mesh.

Figure 3.15: Axisymmetric sphere temperature contours and ablated shape at 20.0 $s$.

| **Mesh** | Surf. Temp. | % Diff from 3800 K | Ablation Depth |
|---|---|---|---|
| 30x5 element mesh | 3802.4 K | 0.06 % | 0.0232 m |
| 30x10 element mesh | 3800.0 K | 0.00 % | 0.0221 m |
| 30x20 element mesh | 3801.5 K | 0.04 % | 0.0216 m |

Table 3.1: Axisymmetric sphere surface temperatures and ablation depth at 20.0 $s$.

# Chapter 4

# Computational Structural Dynamics

## 4.1 Introduction

This chapter's primary purpose is to overview the elastodynamic equations, its solution via finite element methods, and the generalized-$\alpha$ time integrator used to solve its semi-discrete form. A tightly coupled thermoelastic formulation will be presented that allows the solution of the elastodynamic equations in conjunction with the transient heat equation. The developments of this chapter will enable us to simultaneously solve the Navier-Stokes equations with the thermoelastodynamic equations via the coupling schemes discussed in the next chapter.

## 4.2 Elastodynamic Equations

The elastodynamic equations govern the time-dependent response of a solid body subject to external forces, and represents conservation of linear momentum. The response of a solid body can be grouped into the following three categories based on the nature of the deformations. (1) A small displacement, small strain problem assumes the undeformed and deformed configurations of a body are identical and the material exhibits linear behavior. Hence, the geometric response of the structure is assumed to be linear and the material response is assumed to be linear. (2) A large displacement, small strain problem assumes the undeformed and deformed configurations of a body may be significantly different, however the strains remain small and behave linearly with the material. The stiffness of the structure is dependent on the displacement field, and this type of problem is categorized as geometrically nonlinear, but materially linear. Problems falling under this

category typically involve long flexible structures. (3) A large displacement, large strain problem assumes both significant displacements and strains. This type of problem is not only geometrically nonlinear, but also materially nonlinear. Such problems requiring this type of analysis are hyper-elastic materials such as elastomers or the elastoplastic response of a metallic structure. For the purposes of this thesis, only small displacement, small strain structural dynamics problems with linear elastic material relationships will be considered. However, it is important to note that framework developed herein for solving aerothermoelastic problems may easily be extended to problems with nonlinear geometric or material behaviors.

### 4.2.1 Differential Form of the Elastodynamic Equation

The elastodynamic equations may be expressed as

$$\rho\frac{\partial^2 u_j}{\partial t^2} + \phi(u_j)\frac{\partial u_j}{\partial t} - \frac{\partial \sigma_{ij}}{\partial x_j} - b_j = \mathbf{0} \tag{4.1}$$

where $u_j$ are displacement degrees-of-freedom, $\phi(u_j)$ is a damping coefficient, $\sigma_{ij}$ is the Cauchy stress tensor, and $b_j$ are volumetric or body forces. Here we express the damping coefficient as a function of the displacements alone; in general, however, the damping coefficient may depend on the displacements, velocities, and even accelerations.

Equation 4.1 is completed by the Dirichlet boundary conditions

$$u_j = \bar{u}_j(\boldsymbol{x}, t) \quad \text{on} \quad \Gamma_u \tag{4.2}$$

which prescribe the displacements $\bar{u}_j$ on boundary $\Gamma_u$ and the Neumann boundary conditions

$$\sigma_{ij} \cdot \hat{n}_i = \bar{t}_j(\boldsymbol{x}, t) \quad \text{on} \quad \Gamma_{\bar{t}} \tag{4.3}$$

which specifies the traction $\bar{t}_j$ on the boundary $\Gamma_{\bar{t}}$. Note that when a traction is integrated over a surface it results in a quantity with units of force.

Initial conditions must be specified for dynamic problems. Since the elastodynamic equation is second-order in time, the initial state (the displacements $u_j$) and its first-order time derivatives

(the velocities $\dot{u}_j$) must be known at the initial time to properly characterize the initial value problem. Hence, the initial conditions are expressed as

$$u_j(x_i, t = 0) = u_j^0(x_i) \quad \text{in} \quad \Omega_s \tag{4.4}$$

$$\dot{u}_j(x_i, t = 0) = \dot{u}_j^0(x_i) \quad \text{in} \quad \Omega_s \tag{4.5}$$

### 4.2.2 Elastic Material Models

The behavior of a solid material is governed by a constitutive relation which relates stresses to strains. In this thesis we consider only linear elasticity, for which the stress-strain relation may be represented by the equation

$$\sigma_{ij} = D_{ijkl}\left(\epsilon_{kl}^s - \epsilon_{kl}^t\right) \tag{4.6}$$

where $\epsilon_{kl}^s$ is an elastic strain tensor and is defined as

$$\epsilon_{kl}^s = \frac{1}{2}\left(\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k}\right) \tag{4.7}$$

and $\epsilon_{kl}^t$ is a thermal strain tensor arising from thermal expansion effects in the material

$$\epsilon_{kl}^t = \alpha_{kl}[T - T_{ref}] \tag{4.8}$$

## 4.3 Finite Element Discretization of the Elastodynamic Equation

This section discusses the spatial discretization of the elastodynamic equation via a standard Galerkin method.

### 4.3.1 Galerkin Discretization

The weak form of the elastodynamic equation is obtained by weighting it with arbitrary test functions and integrating of over the domain. By invoking the principal of virtual work, the arbitrary test functions are virtual displacements $\delta u_j$ and we obtain the following variational form

$$\int_{\Omega_s} \delta u_j \cdot \left[\rho \frac{\partial^2 u_j}{\partial t^2} + \phi(u_j)\frac{\partial u_j}{\partial t} - \frac{\partial \sigma_{ij}}{\partial x_j} - b_j\right] d\Omega = 0 \tag{4.9}$$

Integrating by parts the stress divergence term leads to the following statement

$$\int_{\Omega_s} \left[ \delta u_j \cdot \left( \rho \frac{\partial^2 u_j}{\partial t^2} + \phi(u_j) \frac{\partial u_j}{\partial t} - b_j \right) + \frac{\partial \delta u_j}{\partial x_i} \cdot \sigma_{ij} \right] d\Omega - \int_{\Gamma_s} \delta u_j \cdot \sigma_{ij} n_i \, d\Gamma = 0 \tag{4.10}$$

where $\partial \delta u_j / \partial x_i$ is the virtual strain term and is defined as follows

$$\frac{\partial \delta u_j}{\partial x_i} = \delta \epsilon_{ij} = \frac{1}{2} \left( \frac{\partial \delta u_i}{\partial x_j} + \frac{\partial \delta u_j}{\partial x_i} \right) \tag{4.11}$$

### 4.3.2 Finite Element Spatial Approximation

The solution of equation 4.10 is obtained by first discretizing the domain $\Omega$ into elements $\Omega^e$ and solving for the approximate displacement solutions $u_j^h$ of the weak statement.

The approximate trial solutions $\boldsymbol{u}^h$ are sought by solving the weak statement represented by a linear combination of the test functions $\boldsymbol{W}^h$. The trial solution and test function spaces are defined in a similar manner to equations 2.43 and 2.44 from the previous chapter.

$$\mathcal{S}^h = \left[ \boldsymbol{u}^h \in \left[ C^0(\Omega) \right]^{n_{dof}} \;,\; \boldsymbol{u}^h \mid_{\Omega^e} \in \left[ P^k(\Omega^e) \right]^{n_{dof}} \;,\; \boldsymbol{u}^h = \bar{u} \text{ on } \Gamma_u \right] \tag{4.12}$$

which states that the trial solution must be $C^0$ continuous, representable by an interpolation polynomial $P^k$ of order $k$, and must satisfy the Dirichlet conditions on the boundary. Similarly, the test function space is defined by

$$\mathcal{V}^h = \left[ \boldsymbol{W}^h \in \left[ C^0(\Omega) \right]^{n_{dof}} \;,\; \boldsymbol{W}^h \mid_{\Omega^e} \in \left[ P^k(\Omega^e) \right]^{n_{dof}} \;,\; \boldsymbol{W}^h = 0 \text{ on } \Gamma_u \right] \tag{4.13}$$

The standard approximation for $u_j$ is represented using element basis functions via a split in the time and spatial dependence of the variable according to the equation

$$u_j^h(\boldsymbol{x}, t) = \sum_{n=1}^{n_n} N^n(\boldsymbol{x}^n) \, u_j^n(t). \tag{4.14}$$

where $n = 1, ..., n_n$ symbolizes the nodes of the element. Likewise the virtual displacements are approximated by the equation

$$\delta u_j^h(\boldsymbol{x}, t) = \sum_{n=1}^{n_n} N^n(x_j^n) \, u_j^n(t). \tag{4.15}$$

The relation between strains and displacements can be expressed using the matrix-vector notation as follows

$$\boldsymbol{\epsilon} = \mathbf{B}\,\boldsymbol{u} \tag{4.16}$$

where $\mathbf{B}$ is the so-called strain-displacement operator. Now, substituting the relations for displacement and strain we obtain the following expression

$$\int_{\Omega_s} \rho \boldsymbol{W}^T \boldsymbol{W}\, \ddot{\boldsymbol{u}}\; d\Omega + \int_{\Omega_s} \phi(\boldsymbol{u}) \boldsymbol{W}^T \boldsymbol{W}\, \dot{\boldsymbol{u}}\; d\Omega - \int_{\Omega_s} \mathbf{W}^T \boldsymbol{\sigma}\; d\Omega - \int_{\Omega_s} \boldsymbol{W}^T \boldsymbol{b}\; d\Omega - \int_{\Gamma_s} \boldsymbol{W}^T \bar{\boldsymbol{t}}\; d\Gamma = 0 \tag{4.17}$$

Summing all finite elements we obtain a semi-discrete, second-order ordinary differential equation of the following form

$$\boldsymbol{\mathfrak{R}}_s\left(\ddot{\boldsymbol{u}}, \dot{\boldsymbol{u}}, \boldsymbol{u}\right) \equiv \mathbf{M}\,\ddot{\boldsymbol{u}} + \mathbf{C}\left(\boldsymbol{u}\right)\,\dot{\boldsymbol{u}} + \boldsymbol{R}\left(\boldsymbol{\sigma}\right) - \boldsymbol{F}\left(t\right) = \mathbf{0} \tag{4.18}$$

where the global mass matrix, damping matrix, static residual vector, and force vector are defined, respectively, as

$$\mathbf{M} = \sum_{e=1}^{n_e} \mathbf{M}^e \;,\; \mathbf{C}\left(\boldsymbol{u}\right) = \sum_{e=1}^{n_e} \mathbf{C}^e\left(\boldsymbol{u}\right) \;,\; \boldsymbol{R}\left(\boldsymbol{\sigma}\right) = \sum_{e=1}^{n_e} \boldsymbol{R}^e\left(\boldsymbol{\sigma}\right) \;,\; \boldsymbol{F}\left(t\right) = \sum_{e=1}^{n_e} \boldsymbol{F}^e\left(t\right) \tag{4.19}$$

The global matrices and vectors defined by equation 4.19 are assembled from element level contributions which are given by

$$\mathbf{M}^e = \int_{\Omega_s^e} \rho \boldsymbol{W}^T \boldsymbol{W}\; d\Omega \tag{4.20}$$

$$\mathbf{C}^e\left(\boldsymbol{u}\right) = \int_{\Omega_s^e} \phi\left(\boldsymbol{u}\right) \boldsymbol{W}^T \boldsymbol{W}\; d\Omega \tag{4.21}$$

$$\boldsymbol{R}^e\left(\boldsymbol{\sigma}\right) = \int_{\Omega_s^e} \mathbf{B}^T \boldsymbol{\sigma}\; d\Omega \tag{4.22}$$

$$\boldsymbol{F}^e\left(t\right) = \int_{\Omega_s^e} \boldsymbol{W}^T \boldsymbol{b}\left(t\right)\; d\Omega - \int_{\Gamma_s^e} \boldsymbol{W}^T \bar{\boldsymbol{t}}\left(t\right)\; d\Gamma \tag{4.23}$$

An alternative expression for equation 4.18 is sometimes given by recognizing that $\boldsymbol{\sigma} = \boldsymbol{\sigma}\left(\boldsymbol{\epsilon}\right)$ and $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}\left(\boldsymbol{u}\right)$, so then $\boldsymbol{\sigma} = \boldsymbol{\sigma}\left(\boldsymbol{u}\right)$ and equation 4.18 may then be expressed as

$$\boldsymbol{R}_s\left(\ddot{\boldsymbol{u}}, \dot{\boldsymbol{u}}, \dot{\boldsymbol{u}}\right) \equiv \mathbf{M}\ddot{\boldsymbol{u}} + \mathbf{C}\left(\boldsymbol{u}\right)\,\dot{\boldsymbol{u}} + \boldsymbol{R}_s\left(\boldsymbol{u}\right) - \boldsymbol{F}\left(t\right) = \mathbf{0} \tag{4.24}$$

### 4.3.3 Damping Matrix

Determining an appropriate damping coefficient $\phi$ and damping term $\boldsymbol{C}(\boldsymbol{u})$ is often difficult, if not impossible for most problems. A common approach is to approximate the damping matrix using the Rayleigh damping method. In this case,

$$\mathbf{C}^e(\boldsymbol{u}) = \alpha_d \mathbf{M}^e + \beta_d \frac{\partial \boldsymbol{R}(\boldsymbol{u})}{\partial \boldsymbol{u}} \tag{4.25}$$

where the last term in the previous equation is the element tangent stiffness matrix.

## 4.4 Solution Strategies for the Elastodynamic Equation

The dynamic response of a structural system can be characterized by any number of methods, including mode superposition, direct time integration, or modal analysis. However, for general analysis of structural dynamics problems direct time integration is the most common choice.

The Newmark method [109] is arguably the most predominant direct time integration scheme for second-order ordinary differential equations arising from the elastodynamic equations. Several improvements to the basic Newmark algorithm have been introduced since it inception, including the Hilber, Hughes, & Taylor[54] (HHT-$\alpha$) method and the Wood, Bossak, & Zienkiewicz [143] (WBZ-$\alpha$) method. The generalized-$\alpha$ method was introduced by Chung and Hulbert [30] in 1993 and is essentially a generalization of the three aforementioned schemes. The generalized-$\alpha$ method is briefly overviewed here and used for several numerical examples shown during the remainder of this document.

### 4.4.1 Time Integration via Generalized-$\alpha$ Method

Beginning from the semi-discrete form of the residual equation (4.24), repeated here for convenience,

$$\boldsymbol{\mathfrak{R}}_s(\ddot{\boldsymbol{u}}, \dot{\boldsymbol{u}}, \boldsymbol{u}) \equiv \mathbf{M}\ddot{\boldsymbol{u}} + \mathbf{C}(\boldsymbol{u})\ \dot{\boldsymbol{u}} + \boldsymbol{R}(\boldsymbol{u}) - \boldsymbol{F}(t) = \mathbf{0} \tag{4.26}$$

the generalized-$\alpha$ method solves the equilibrium equation at the intermediate time state $n+1-\alpha$ according to the equation

$$\mathfrak{R}_s \equiv \mathbf{M}\ddot{\boldsymbol{u}}^{n+1-\alpha_m} + \mathbf{C}\left(\boldsymbol{u}^{n+1-\alpha_f}\right)\dot{\boldsymbol{u}}^{n+1-\alpha_f} + \boldsymbol{R}\left(\boldsymbol{u}^{n+1-\alpha_f}\right) - \boldsymbol{F}\left(t^{n+1-\alpha_f}\right) = \mathbf{0} \tag{4.27}$$

and the intermediate states are defined by the equations

$$\ddot{\boldsymbol{u}}^{n+1-\alpha_m} = (1-\alpha_m)\,\ddot{\boldsymbol{u}}^{n+1} + \alpha_m \ddot{\boldsymbol{u}}^{n} \tag{4.28}$$

$$\dot{\boldsymbol{u}}^{n+1-\alpha_f} = (1-\alpha_f)\,\dot{\boldsymbol{u}}^{n+1} + \alpha_f \dot{\boldsymbol{u}}^{n} \tag{4.29}$$

$$\boldsymbol{u}^{n+1-\alpha_f} = (1-\alpha_f)\,\boldsymbol{u}^{n+1} + \alpha_f \boldsymbol{u}^{n} \tag{4.30}$$

$$t^{n+1-\alpha_f} = (1-\alpha_f)\,t^{n+1} + \alpha_f t^{n} \tag{4.31}$$

where $\alpha_m$ and $\alpha_f$ are algorithmic parameters that interpolate between the states at $n$ and $n+1$.

The Newmark approximations for $\ddot{\boldsymbol{u}}_{n+1}$ and $\dot{\boldsymbol{u}}_{n+1}$ are expressed by

$$\ddot{\boldsymbol{u}}^{n+1} = \left(1 - \tfrac{1}{2\beta}\right)\ddot{\boldsymbol{u}}^{n} - \tfrac{1}{\beta\Delta t}\dot{\boldsymbol{u}}^{n} + \tfrac{1}{\beta\Delta t^2}\left(\boldsymbol{u}^{n+1} - \boldsymbol{u}^{n}\right) \tag{4.32}$$

$$\dot{\boldsymbol{u}}^{n+1} = \dot{\boldsymbol{u}}^{n} + \Delta t\,(1-\gamma)\,\ddot{\boldsymbol{u}}^{n} + \Delta t\gamma\ddot{\boldsymbol{u}}^{n+1} \tag{4.33}$$

where $\beta$ and $\gamma$ are Newmark algorithmic parameters. The approximated accelerations (4.32) and velocities (4.33) are based on the current degree-of-freedom solution $\boldsymbol{u}^{n+1}$ and previously known displacements, velocities, and accelerations. The terms are computed and substituted into the generalized-$\alpha$ equations for acceleration (4.28) and velocity (4.29).

According to Chung and Hulbert's analysis [30] the generalized-$\alpha$ method is second-order accurate when

$$\gamma = \frac{1}{2} - \alpha_m + \alpha_f \tag{4.34}$$

and unconditionally stable (for linear problems) when

$$\alpha_m \leq \alpha_f \leq \tfrac{1}{2} \quad , \quad \beta \geq \tfrac{1}{4} + \tfrac{1}{2}\left(\alpha_f - \alpha_m\right) \tag{4.35}$$

### 4.4.2 Nonlinear Solution via Newton's Method

In order to solve the following nonlinear, second-order-in-time residual equation,

$$\mathfrak{R}_s\left(\ddot{\boldsymbol{u}}^{n+1-\alpha_m}, \dot{\boldsymbol{u}}^{n+1-\alpha_f}, \boldsymbol{u}^{n+1-\alpha_f}\right) = \mathbf{M}\ddot{\boldsymbol{u}}^{n+1-\alpha_m} + \mathbf{C}\left(\boldsymbol{u}^{n+1-\alpha_f}\right)\dot{\boldsymbol{u}}^{n+1-\alpha_f} + \boldsymbol{R}_s^{n+1-\alpha_f} - \boldsymbol{F}^{n+1-\alpha_f} = \mathbf{0} \quad (4.36)$$

Newton's method is used to linearize the problem and compute an increment to the solution. First, however, the generalized mid-point rule is used to interpret the nonlinear static residual $\boldsymbol{R}_s^{n+1-\alpha_f}$ and external force vector $\boldsymbol{F}^{n+1-\alpha_f}$ via

$$\boldsymbol{R}_s^{n+1-\alpha_f} = \boldsymbol{R}_s\left((1-\alpha_f)\boldsymbol{u}^{n+1} + \alpha_f\,\boldsymbol{u}^n\right) \quad (4.37)$$

$$\boldsymbol{F}^{n+1-\alpha_f} = \boldsymbol{F}\left((1-\alpha_f)t^{n+1} + \alpha_f\,t^n\right)\ . \quad (4.38)$$

The linearization of the total residual, which is expressed in terms of a state at $n+1-\alpha$, takes place about the state $n+1$ as follows

$$\mathfrak{R}_s\left(\boldsymbol{u}^{(m),n+1-\alpha}\right) + \underbrace{\left[\frac{\partial \mathfrak{R}_s\left(\boldsymbol{u}^{(m),n+1-\alpha}\right)}{\partial \boldsymbol{u}^{n+1}}\right]}_{\mathfrak{J}_s} \Delta\boldsymbol{u}^{(m),n+1-\alpha} = \mathbf{0} \quad (4.39)$$

where $(\cdot)^{(m),n+1-\alpha}$ represents the linearization at the $m$-th nonlinear iteration and the time state $n+1-\alpha$.

The Jacobian matrix $\mathfrak{J}_s$ is then defined by

$$\mathfrak{J}_s = \mathbf{M}\frac{\partial \ddot{\boldsymbol{u}}^{n+1-\alpha_m}}{\partial \boldsymbol{u}^{n+1}} + \mathbf{C}^{n+1-\alpha_f}\frac{\partial \dot{\boldsymbol{u}}^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}} + \frac{\partial \mathbf{C}^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}}\dot{\boldsymbol{u}}^{n+1-\alpha_f} + \frac{\partial \boldsymbol{R}_s^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}} - \frac{\partial \boldsymbol{F}^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}} \quad (4.40)$$

where the derivatives of $\boldsymbol{R}^{n+1-\alpha_f}$ and $\boldsymbol{F}^{n+1-\alpha_f}$ are

$$\frac{\partial \boldsymbol{R}_s^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}} = (1-\alpha_f)\,\frac{\partial \boldsymbol{R}\left(\boldsymbol{u}^{n+1}\right)}{\partial \boldsymbol{u}^{n+1}} \quad (4.41)$$

$$\frac{\partial \boldsymbol{F}^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}} = \mathbf{0} \quad (4.42)$$

The derivative of the damping matrix $\mathbf{C}^{n+1-\alpha_f}$ with respect to $\boldsymbol{u}_{n+1}$ must be accounted for according to the damping model used (e.g. Rayleigh damping). Finally, the derivatives of $\ddot{\boldsymbol{u}}^{n+1-\alpha_m}$

and $\dot{\boldsymbol{u}}^{n+1-\alpha_f}$ with respect to $\boldsymbol{u}^{n+1}$ are expressed as follows

$$\frac{\partial \ddot{\boldsymbol{u}}^{n+1-\alpha_m}}{\partial \boldsymbol{u}^{n+1}} = \frac{1-\alpha_m}{\beta \Delta t^2} \tag{4.43}$$

$$\frac{\partial \dot{\boldsymbol{u}}^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}} = \frac{(1-\alpha_f)\,\gamma}{\beta \Delta t} \tag{4.44}$$

Thus, the Jacobian matrix as shown in equation 4.40 can be expressed in the following form for the generalized-$\alpha$ method

$$\mathfrak{J}_s = \left(\frac{1-\alpha_m}{\beta \Delta t^2}\right)\mathbf{M} + \left(\frac{(1-\alpha_f)\,\gamma}{\beta \Delta t}\right)\mathbf{C}^{n+1-\alpha_f} + \frac{\partial \mathbf{C}^{n+1-\alpha_f}}{\partial \boldsymbol{u}^{n+1}}\dot{\boldsymbol{u}}^{n+1-\alpha_f} + (1-\alpha_f)\,\frac{\partial \boldsymbol{R}_s\left(\boldsymbol{u}_{n+1}\right)}{\partial \boldsymbol{u}^{n+1}} \tag{4.45}$$

## 4.5 Thermoelastic Coupling

A coupled thermoelastodynamic system can be assembled by piecing together the elastodynamic equations and the transient heat equation in the following fashion

$$\begin{bmatrix} \mathbf{M}_{uu} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{u}} \\ \ddot{\boldsymbol{T}} \end{bmatrix} + \begin{bmatrix} \mathbf{C}_{uu} & \mathbf{0} \\ \mathbf{C}_{Tu} & \mathbf{C}_{TT} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{u}} \\ \dot{\boldsymbol{T}} \end{bmatrix} + \begin{bmatrix} \mathbf{K}_{uu} & \mathbf{K}_{uT} \\ \mathbf{0} & \mathbf{K}_{TT} \end{bmatrix} \begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{T} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{Q} \end{bmatrix} \tag{4.46}$$

The sub-matrices located on the diagonal of the above matrix equation have been previously defined, but are repeated here for the sake of convenience. The element-level structural mass, stiffness, and damping matrices are defined as, respectively,

$$\boldsymbol{M}^e_{uu} = \int_{\Omega^e} \boldsymbol{W}^T \rho \boldsymbol{W}\; d\Omega \tag{4.47}$$

$$\boldsymbol{K}^e_{uu} = \int_{\Omega^e} \boldsymbol{B}^T \boldsymbol{\sigma}\; d\Omega = \int_{\Omega^e} \boldsymbol{B}^T \boldsymbol{D}_s \boldsymbol{B}\; d\Omega \tag{4.48}$$

$$\boldsymbol{C}^e_{uu} = \int_{\Omega^e} \phi\left(\boldsymbol{u}\right) \boldsymbol{W}^T \rho \boldsymbol{W}\; d\Omega \tag{4.49}$$

where $\boldsymbol{D}_s$ is the structural constitutive matrix that relates stress and strain via $\boldsymbol{\sigma} = \boldsymbol{D}_s\,\boldsymbol{\epsilon}$. The thermal capacitance and conduction matrices are defined as

$$\boldsymbol{C}^e_{TT} = \int_{\Omega^e} \rho C_p \boldsymbol{W}^T \boldsymbol{W}\; d\Omega \tag{4.50}$$

$$\boldsymbol{K}^e_{TT} = \int_{\Omega^e} \boldsymbol{B}^T \boldsymbol{D}_t \boldsymbol{B}\; d\Omega \tag{4.51}$$

The matrix $\boldsymbol{K}_{uT}$ couples displacements in the structure and the temperature field through the coefficient of thermal expansion. The matrix contribution takes the following form

$$\boldsymbol{K}^e_{uT} = \int_{\Omega^e} \boldsymbol{B}^T \boldsymbol{D}_s \boldsymbol{\alpha} \boldsymbol{N} \; d\Omega \; . \tag{4.52}$$

where $\boldsymbol{\alpha}$ is the vector of coefficients of thermal expansion. The matrix $\boldsymbol{C}^{Tu}$ is often referred to as the "thermoelastic damping" matrix and is defined as

$$\boldsymbol{C}^e_{Tu} = -T_{ref} \left[\boldsymbol{K}^e_{uT}\right]^T \; . \tag{4.53}$$

and $T_{ref}$ is the coefficient of thermal expansion reference temperature (i.e. temperature at which thermal expansion is zero).

## 4.6 Numerical Example Problems

### 4.6.1 2-Dimensional Thermoelastic Dynamic Beam

This problem demonstrates the coupled thermoelastic formulation presented in the previous section for computing the coupled heat transfer and elastic response of a cantilevered beam.

#### 4.6.1.1 Problem Description

The geometry of the beam is shown in Figure 4.1 and consists of a $1.0 \times 0.05$ $m$ rectangular beam with unit thickness. The beam is fully clamped on its left edge and all boundaries are adiabatic except the top surface. The beam is initially at rest and at a uniform temperature of $288.0$ $K$.

The top surface of the beam is heated by a convective heat flux whose heat transfer coefficient take a step function profile. The heat transfer coefficient is $h = 1.0 \times 10^5$ $W/m^2 \cdot K$ from $0 \leq t \leq 0.1$ $s$ and $h = 0.0$ $W/m^2 \cdot K$ for $t > 0.1$ $s$. The thermal conductivity of the material is $k = 300.0$ $W/m \cdot K$, the density is $\rho = 7833.0$ $kg/m^3$, and the thermal capacitance is $C_p = 465.0$ $J/kg \cdot K$.

The elastic modulus of the beam is $E = 3.0 \times 10^{10}$ $N/m^2$, and the Poisson ratio is $\nu = 0.33$. The top half of the beam has a coefficient of thermal expansion (CTE) of $10.0 \times 10^{-6}/K$ and

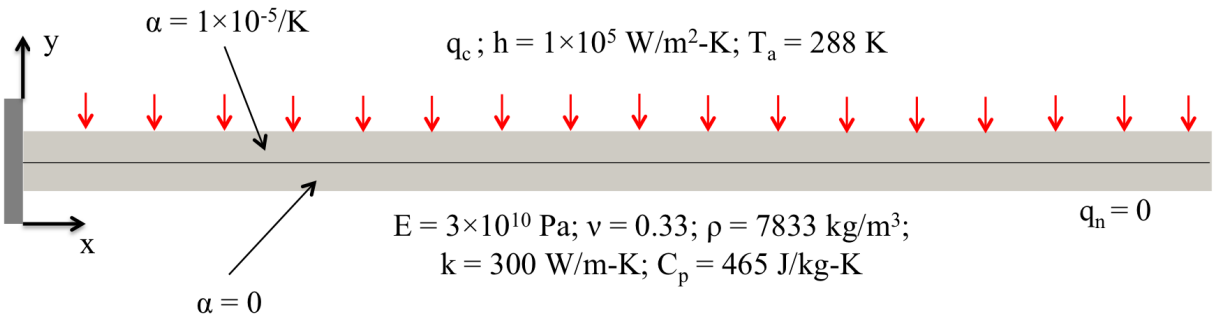



Figure 4.1: Thermoelastic beam geometry and boundary conditions.

a reference temperature of $T_{ref} = 288.0\ K$ while the bottom half has no coeffient of thermal expansion. This mismatch in CTE induces a continuous bending moment for a uniform temperature of the beam that is above or below $T_{ref}$. Proportional damping is used to bring the beam into a stationary resting position after the transient behavior dies out. The proportional damping coefficients used for this problem are: $\alpha_d = 2.0 \times 10^{-3}$ and $\beta_d = 2.0 \times 10^{-4}$.

#### 4.6.1.2 Computational Mesh and Model

The problem is time integrated via the generalized-$\alpha$ integrator presented above. A constant time step of $3.0 \times 10^{-3}\ s$ is used for 2000 time iterations. Three nested meshes are used for the problem with the coarsest mesh being containing a 50x20 grid of quadrilateral elements. This mesh is shown in figure 4.2.

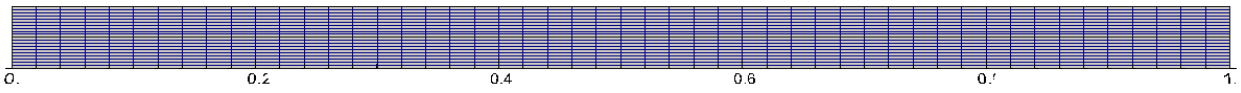



Figure 4.2: Thermoelastic beam 50x20 element computational mesh.

#### 4.6.1.3 Results and Discussion

Figure 4.3 shows the temperature contours for the beam at $t = 0.001\ s$. It is this initial strong temperature gradient in the top layer of the beam that leads to the thermal expansion and bending moment that results in the dynamic response of the beam. Figure 4.4 shows the XX-stress contours at this same instant in time.

Figure 4.5 shows the dynamic y-displacement response of the node at the top right corner of the beam.

Figure 4.5 shows the temperature time history of the top and bottom surfaces of the beam at the tip of the structure.

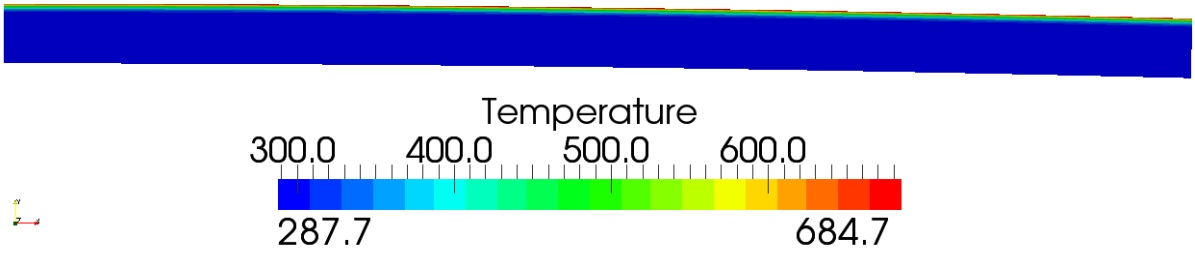



Figure 4.3: Thermoelastic beam temperature contours at t=0.001 s.

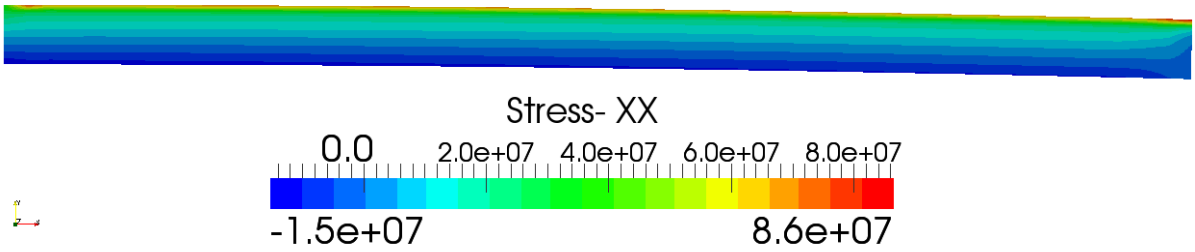



Figure 4.4: Thermoelastic beam XX-stress contours at t=0.001 s.

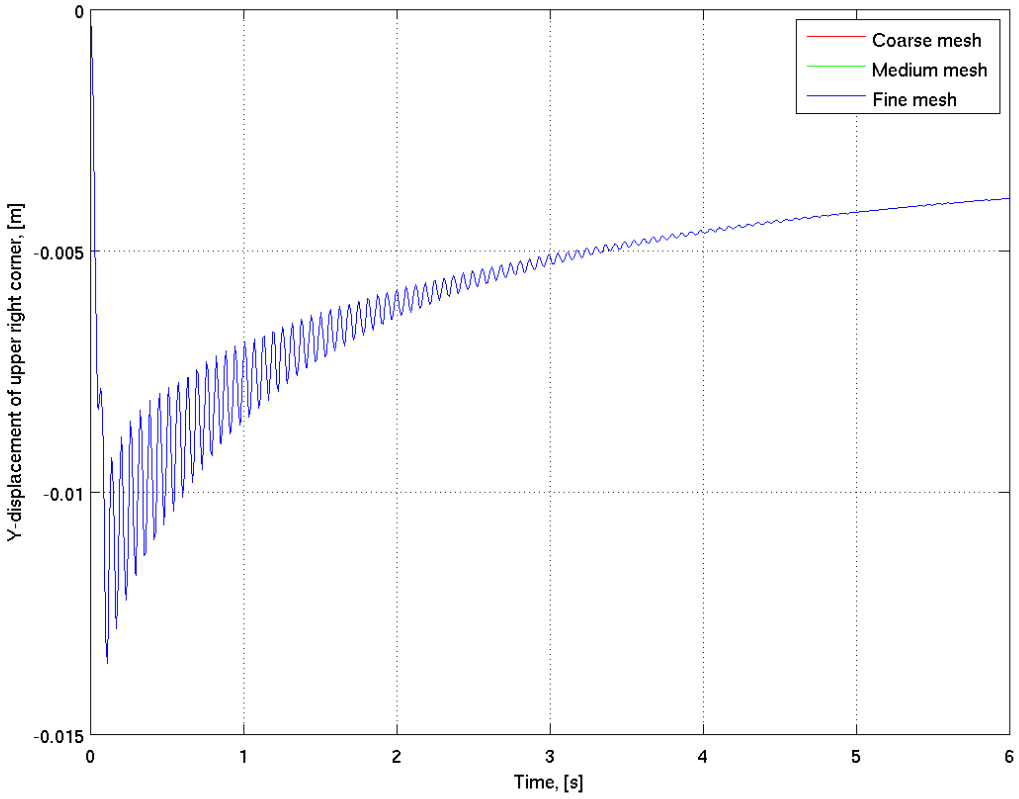



Figure 4.5: Thermoelastic beam y-displacement response for the top right corner node.

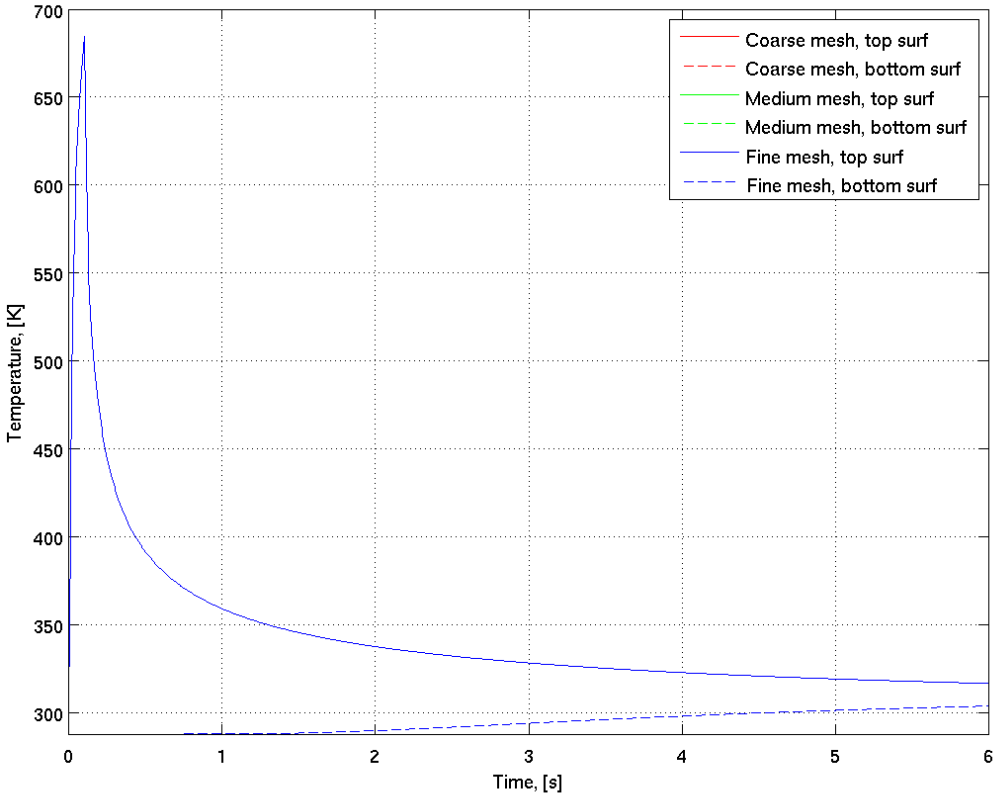



Figure 4.6: Thermoelastic beam top and bottom surface temperature responses.

# Chapter 5

# Coupled Aerothermoelastic Analysis

## 5.1 Introduction

This chapter considers the coupling of the aerodynamic field with the thermal and elastic fields needed to perform a tightly coupled aeroelastic, aerothermal or aerothermoelastic response analysis.

Several forms of coupled solution strategies exist for computational fluid-structure interaction problems. Among them are partitioned solution, staggered solution, and simultaneous or monolithic solution procedures. The partitioned solution approach is discussed by Felippa [45]. As discussed in the introduction, the classical approach to performing a coupled analysis is the staggered coupling method, popularized by research in aeroelasticity [44] and aeroheating [52]. This thesis takes the simultaneous solution approach to fluid-structure coupling whereby the conservation equations of the compressible fluid are solved in conjunction with the energy and/or momentum equations of the solid. This approach of solving a single nonlinear residual equation instead of a sequence of discipline specific sub-problems and passing boundary conditions between them is a potentially promising approach for aeroheating and aerothermoelasticity which is being concurrently pursued by researchers at Sandia National Labs [24].

## 5.2 Aeroelastic Coupling

### 5.2.1 Aeroelastic Interface Conditions

In order for the fluid and solid equations to be coupled across a boundary, the Dirichlet (essential) and Neumman (natural) boundary conditions must be continuous across the interface. That is the jump in values must equal zero. Considering a purely aeroelastic problem for the moment, the Dirichlet conditions that must be satisfied are that the displacement field must be continuous

$$[\![u_j]\!] = u_j^f - u_j^s = 0 \tag{5.1}$$

where the superscripts $f$ and $s$ denote the fluid side and solid side displacements. Equation 5.1 simply states that the displacements at the boundary must be equal. The velocities at the interface must also be continuous

$$[\![\dot{u}_j]\!] = v_j^f - \dot{u}_j^s = 0 \tag{5.2}$$

where $v_j^f$ are the fluid velocity components and $\dot{u}_j^s$ are the solid velocity components. Additionally, the jump in traction across the interface must be satisfied

$$[\![t_j]\!] = -\underbrace{(\tau_{ij} - p\delta_{ij})}_{\sigma_{ij}^f}\hat{n}_i^f - \sigma_{ij}^s\hat{n}_i^s = 0 \tag{5.3}$$

which states that traction vector resulting from the fluid stress tensor must balance the traction vector arising from the solid stress tensor.

### 5.2.2 Aeroelastic Interface Coupling

The procedure for coupling the momentum equations for the fluid and solid begins from the the stabilized weak form of the Navier-Stokes equations and the weak form of the elastodynamic equations. From these starting points we separate out the boundary integral terms we are interested in coupling on the fluid-structure interface.

**Navier-Stokes Momentum Equation Boundary Term**

We start with the stabilized weak form of the Navier-Stokes equations, however with both the

inviscid flux terms $\boldsymbol{F}_i$ and viscous flux terms $\boldsymbol{G}_i$ integrated by parts

$$\begin{aligned}\int_{\Omega_f} \boldsymbol{W}\cdot\left(\frac{\partial \boldsymbol{U}}{\partial t}-\boldsymbol{S}\right)d\Omega+\int_{\Omega_f}\frac{\partial \boldsymbol{W}}{\partial x_i}\cdot\left(\boldsymbol{G}_i-\boldsymbol{F}_i\right)d\Omega+\int_{\Gamma_f}\boldsymbol{W}\cdot\left(\boldsymbol{F}_i-\boldsymbol{G}_i\right)d\Gamma+\\ \sum_{e=1}^{n_e}\int_{\Omega_f^e}\boldsymbol{\mathcal{L}}_{adv}\boldsymbol{W}\boldsymbol{\tau}_{supg}\left[\boldsymbol{\mathcal{L}}\boldsymbol{U}-\boldsymbol{S}\right]d\Omega+\sum_{e=1}^{n_e}\int_{\Omega_f^e}\delta\frac{\partial \boldsymbol{W}}{\partial x_i}\cdot\frac{\partial \boldsymbol{U}}{\partial x_i}d\Omega=\boldsymbol{0}\end{aligned} \tag{5.4}$$

We can then write out only the Navier-Stokes momentum equation associated with a node $n$ on the fluid side of the boundary $\Gamma_f$

$$\begin{aligned}\int_{\Omega_f} W^n\left(\frac{\partial(\rho v_j)}{\partial t}-S_j^m\right)d\Omega+\int_{\Omega_f}\frac{\partial W^n}{\partial x_i}\left[\tau_{ij}-(\rho v_i v_j+p\delta_{ij})\right]d\Omega+\\ \int_{\Gamma_f}W^n\left[(pv_iv_j+p\delta_{ij})-\tau_{ij}\right]\hat{n}_i\;d\Gamma+\\ \int_{\Omega}\mathcal{L}_{adv}W^n\tau\left[\mathcal{L}\left[\rho v_j\right]-S_j^m\right]d\Omega+\int_{\Omega_f}\delta\frac{\partial W^n}{\partial x_i}\cdot\frac{\partial(\rho v_j)}{\partial x_i}\;d\Omega=0\end{aligned} \tag{5.5}$$

and further separate the boundary integral term by defining $\Gamma_f=\Gamma_{nfsi}\cup\Gamma_{fsi}$ which states that the fluid boundary $\Gamma_f$ is composed of the union of the non-fluid-structure interface boundary $\Gamma_{nfsi}$ and the fluid-structure interface boundary $\Gamma_{fsi}$. According to this definition we make the following split in the boundary integral terms

$$\begin{aligned}\int_{\Omega_f} W^n\left(\frac{\partial(\rho v_j)}{\partial t}-S_j^m\right)d\Omega+\int_{\Omega_f}\frac{\partial W^n}{\partial x_i}\left[\tau_{ij}-(\rho v_i v_j+p\delta_{ij})\right]d\Omega+\\ \int_{\Gamma_f}W^n\left[pv_iv_j\right]\hat{n}_i\;d\Gamma-\int_{\Gamma_{nfsi}}W^n\left[\tau_{ij}-p\delta_{ij}\right]\hat{n}_i\;d\Gamma-\int_{\Gamma_{fsi}}W^n\left[\tau_{ij}-p\delta_{ij}\right]\hat{n}_i\;d\Gamma\\ \int_{\Omega_f}\mathcal{L}^aW^n\tau\left[\mathcal{L}\left[\rho v_j\right]-S_j^m\right]d\Omega+\int_{\Omega_f}\delta\frac{\partial W^n}{\partial x_i}\cdot\frac{\partial(\rho v_j)}{\partial x_i}\;d\Omega=0\end{aligned} \tag{5.6}$$

Following this split, we can group every volume and boundary integral term not associated with the viscous traction on the fluid-structure interface boundary into $\tilde{\mathfrak{R}}_{jf}^n$ as follows

$$\tilde{\mathfrak{R}}_{jf}^n-\int_{\Gamma_{fsi}}W^n\left[\tau_{ij}-p\delta_{ij}\right]\hat{n}_i\;d\Gamma=0\;. \tag{5.7}$$

The tilde over $\mathfrak{R}$ is used to symbolize that this term is an incomplete total residual that only accounts for the integrals over $\int_{\Omega_f}$ and $\int_{\Gamma_{nfsi}}$.

**Elastodynamic Momentum Equation Boundary Term**

In a similar manner to the previous section we can separate out the traction term on the fluid-structure interface in the weak form the elastodynamic equation. The elastodynamic weak form is

repeated here

$$\int_{\Omega_s} \delta u_j \rho \frac{\partial^2 u_j}{\partial t^2} d\Omega + \int_{\Omega_s} \frac{\partial \delta u_j}{\partial x_i} \sigma_{ij} d\Omega - \int_{\Omega_s} \delta u_j b_j d\Omega - \int_{\Gamma_s} \delta u_j \sigma_{ij} \hat{n}_i d\Gamma = 0 \tag{5.8}$$

Again separating the boundary into two distinct regions via $\Gamma_s = \Gamma_{nfsi} \cup \Gamma_{fsi}$ where $\Gamma_{nfsi}$ indicates the non-fluid-structure interface boundary $\Gamma_{nfsi}$ and $\Gamma_{fsi}$ represents the fluid-structure interface boundary. This leads to the previous equation being written for a particular node $n$ as

$$\int_{\Omega_s} \delta u_j^n \rho \frac{\partial^2 u_j}{\partial t^2} d\Omega + \int_{\Omega_s} \frac{\partial \delta u_j^n}{\partial x_i} \sigma_{ij} d\Omega - \int_{\Omega_s} \delta u_j^n b_j d\Omega - \int_{\Gamma_{nfsi}} \delta u_j^n \sigma_{ij} \hat{n}_i d\Gamma - \int_{\Gamma_{fsi}} \delta u_j^n \sigma_{ij} \hat{n}_i d\Gamma = 0 \tag{5.9}$$

which can then be re-written to group residual contributions arising from all volume and boundary integral terms not on the fluid-structure interface into the term $\tilde{\mathfrak{R}}_{j_s}^n$ and the remaining fluid-structure fluid-structure interface boundary integral as

$$\tilde{\mathfrak{R}}_{j_s}^n - \int_{\Gamma_{fsi}} \delta_j^n \sigma_{ij} \hat{n}_i \, d\Gamma = 0 \ . \tag{5.10}$$

**Aeroelastic Interface Enforcement**

Given the traction jump condition of equation 5.3 we can now enforce continuity of the traction by simply replacing the residual terms for the fluid momentum equations at a node on the interface as follows

$$\mathfrak{R}_{j_f}^n \leftarrow \tilde{\mathfrak{R}}_{j_f}^n + \tilde{\mathfrak{R}}_{j_s}^n \tag{5.11}$$

Equation 5.11 implies that the total residual equations for fluid momentum conservation are replaced by the sum of volume and partial boundary integrals of the fluid and solid momentum residual equations.

The velocity jump condition of equation 5.2 is then enforced by replacing the total residual entries of the solid momentum equations at a node on the interface with the constraint equations

$$\mathfrak{R}_{j_s}^n = v_j^n - \dot{u}_j^n \ . \tag{5.12}$$

which explicitly enforces the zero velocity jump condition.

The displacement jump condition of equation 5.1 may be handled in two manners. If the degree-of-freedom type assigned to govern the fluid mesh motion is the same degree-of-freedom type assigned to the solid displacement field, then the fluid-side boundary displacements and the solid-side boundary displacements are automatically equal because they are the same degrees-of-freedom. This, however, means that the fluid mesh stiffness affects the solid's displacements since they are coupled via the same displacement degree-of-freedom. The stiffness contribution of the fluid to the solid's stiffness can be reduced by giving the fluid mesh motion elements an elastic modulus much less than the solid's elastic modulus. The drawback in this case is that scaling issues can arise if a great disparity between element stiffnesses exist. This problem can be alleviated by assigning different displacement degrees-of-freedom to the fluid mesh and solid mesh, hence decoupling the fluid mesh stiffness from the solid mesh stiffness. The additional momentum equations introduced for the fluid mesh motion can then be used for enforcing compatibility of the displacements (i.e. equation 5.1) on the boundary via simple constraint equations as was done for the velocities in equation 5.12.

## 5.3 Aerothermal Coupling

### 5.3.1 Aerothermal Interface Conditions

In order for the fluid and heat equations to be coupled across a boundary, the Dirichlet (essential) and Neumman (natural) boundary conditions must also be continuous across the interface. Considering a purely aerothermal problem for the moment, the jump conditions that must be satisfied are that no jump in the temperature field at the boundary must exist, that is

$$[\![T]\!] = T_f - T_s = 0 \ . \tag{5.13}$$

The subscripts $f$ and $s$ denote the fluid side and solid side temperatures and states that the two temperatures must be equal at the boundary. Additionally, the jump in heat fluxes across the interface must be satisfied

$$[\![q_n]\!] = q_{i_f}\hat{n}_{i_f} - q_{i_f}\hat{n}_{i_s} = 0 \tag{5.14}$$

which states that heat flux normal to the fluid side boundary must balance the heat flux normal to the solid side boundary.

### 5.3.2 Aerothermal Interface Coupling

The so-called residual based enforcement presented here is similar to work developed in reference [59] and is best described as a residual manipulation method. In the case of aerothermal coupling, the nodal energy equation residual terms on the fluid-structure boundary are modified to enforce the jump conditions listed above. Beginning from the weak form of the residual (equation 2.75), repeated here for convenience

$$\begin{gathered}\int_{\Omega} \boldsymbol{W} \cdot \left( \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - \boldsymbol{S}_i \right) d\Omega + \int_{\Omega} \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \boldsymbol{G}_i d\Omega - \int_{\Gamma} \boldsymbol{W} \boldsymbol{G}_i d\Gamma - \\ \sum_{e=1}^{n_{ele}} \int_{\Omega} \frac{\partial \boldsymbol{W}}{\partial x_k} \boldsymbol{A}_k \boldsymbol{\tau}_{supg} \left[ \frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_i}{\partial x_i} - \frac{\partial \boldsymbol{G}_i}{\partial x_i} - \boldsymbol{S}^h \right] d\Omega + \sum_{e=1}^{n_{ele}} \int_{\Omega} \delta \frac{\partial \boldsymbol{W}}{\partial x_i} \cdot \frac{\partial \boldsymbol{U}}{\partial x_i} d\Omega = \boldsymbol{0}\end{gathered} \tag{5.15}$$

we can single out only the Navier-Stokes energy residual associated with a node $n$ on the fluid side of the boundary $\Gamma_f$

$$\begin{gathered}\int_{\Omega_f} W^n \cdot \left( \frac{\partial (\rho E)}{\partial t} + \frac{\partial (\rho E u_i + P u_i)}{\partial x_i} - S_i^e \right) d\Omega + \int_{\Omega_f} \frac{\partial W^n}{\partial x_i} \cdot (\tau_{ij} u_j - q_i)\, d\Omega - \\ \int_{\Gamma_f} W^n \cdot (\tau_{ij} u_j - q_i)\, \hat{n}_i \, d\Gamma - \\ \int_{\Omega_f} \frac{\partial W^n}{\partial x_k} \boldsymbol{A}_k \boldsymbol{\tau}_{supg} \left[ \frac{\partial (\rho E)}{\partial t} + \frac{\partial (\rho E u_i + P u_i)}{\partial x_i} - \frac{\partial (\tau_{ij} u_j - q_i)}{\partial x_i} - S_i^e \right] d\Omega_f + \\ \int_{\Omega_f} \delta \frac{\partial W^n}{\partial x_i} \cdot \frac{\partial (\rho E)}{\partial x_i} \, d\Omega = 0 \ .\end{gathered} \tag{5.16}$$

The previous equation can be rearranged to a more compact form which singles out the viscous boundary integral; the residual equation may now be written as

$$\tilde{\mathfrak{R}}_f^n - \int_{\Gamma_f} W^n \cdot (\tau_{ij} u_j - q_i)\, \hat{n}_i \, d\Gamma = 0. \tag{5.17}$$

where again the tilde indicates that this is only the part of the total residual that includes the volume integral and the boundary integral that is not on the fluid-structure interface. Note that in case of a viscous no-slip boundary condition on the interface $u_i = 0$ so the previous equation

reduces to

$$\tilde{\mathfrak{R}}_f^n + \int_{\Gamma_f} W^n \; q_i \; \hat{n}_{i_f} \; d\Gamma = 0. \tag{5.18}$$

In a similar manner, the weak form of the energy equation for transient heat transfer associated with a node $n$ on the solid side of the boundary $\Gamma_s$ may be written as

$$\int_{\Omega_s} W^n \left( \rho C \frac{\partial T}{\partial t} - \dot{Q} \right) d\Omega - \int_{\Omega_s} \frac{\partial W^n}{\partial x_i} \; k_{ij} \frac{\partial T}{\partial x_i} d\Omega + \int_{\Gamma_s} W^n \; q_i \hat{n}_i \; d\Gamma = 0 \tag{5.19}$$

which can also be written in a form that singles out the boundary heat flux term

$$\tilde{\mathfrak{R}}_s^n + \int_{\Gamma_s} W^n \; q_i \; \hat{n}_{i_s} \; d\Gamma = 0. \tag{5.20}$$

Returning now to the compatibility conditions listed in equations 5.13 and 5.14 we can combine equations 5.18 and 5.20 to enforce the flux condition

$$\int_{\Gamma_f} W^n \; q_i \; \hat{n}_{i_f} \; d\Gamma - \int_{\Gamma_s} W^n \; q_i \; \hat{n}_{i_s} \; d\Gamma = \tilde{\mathfrak{R}}_f^n - \tilde{\mathfrak{R}}_s^n = 0. \tag{5.21}$$

Based on this we can manipulate the nodal residual equations to satisfy the energy balance. The residual contribution for the fluid's Navier-Stokes energy equation at node $n$ becomes

$$\mathfrak{R}_f^n \leftarrow \tilde{\mathfrak{R}}_f^n - \tilde{\mathfrak{R}}_s^n \; . \tag{5.22}$$

and the residual contribution for the solid's heat transfer energy equation at node $n$ becomes

$$\mathfrak{R}_s^n = T_s^n - T_f^n = 0. \tag{5.23}$$

Equations 5.22 and 5.23 above effectively enforce the two jump conditions given by equations 5.13 and 5.14.

## 5.4 Aerothermoelastic Coupling

Aerothermoelastic problems involve solving the governing equations for Navier-Stokes, transient heat transfer, and elastodynamics in a coupled fashion. In this work, aerothermoelastic problems are solved by combining the finite element formulation for compressible gas dynamics

described in chapter 2 with the thermoelastic coupling approach described in chapter 4. The interface coupling procedures for aeroelasticity and aeroheating described in sections 5.2 and 5.3 of this chapter are simply combined to couple the momentum equations and energy equation at the fluid-structure interface. The only additional consideration is that the wall velocity is non-zero for the energy equation and the mesh is moving. Hence, the ALE based formulation discussed in chapter 2 applies.

Time integration of the aerothermoelastic problems contained is thesis are performed with a hybrid BDF-Generalized-$\alpha$ method. The BDF time derivative approximation given in section 2.4.2 is used for the first-order-in-time equations for the fluid and heat transfer problems. The Generalized-$\alpha$ time derivatives are used for the second-order-in-time equations that arise from the structural dynamics part of the coupled system. This hybrid scheme efficiently handles the truly monolithic coupling for aerothermoelastic problems in an elegant and easy to implement manner.

## 5.5 Numerical Example Problems

### 5.5.1 2-Dimensional Nose Tip Aeroheating Response

This example problem uses the aerothermal coupling formulation presented in section 5.3 to solve the coupled aerodynamic/heat transfer problem for the nose tip example shown in chapter 2.6.3.

#### 5.5.1.1 Problem Description

The problem geometry and boundary conditions are shown in Figure 5.1, however, now the energy equations at the fluid/structure interface are governed by equations 5.13 and 5.14. The flow conditions for this problem are that of a $M_\infty = 3.0$ flow at 40 $km$ altitude where $\rho_\infty = 3.99641 \times 10^{-3}\ kg/m^3$ and $T_\infty = 250.35\ K$. Freestream flow values for $\rho, \rho v_x, \rho v_y$, and $\rho E$ are prescribed on the inlet boundary. Slip boundary condition ($v_y = 0$) is used for the edge lying on the x-axis. A no-slip wall boundary condition is used for the surface of the body. No values

are prescribed for the outflow edge and the viscous fluxes are integrated to be consistent with the integrated-by-parts weak form of the viscous fluxes.

The gas constants for air are specified as $R = 287.0$ and $\gamma = 1.4$. The two-coefficient Sutherland model for air is used to compute the viscosity ($\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110.4\ K$) and the Prandtl number ($Pr = 0.70$) is used to compute the thermal conductivity of the air. The material properties of the solid are $\rho = 2700\ m/kg^3$, $k = 200\ W/(m \cdot K)$ or $k = 2\ W/(m \cdot K)$, and $C_p = 870\ J/(kg \cdot K)$.

#### 5.5.1.2 Computational Mesh and Model

The computational mesh used for this problem contains a region for solving the Navier-Stokes equations of the compressible fluid flow and a region for solving the heat transfer and ablation response of the structure. These two zones of the mesh are connected or contiguous at the fluid/solid interface as is required by the coupling formulation presented in this chapter. As previously mentioned, contiguous meshes are not a strict requirement of this coupling scheme as it is also easily extensible to non-matching meshes. The fluid mesh is identical to the coarse fluid mesh used in example 2.6.3, which showed that this coarsest mesh yields solutions that are very close to much more refined meshes. The solid mesh fills the region of the solid, first beginning with a structured grid going inward normal to the fluid/solid interface and then progressing to a unstructured grid to fill the remainder of the region. Figure 5.2 shows this fluid/solid computational mesh.

The simulation is run until 200.0 $s$ physical time using the BDF-2 time integrator with adaptive time stepping where an initial time step size of $1 \times 10^{-7}\ s$ and a maximum time step increase of 1.2 are used. Newton's method with an approximate linearization is used to solve the nonlinear problem at each time step. A nonlinear relaxation factor of 0.9 is used with a nonlinear residual drop criteria of $\epsilon = 0.1$. The Trilinos/Amesos UMFPACK direct solver is used for solving the linear problem at each nonlinear step. The SUPG term in the weak residual equation uses node averaged stabilization and $\delta$ discontinuity capturing parameters.

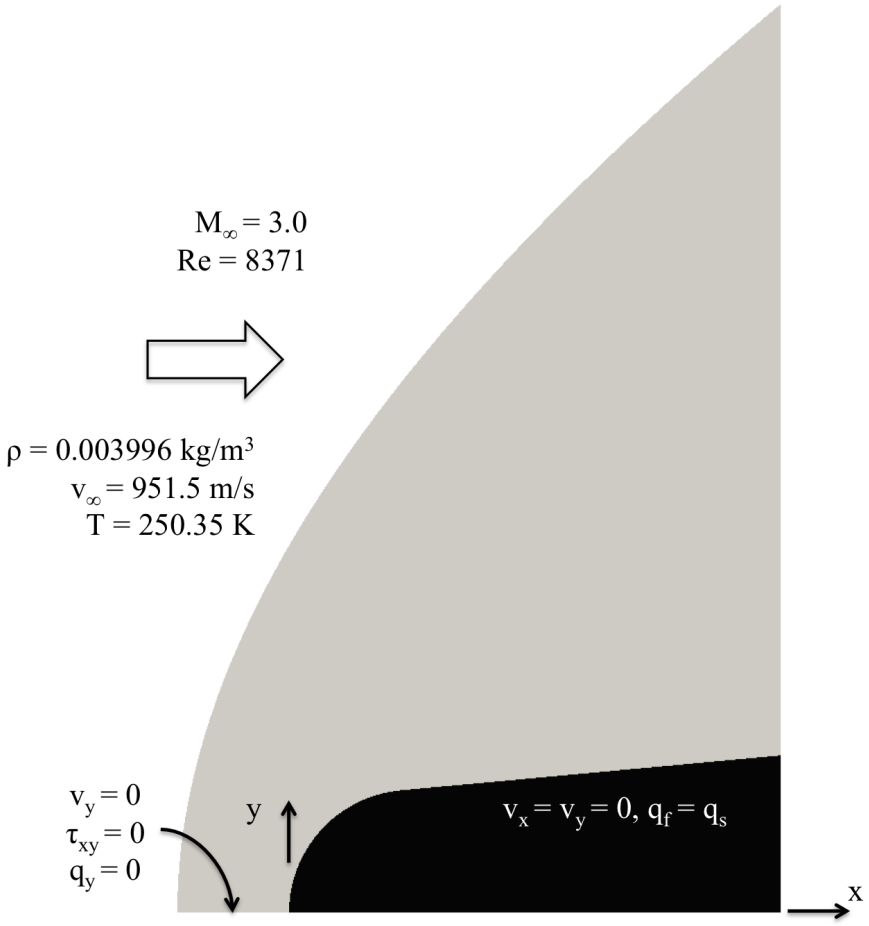



Figure 5.1: Two-dimensional nose tip aerodynamic heating problem geometry and boundary conditions.

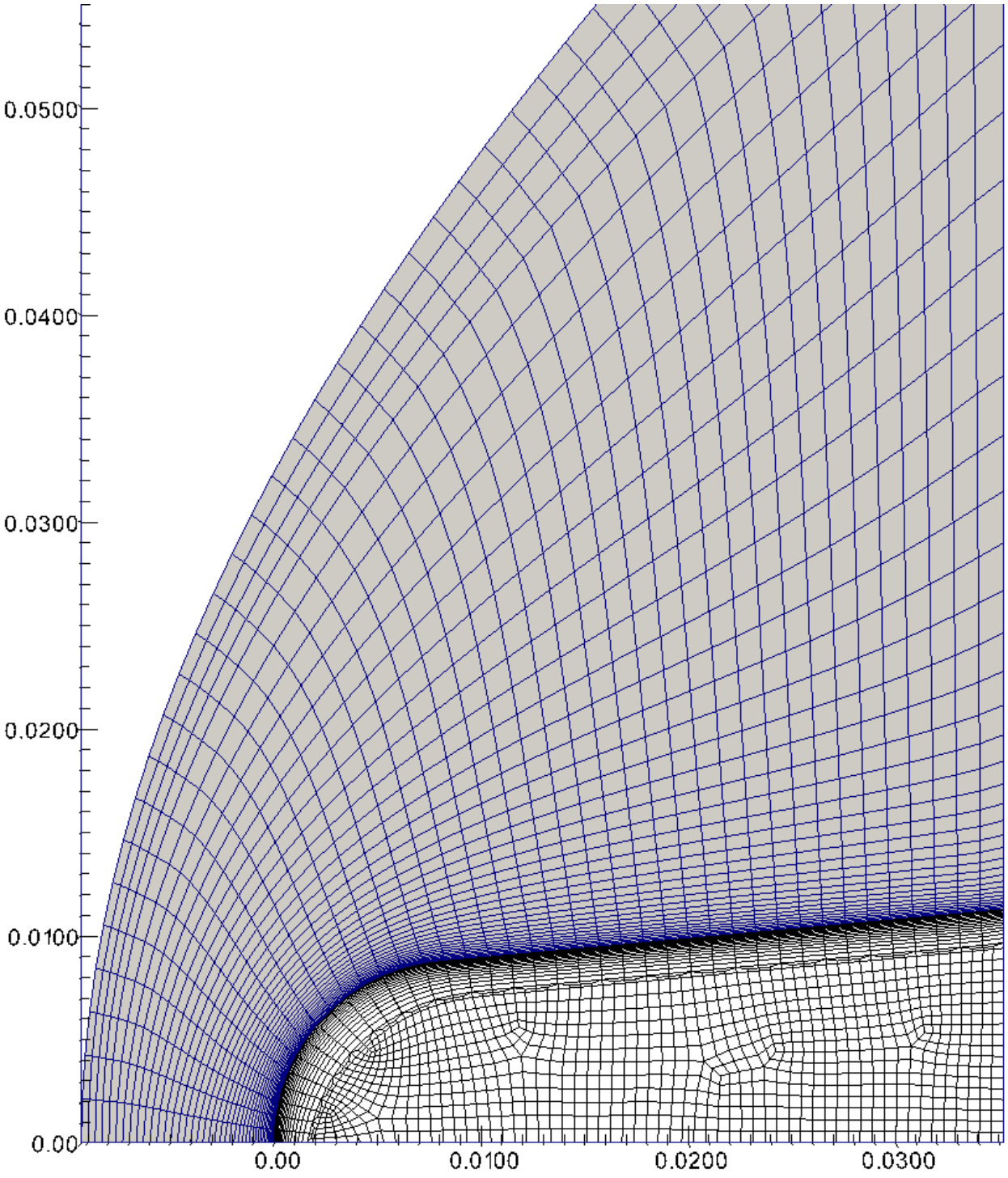
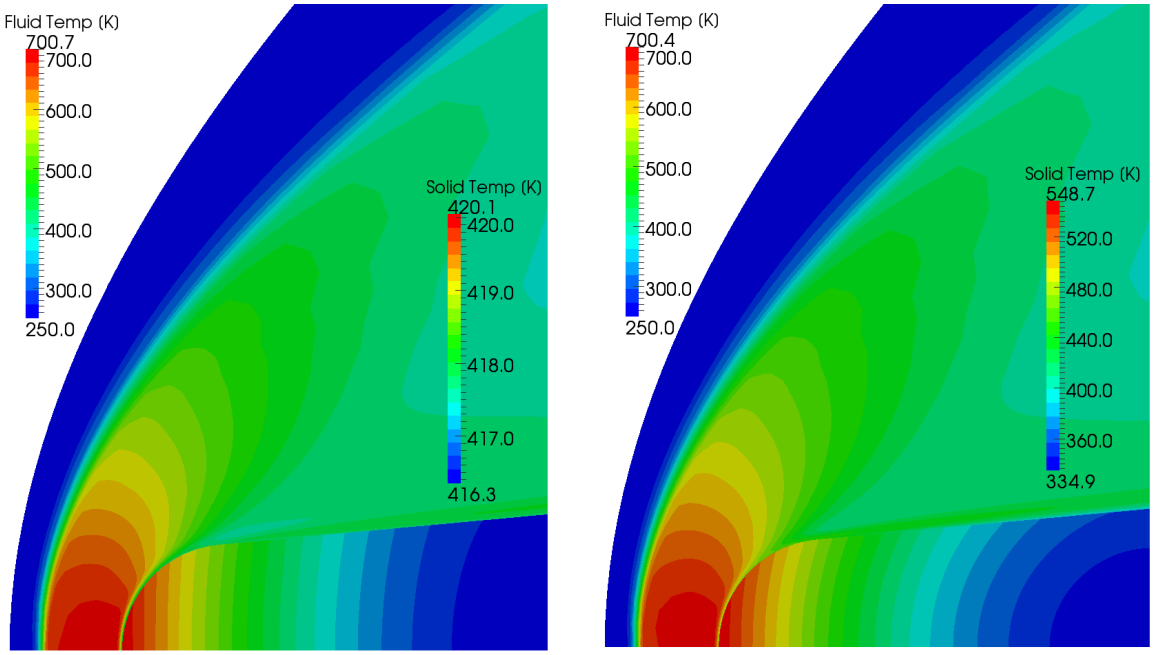



Figure 5.2: Two-dimensional nose tip aerodynamic heating problem fluid/solid mesh.

#### 5.5.1.3 Results and Discussion

Figure 5.3 shows the fluid and solid temperature contours at 200.0 $s$. Several interesting observations can be made from this figure. First, in contrast to the adiabatic wall temperature contour shown in Figure 2.18(d) the boundary layer appears much cooler as result of the heat being transferred into the structure. In Figure 5.3 the solid is nearly all the same temperature throughout (the minimum and maximum temperatures span less than 5 $K$) due to the high ($k = 200\ W/(m \cdot K)$) thermal conductivity of the solid. When the problem is run with a much lower thermal conductivity ($k = 2\ W/(m \cdot K)$), Figure 5.3(b) shows the disparity between the minimum and maximum temperature of the solid is much greater (approximately 214 $K$). This observation is a motivating factor for using the GGLS formulation presented in section 3.4.2 for aeroheating problems with high wall heat fluxes and low thermal conductivities of the solid where strong temperature gradients may lead to non-physical solution behavior.




(a) Solid thermal conductivity k = 200 $W/m - K$ (b) Solid thermal conductivity k = 2 $W/m - K$

Figure 5.3: 2D nose tip aerodynamic heating problem temperature contours at 200.0 $s$.

The two temperature scales in Figure 5.3 make it difficult to observe that the fluid tempera-

ture and the solid temperature at the interface are indeed the same. Figure 5.5, however, shows the fluid and solid wall temperature profiles at several instances in time. From this it is easily observed that the fluid and solid temperatures match exactly at each time step.

The next figure shows the temperature at various x-coordinate locations along the centerline. From this figure it is obvious that the high thermal conductivity material response produces and essentially uniform temperature field while the low thermal conductivity material produces a greatly differing temperature distribution throughout the body. This is an important consideration from a transient aerodynamic heating perspective – under what transient conditions is it beneficial to reduce peak structural temperatures with the use of high thermal conductivity materials and under what conditions is it better to reduce heating in certain regions at the expense of much higher temperatures in other regions? The answer to this question difficult and often hard to address, and is the subject of later chapters on design optimization for transient problems.

### 5.5.2 2-Dimensional Cylinder Aerothermoelastic Response

In this problem, the aeroelastic and aerothermal coupling formulations presented in sections 5.2 and 5.3 are combined to solve the three-field aerodynamic/heat transfer/elastic response problem for a cylinder in a supersonic flow.

#### 5.5.2.1 Problem Description

Figure 5.6 shows the geometry and boundary conditions for this problem. The free-stream conditions of the flow are $M_\infty = 5.0$ at 40 $km$ altitude where $\rho_\infty = 3.99641 \times 10^{-3}\ kg/m^3$ and $T_\infty = 250.35\ K$. Freestream flow values for $\rho, \rho v_x, \rho v_y$, and $\rho E$ are prescribed on the inlet boundary. Slip boundary condition ($v_y = 0$) is used for the edge lying on the x-axis. The wall momentum and energy equations are governed by the coupling conditions as described in sections 5.2 and 5.3. No values are prescribed for the outflow edge however the boundary flux integrals are performed as has been done in many of the previous problems.

The gas constants for air are specified as $R = 287.0$ and $\gamma = 1.4$. The two-coefficient

Sutherland model for air is used to compute the viscosity ($\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110.4\ K$) and the Prandtl number ($Pr = 0.70$) is used to compute the thermal conductivity of the air. The material properties of the solid are $\rho = 2700\ m/kg^3$, $k = 50\ W/(m \cdot K)$, and $C_p = 870\ J/(kg \cdot K)$. The elastic modulus of the solid is $E = 1.0 \times 10^6\ N/m^2$, the Poisson ratio is $\nu = 0.33$, the coefficient of thermal expansion is $\alpha = 1.0 \times 10^{-}5/K$ with a reference temperature $T_{ref} = 250\ K$. Rayleigh damping is used for the structural dynamics problem where $\alpha_d = 0.002$ and $\beta_d = 0.0002$.

#### 5.5.2.2 Computational Mesh and Model

Figure 5.7 show the computational mesh for this problem, which is a 48x24 element structured mesh for the fluid domain and a 24x24 element structured mesh for the solid domain. Again, the fluid and solid meshes are connected/contiguous at the interface as this is required by the coupling method implemented here.

The simulation is run until 200.0 $s$ physical time using a hybrid BDF/Generalized-$\alpha$ time integration method with adaptive time stepping where an initial time step size of $1 \times 10^{-9}\ s$ and a maximum time step increase of 1.1 are used. Newton's method with an approximate linearization is used to solve the nonlinear problem at each time step. A nonlinear relaxation factor of 0.5 is used with a nonlinear residual drop criteria of $\epsilon = 0.1$. The Trilinos/Amesos UMFPACK direct solver is used for solving the linear problem at each nonlinear step. The SUPG term in the weak residual equation uses node averaged stabilization and $\delta$ discontinuity capturing parameters. The GGLS scheme is used for the heat equations to limit any strong gradients that may arise. The standard Galerkin formulation shown in section 4.3 is used for the elastodynamic equations.

#### 5.5.2.3 Results and Discussion

Figure 5.9 show various coupled response contours for the aerothermoelastic response at the simulation end time ($100,000\ s$). The deformation of the cylinder has been magnified 10 times to more evidently show the near steady-state conditions of the cylinder.

Figure 5.9(a) shows the temperature profiles through time of the inner and outer surface boundaries of the cylinder at the X- and Y-axes. The structural temperatures begin to rise significantly from the initial 250 $K$ value after about 1 $s$. The fluid temperature rise occurs much faster than this but heat transfer into the structure occurs on a much larger time scale than the fluid. The temperature of the structure reaches a steady-state value of about 1400 $K$ around $70,000$ $s$ of physical time. Figure 5.9(b) shows the displacement response of the structure during the simulation. The X-displacement of the outer surface at Y=0 clearly shows two distinct events. The first this the displacement response that occurs roughly before $t = 1$ $s$. This displacement occurs because of the aerodynamic forces that act to compress the cylinder. The time scale of the aerodynamic forcing response is clearly quite fast as the displacement responses appear to reach a steady state between $t = 0.5$ $s$ and $t = 7,000$ $s$. After roughly $t = 7,000$ $s$ a second displacement response occurs that is associated with the thermal expansion of the structure. Once the solid temperatures rise due to aerodynamic heating the thermal expansion of the solid begins to induce thermal strains. This causes a displacement that "pushes back" against the aerodynamic pressure loading, as seen in Figure 5.9(b).

### 5.5.3 2-Dimensional Flat Plate Aerothermoelastic Response

In this problem, the aeroelastic and aerothermal coupling formulations presented in sections 5.2 5.3 are combined to solve the three-field aerodynamic/heat transfer/elastodynamic response problem for the supersonic flow over a flat plate.

#### 5.5.3.1 Problem Description

Figure 5.10 shows the flat plate problem geometry and boundary conditions. The free-stream conditions of the flow are $M_\infty = 2.0$ with $\rho_\infty = 0.4$ $kg/m^3$ and $T_\infty = 223.95$ $K$. Freestream flow values for $\rho, \rho v_x, \rho v_y$, and $\rho E$ are prescribed on the inlet boundary. Slip boundary condition ($v_y = 0$) is used for the edge lying on the x-axis. The wall momentum and energy equations are governed by the coupling conditions as described in sections 5.2 and 5.3. No values are prescribed for the

outflow edge however the boundary flux integrals are performed as has been done in many of the previous problems.

The gas constants for air are specified as $R = 287.0$ and $\gamma = 1.4$. The two-coefficient Sutherland model for air is used to compute the viscosity ($\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110.4\ K$) and the Prandtl number ($Pr = 0.71$) is used to compute the thermal conductivity of the air. The thermal material properties of the solid are $\rho = 2800\ m/kg^3$, $k = 170\ W/(m \cdot K)$, and $C_p = 875\ J/(kg \cdot K)$. The elastic modulus of the solid is $E = 7.3 \times 10^1 0\ N/m^2$, the Poisson ratio is $\nu = 0.33$, the coefficient of thermal expansion is $\alpha = 22.5 \times 10^- 6/K$ with a reference temperature $T_{ref} = 223.95\ K$. Rayleigh damping is used for the structural dynamics problem where $\alpha_d = 0.002$ and $\beta_d = 0.0002$.

#### 5.5.3.2 Computational Mesh and Model

Figure 5.11 show the computational mesh for this problem, which is a 160x40 element structured mesh for the fluid domain and a 150x16 element structured mesh for the solid domain.

The simulation is run until 0.1 $s$ physical time using a hybrid BDF/Generalized-$\alpha$ time integration method with adaptive time stepping where an initial time step size of $2 \times 10^{-7}\ s$ and a maximum time step increase of 1.1 are used. The maximum time step size is limited to $1 \times 10^{-4}\ s$ in order to capture the high frequency vibrational response of the plate. Newton's method with an approximate linearization is used to solve the nonlinear problem at each time step. A nonlinear relaxation factor of 0.5 is used with a nonlinear residual drop criteria of $\epsilon = 0.1$. The Trilinos/Amesos UMFPACK direct solver is used for solving the linear problem at each nonlinear step. The SUPG term in the weak residual equation uses node averaged stabilization and $\delta$ discontinuity capturing parameters. The standard Galerkin formulation is used for both the heat transfer and the elastodynamic equations.

#### 5.5.3.3 Results and Discussion

Figure 5.12 shows the fluid and solid temperature fields at the simulation end time of 0.1 $s$. The high Reynolds number of the problem results in a very thin boundary layer that lies very close to the plate. Due to the relatively short simulation time, the heat transfer into the plate is not significant and the thermal expansion effect of the plate is minimal.

Figure 5.13 shows the Y-displacement response of the center point of the flat plate through time. It is noted that the aerodynamic forces cause vibrational response of the structure that gradually damps out with time. This observation is consistent with classical examples of high-speed flow aeroelastic panel responses.

Figure 5.14 shows the flat plate temperature profiles at 0.043 $s$ and 0.1. The mean temperature of the plate is not significantly above the thermal expansion reference temperature ($T_{ref}$ = 223.95 $K$, hence the thermal expansion effect is negligable for this problem.

### 5.5.4 2-Dimensional Nose Tip Ablation Response

This example problem uses the aerothermal coupling formulation presented in section 5.3 and the Q* ablation formulation in section 3.4 to solve the coupled aerodynamic/heat transfer/ablation problem for the nose tip example shown in chapter 2.6.3.

#### 5.5.4.1 Problem Description

Figure 5.15 shows the geometry and boundary conditions for this problem. The setup is very similar to example 5.5.1 however now the fluid/solid interface is allowed to recede according the Q* ablation model. This problem is run at three different Mach numbers to examine differences in the recession rate of the structure.

The free-stream conditions of the flow are $M_\infty = 3.0$, $M_\infty = 4.0$, and $M_\infty = 5.0$ with $\rho_\infty = 3.99641 \times 10^{-}3\ kg/m^3$ and $T_\infty = 250\ K$. Freestream flow values for $\rho, \rho v_x, \rho v_y$, and $\rho E$ are prescribed on the inlet boundary. Slip boundary condition ($v_y = 0$) is used for the edge lying on the x-axis. The wall momentum and energy equations are governed by the coupling conditions as

described in sections 5.3. No values are prescribed for the outflow edge however the boundary flux integrals are performed as has been done in many of the previous problems.

The gas constants for air are specified as $R = 287.0$ and $\gamma = 1.4$. The two-coefficient Sutherland model for air is used to compute the viscosity ($\mu_{ref} = 1.458 \times 10^{-6}\ kg/m \cdot s \cdot K^{1/2}$ and $T_{ref} = 110.4\ K$) and the Prandtl number ($Pr = 0.71$) is used to compute the thermal conductivity of the air. The thermal material properties of the solid are similar to SLA-561V, an Apollo era ablation material. The use of a Q* model is not valid for this type of decomposing ablator, however, this example serves the purpose of demonstrating the capabilities of this coupling method. The material properties are $\rho = 480\ m/kg^3$, $k = 0.12\ W/(m \cdot K)$, $C_p = 1172\ J/(kg \cdot K)$, $\mathcal{L}_h = 5.41 \times 10^7 J/kg$, and the ablation temperature is $T_{abl} = 588\ K$.

#### 5.5.4.2 Computational Mesh and Model

This problem is solved with both a 41x40 element and a 74x80 element fluid mesh and corresponding unstructured contiguous solid meshes. Figure 5.16 show the caorsest hybrid computational mesh.

The simulation is run for total physical times of between 27 $s$ and 80 $s$ time using a $\Theta$-scheme time integration method with adaptive time stepping; an initial time step size of $2 \times 10^{-7}\ s$ is used with a maximum time step increase of 1.2. The maximum time step size is limited to $5 \times 10^{-1}\ s$ in order to capture salient features of the ablation response. Newton's method with an approximate linearization is used to solve the nonlinear problem at each time step. A "fail-safe" nonlinear strategy is used: if the nonlinear solve fails to converge using a relaxation factor of 0.9 then it will retry with a relaxation factor 0.6. This has proven be effective for these types of problems that initially behave linearly then become nonlinear during the simulation. A nonlinear residual drop criteria of $\epsilon = 0.02$ is used. The Trilinos/Amesos UMFPACK direct solver is used for solving the linear problem at each nonlinear step. The SUPG term in the weak residual equation uses node averaged stabilization and $\delta$ discontinuity capturing parameters. The standard Galerkin formulation is used for both the heat transfer with Q* ablation elements on the fluid/solid interface.

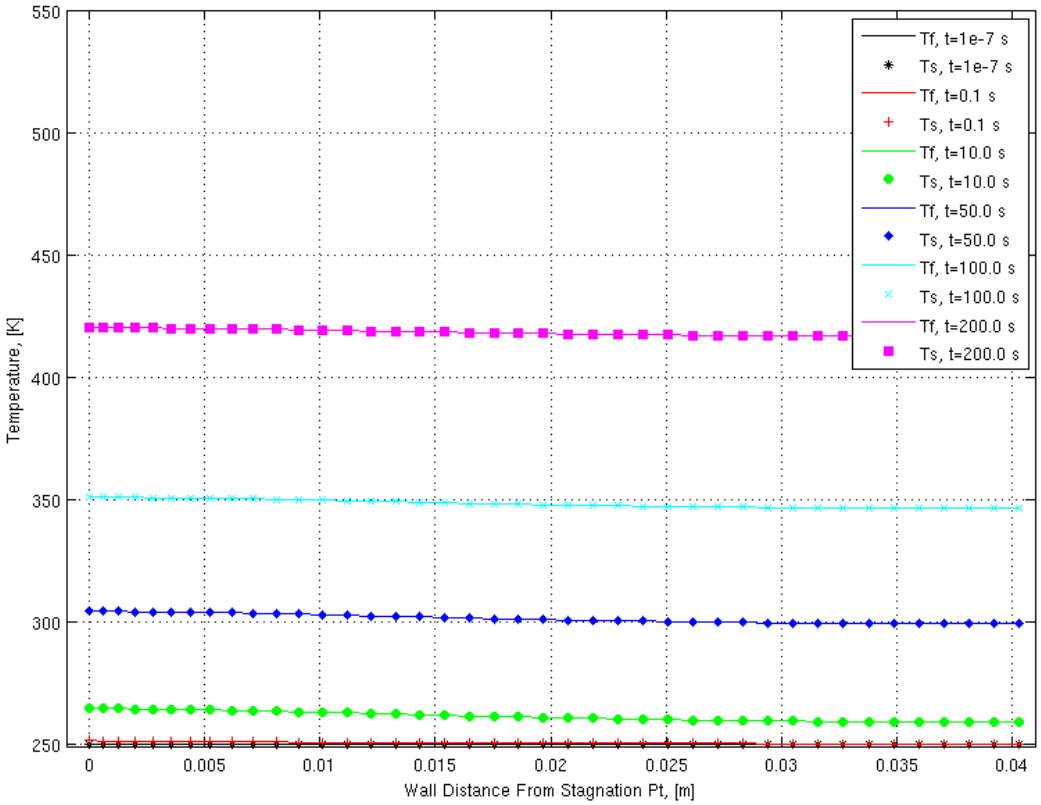
#### 5.5.4.3 Results and Discussion

Figures 5.17–5.19 shows the ablation response at a time snapshot during the simulation for the Mach 3, 4, and 5 freestream flows. Figure 5.17 shows an interesting response. At this instant in time the simulation ended due the mesh collapsing on itself. The peculiar shape of the ablated surface may be explained by the fact that the coupled wall temperature does not exceed the ablation temperature of 588 $K$ at a certain point along the nose. Due to the lower Mach number of this problem ($M_\infty = 3.0$), the heat fluxes do not drive the wall temperature as high as the high Mach cases. Since the point at which the wall temperature does not exceed the ablation temperature occurs on the nose, and because of the (relatively) coarse discrete representation of the surface by finite elements, one of the nodes on the wall becomes a pivot point for the ablation front. However, given a fine enough mesh, this problem should correct itself.

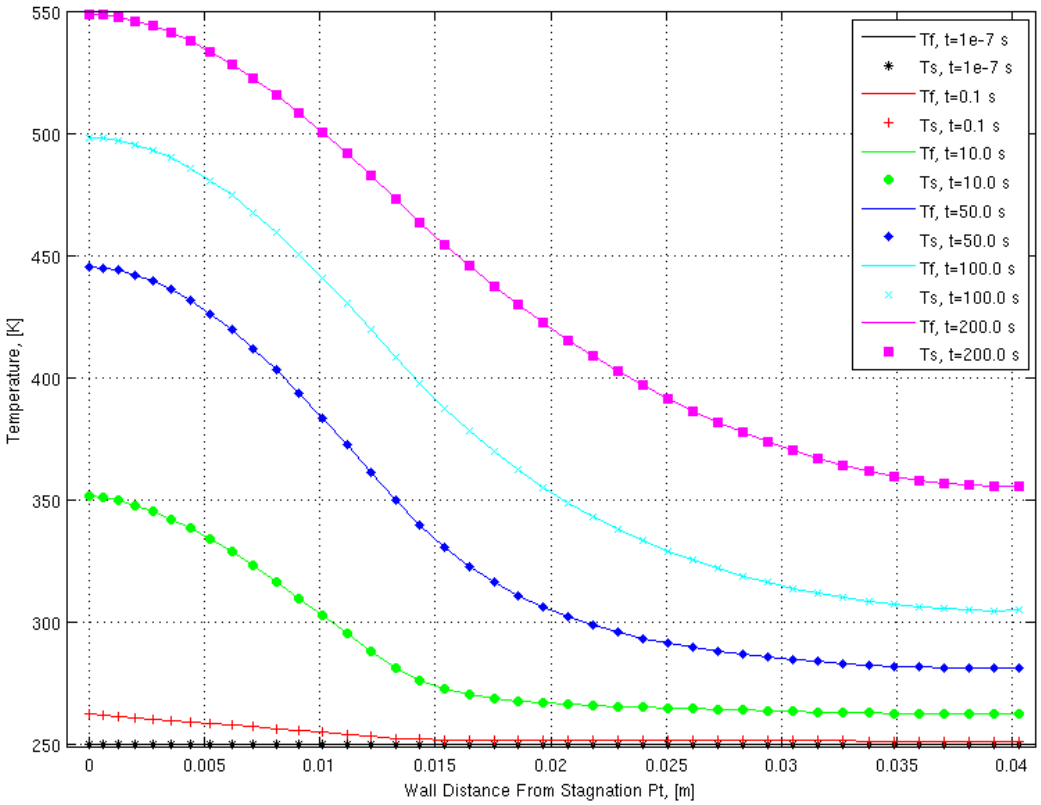
Figures 5.17 and 5.19 show the Mach 4 and Mach 5 ablation responses. These problems clearly do not exhibit the same numerical problems the Mach 3 case displayed. This is due to the fact that the coupled wall temperature exceeds the ablation temperature well past the nose of the body and occurs somewhere farther back on the side wall.

Figure 5.20 shows the surface recession time histories for the various Mach numbers and meshes. The non-smooth response of the coarser meshes is improved by mesh refinement, as shown in the figure.

Figure 5.21 shows the stagnation point temperatures of the various problems. As the surface temperature reaches the ablation temperature, the penalty formulation used to enforce the ablating wall temperature does a relatively good job of holding the ablation temperature. However, a higher penalty factor would lead to a tighter temperature constraint but at the expense of worsening scaling of the linear system.




(a) Solid thermal conductivity k = 200 $W/m-K$




(b) Solid thermal conductivity k = 2 $W/m-K$

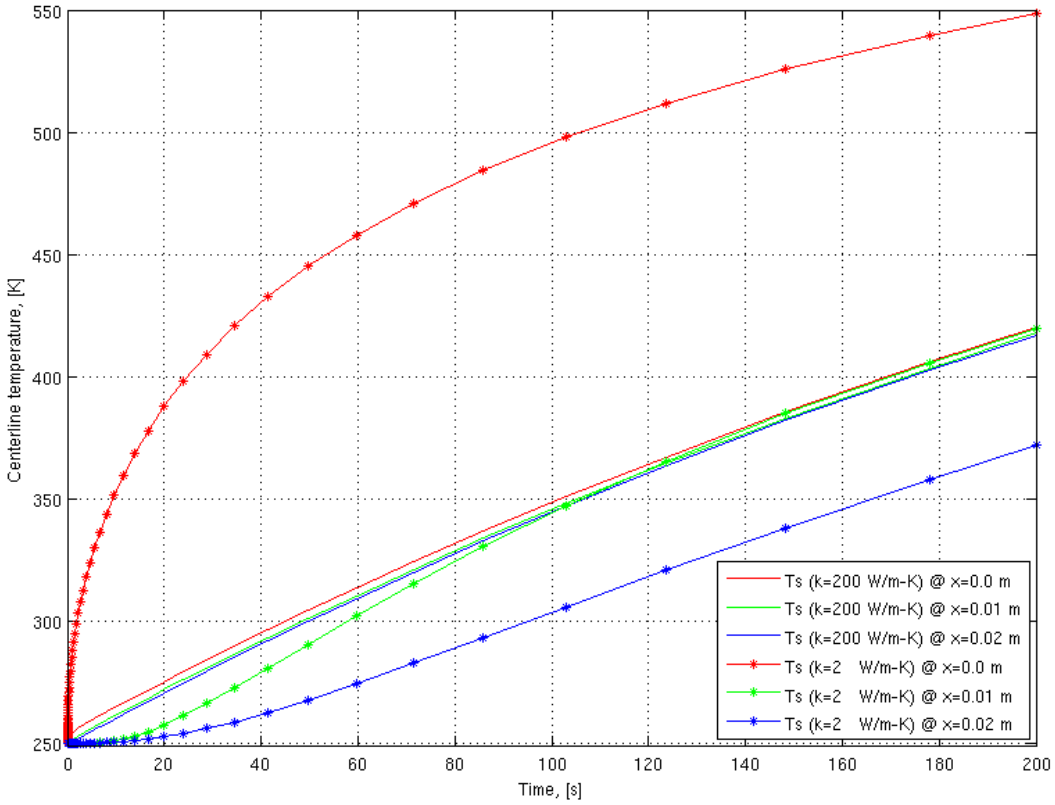



Figure 5.5: 2D nose tip aerodynamic heating problem centerline temperatures

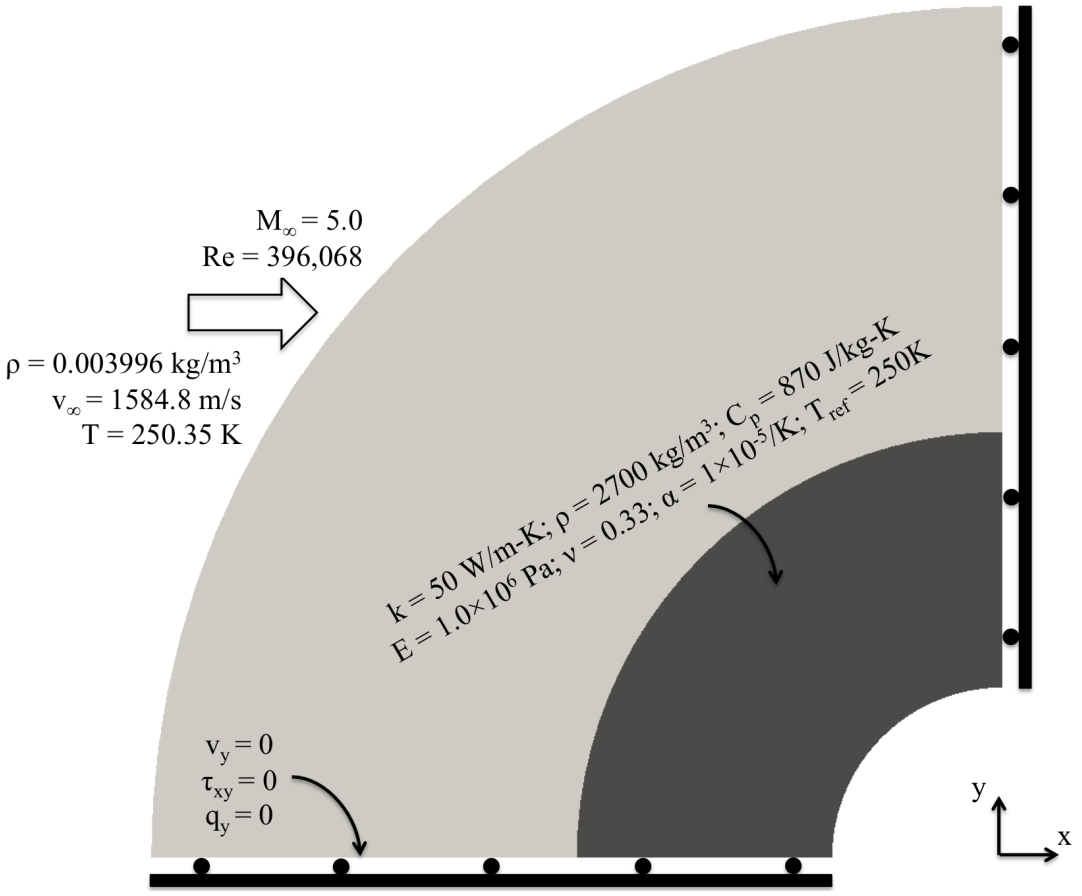



Figure 5.6: 2D cylinder aerothermoelastic problem geometry and boundary conditions.

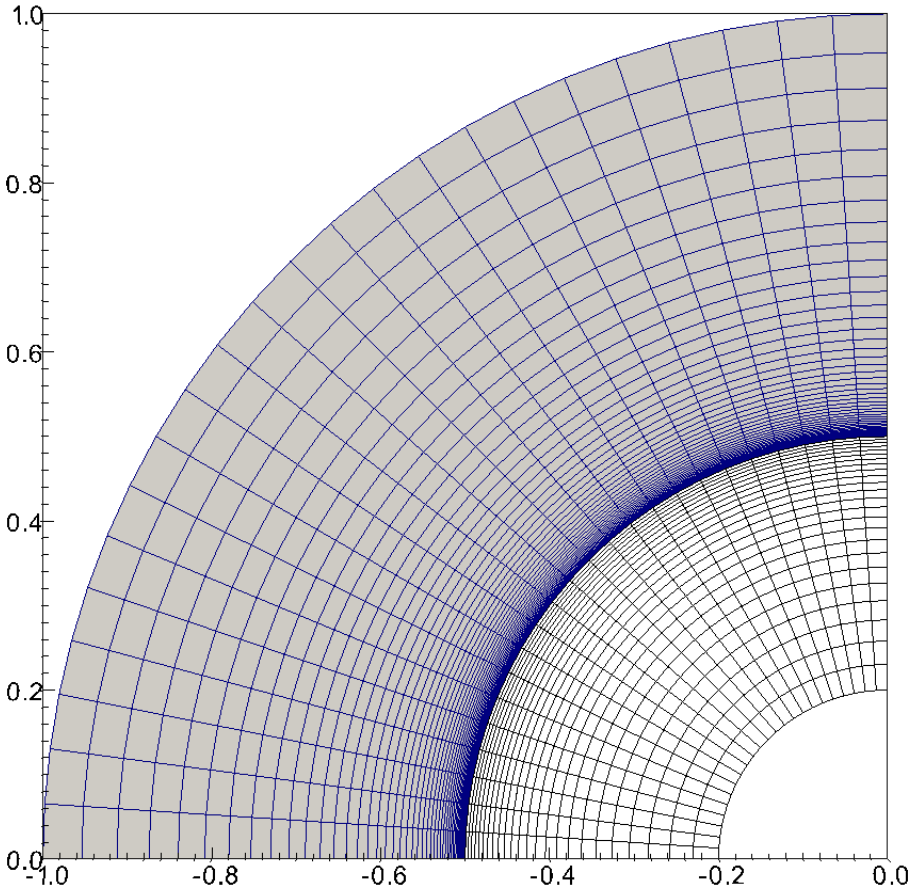
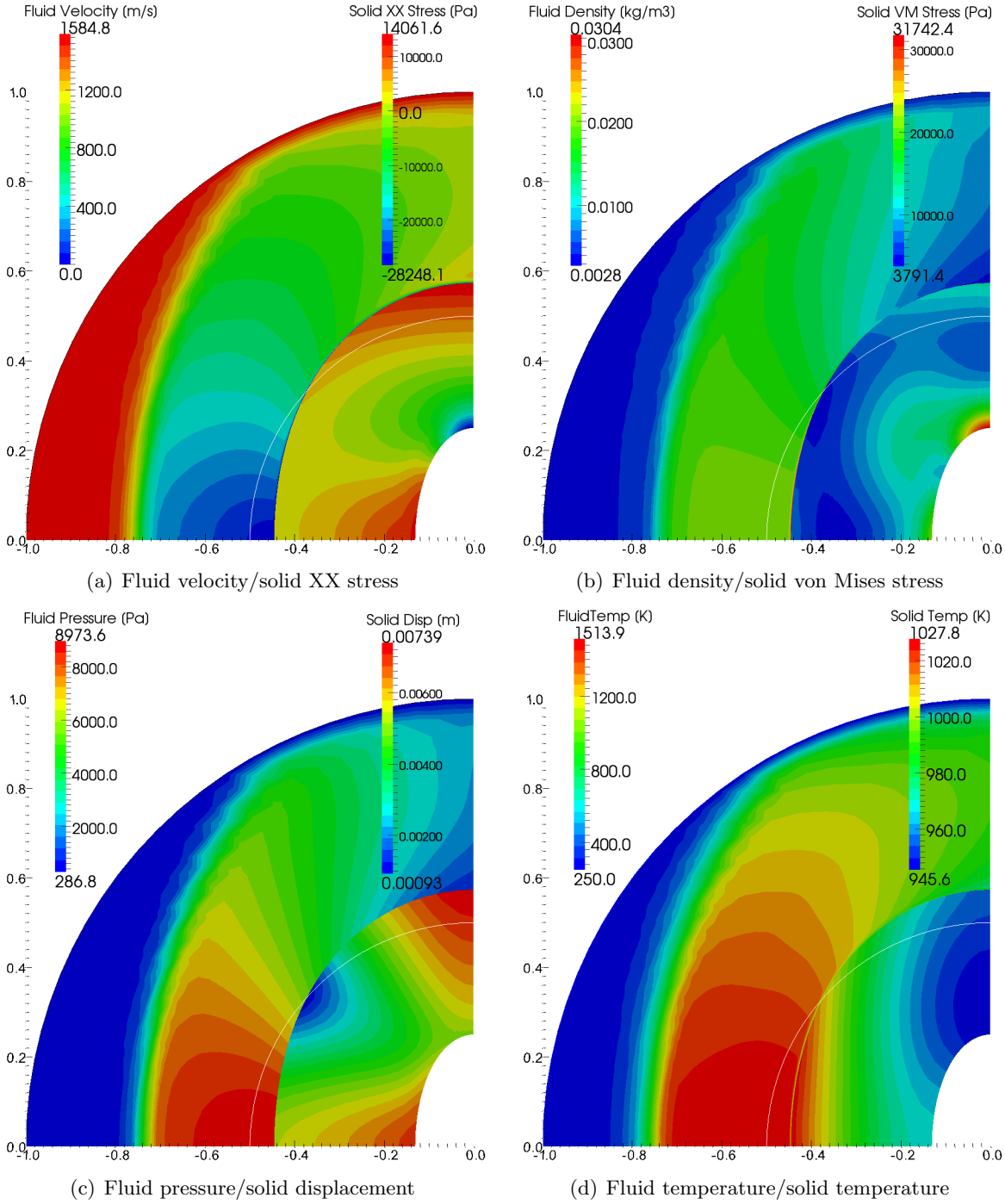



Figure 5.7: 2D cylinder aerothermoelastic problem fluid/solid mesh.



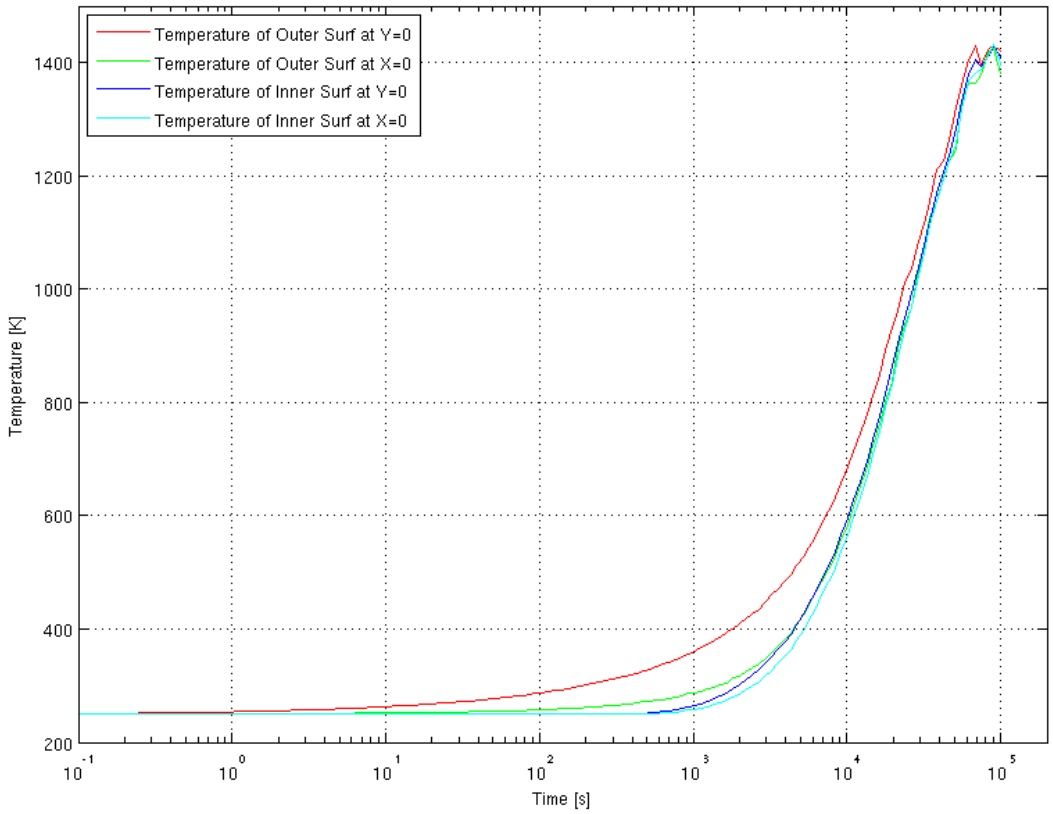

(a) Fluid velocity/solid XX stress

(b) Fluid density/solid von Mises stress

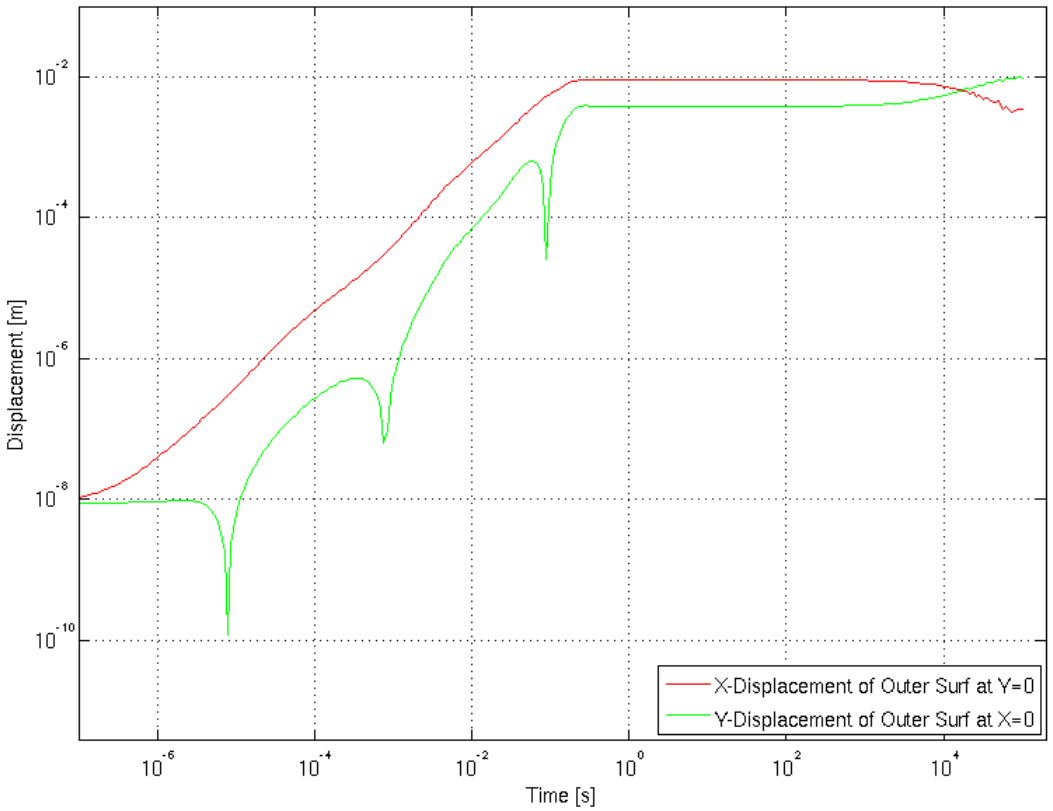
(c) Fluid pressure/solid displacement

(d) Fluid temperature/solid temperature

Figure 5.8: 2D cylinder aerothermoelastic response contours at simulation end time. (NOTE: the structural deformations shown have been magnified 10x.)




(a) Temperature response.




(b) Displacement response.

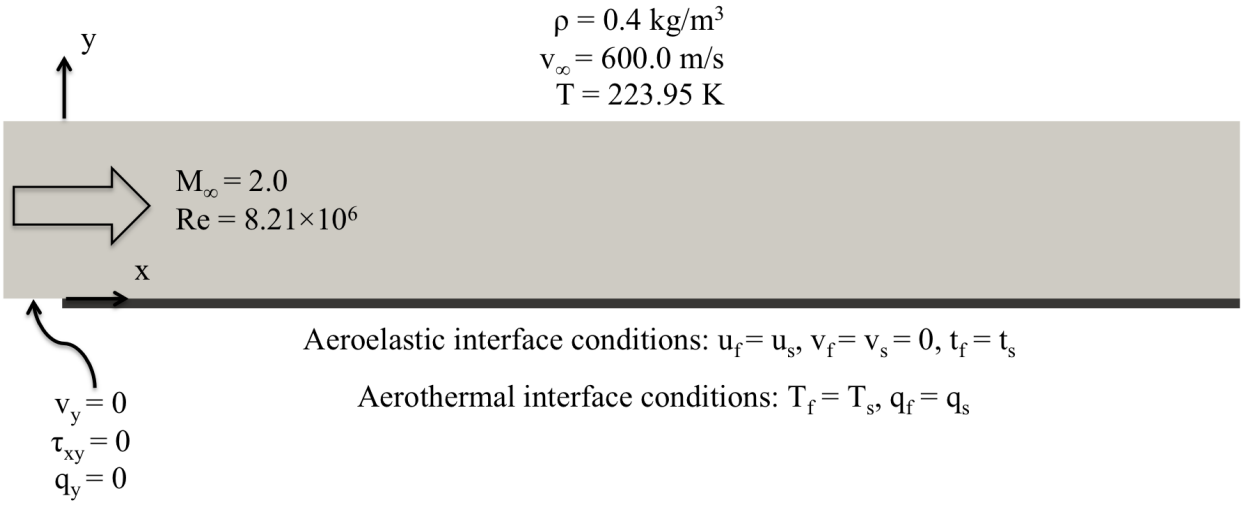



Figure 5.10: 2D flat plate aerothermoelastic problem geometry and boundary conditions.

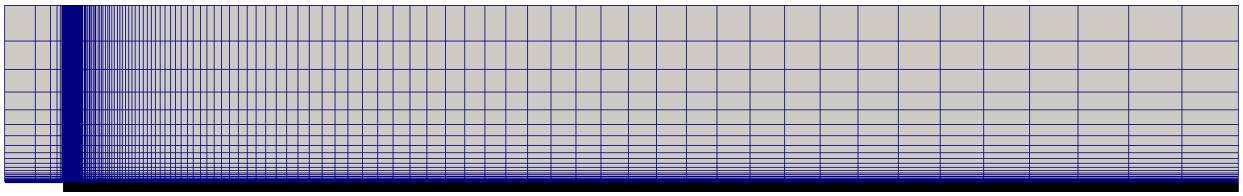


Figure 5.11: Two-dimensional flat plate aerothermoelastic problem fluid/solid mesh.

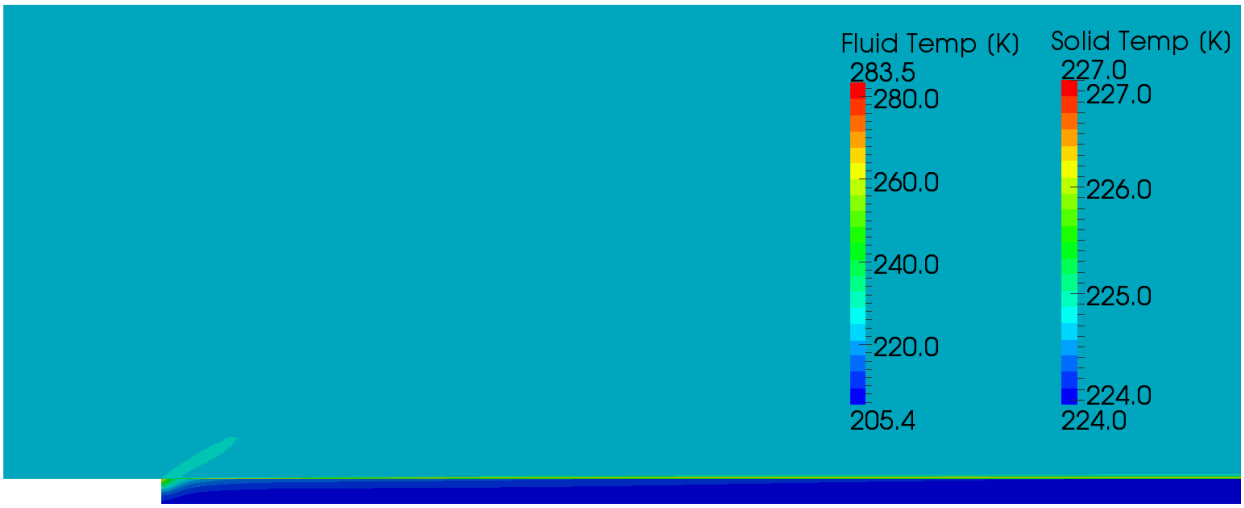



Figure 5.12: 2D flat plate aerothermoelastic temperature contours at 0.1 $s$.

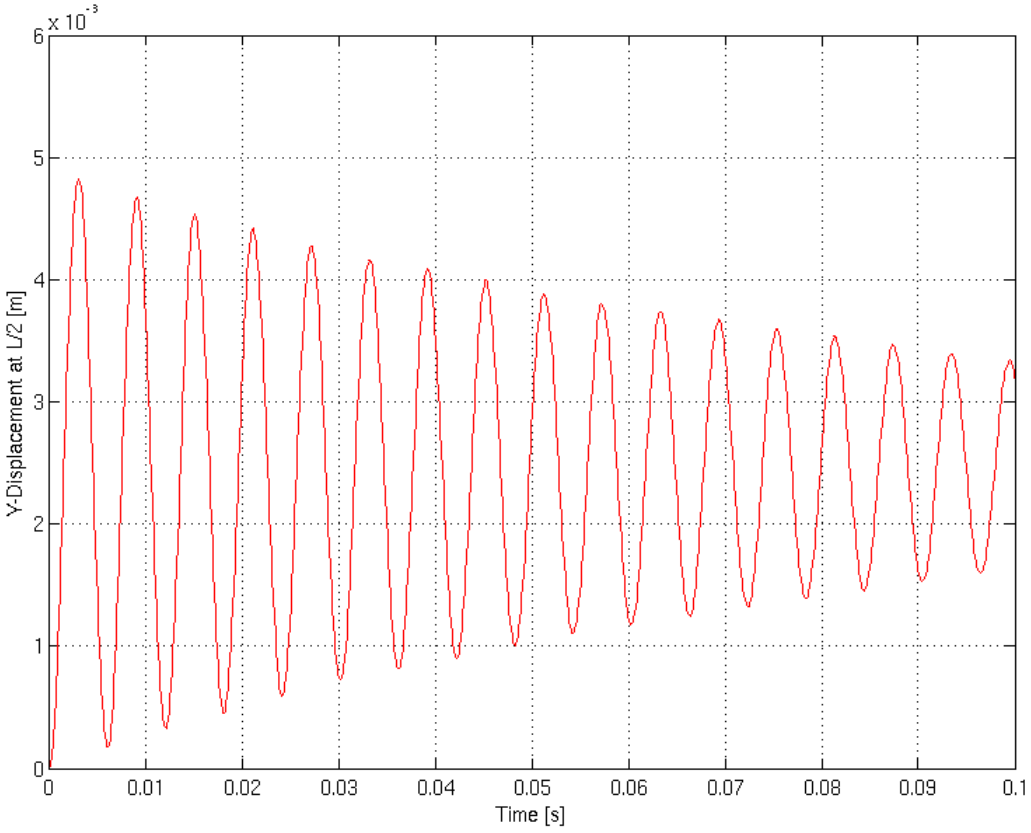



Figure 5.13: 2D flat plate aerothermoelastic displacements at x=L/2.

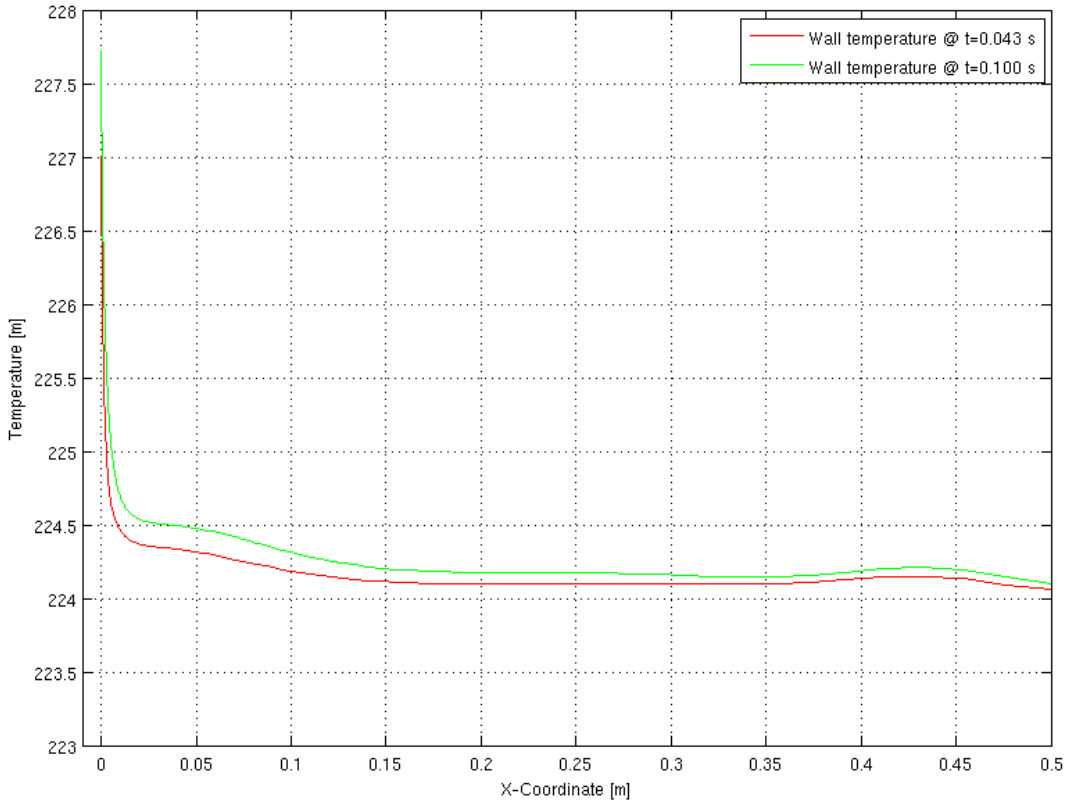



Figure 5.14: 2D flat plate aerothermoelastic wall temperature profiles.

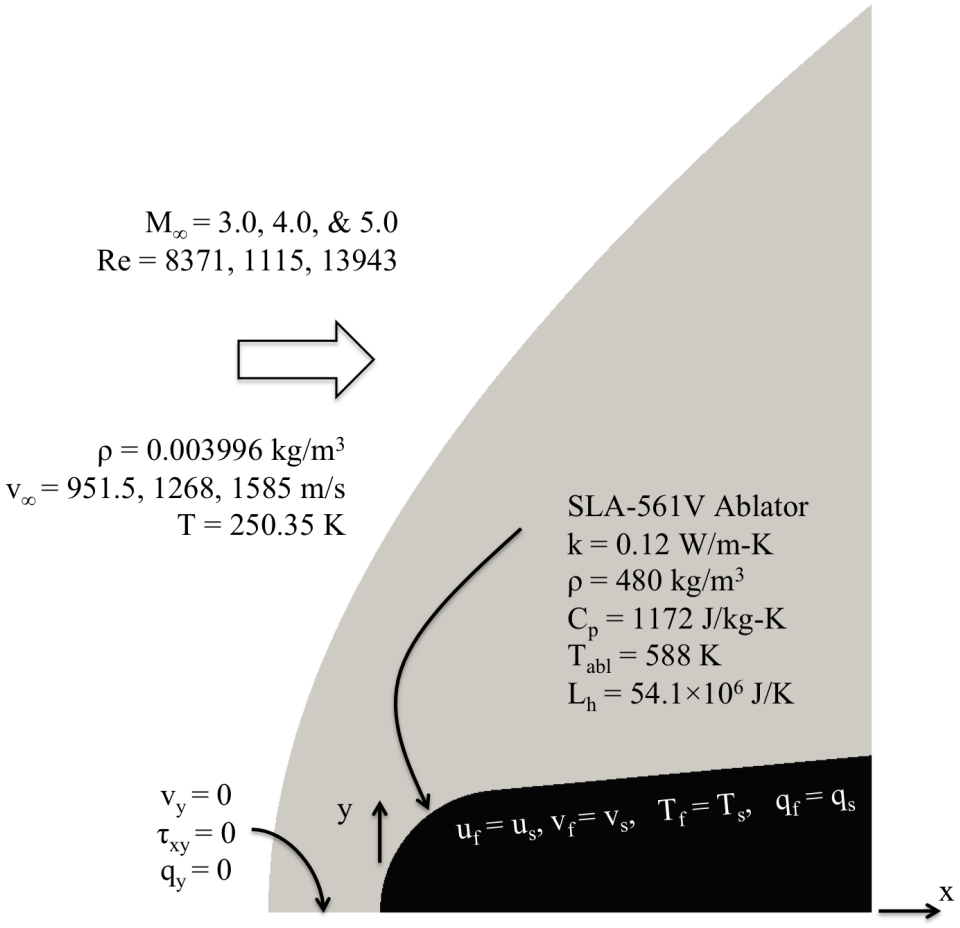



Figure 5.15: Two-dimensional nose tip ablation problem geometry and boundary conditions.

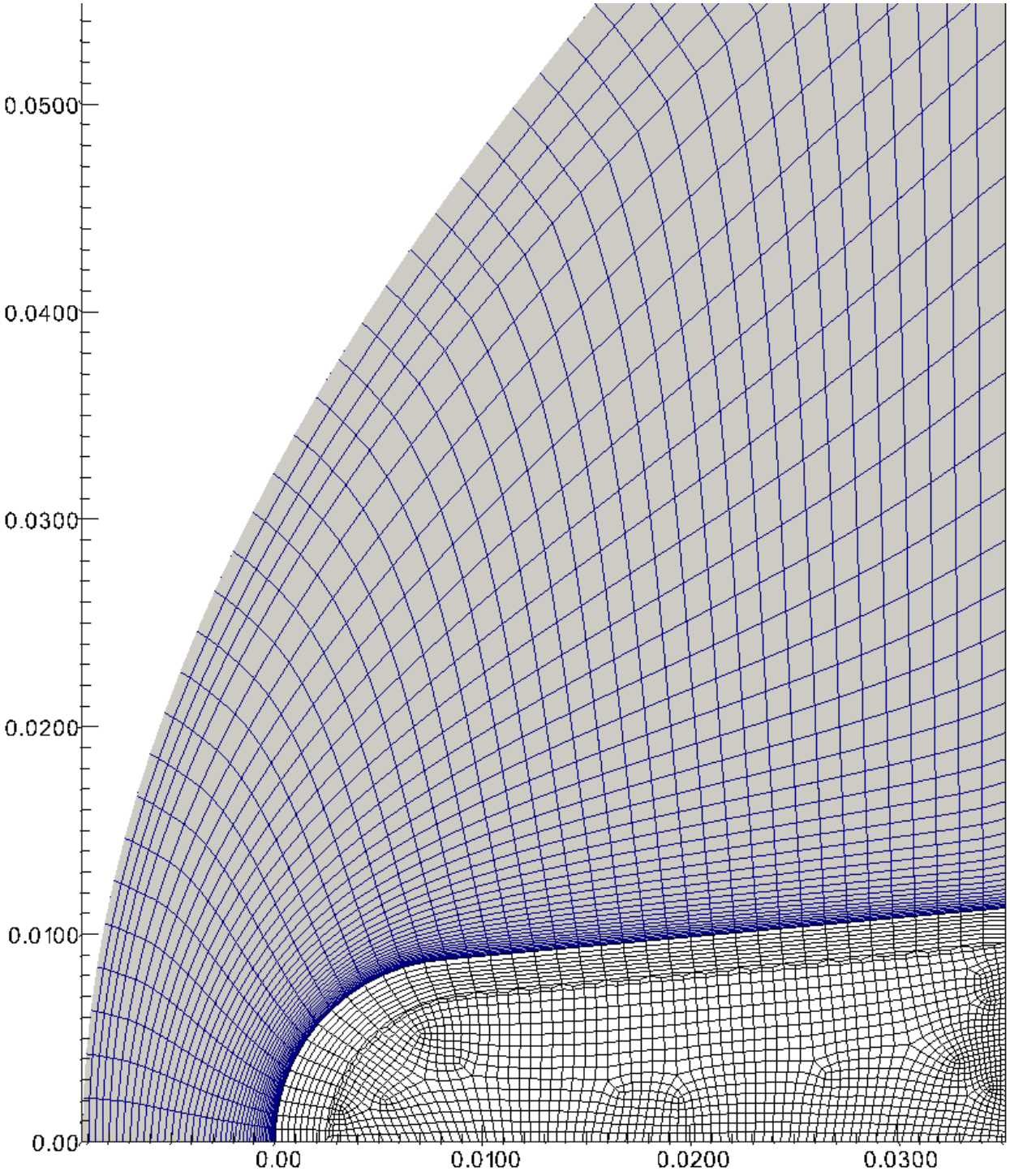



Figure 5.16: Two-dimensional nose tip ablation problem fluid/solid mesh.

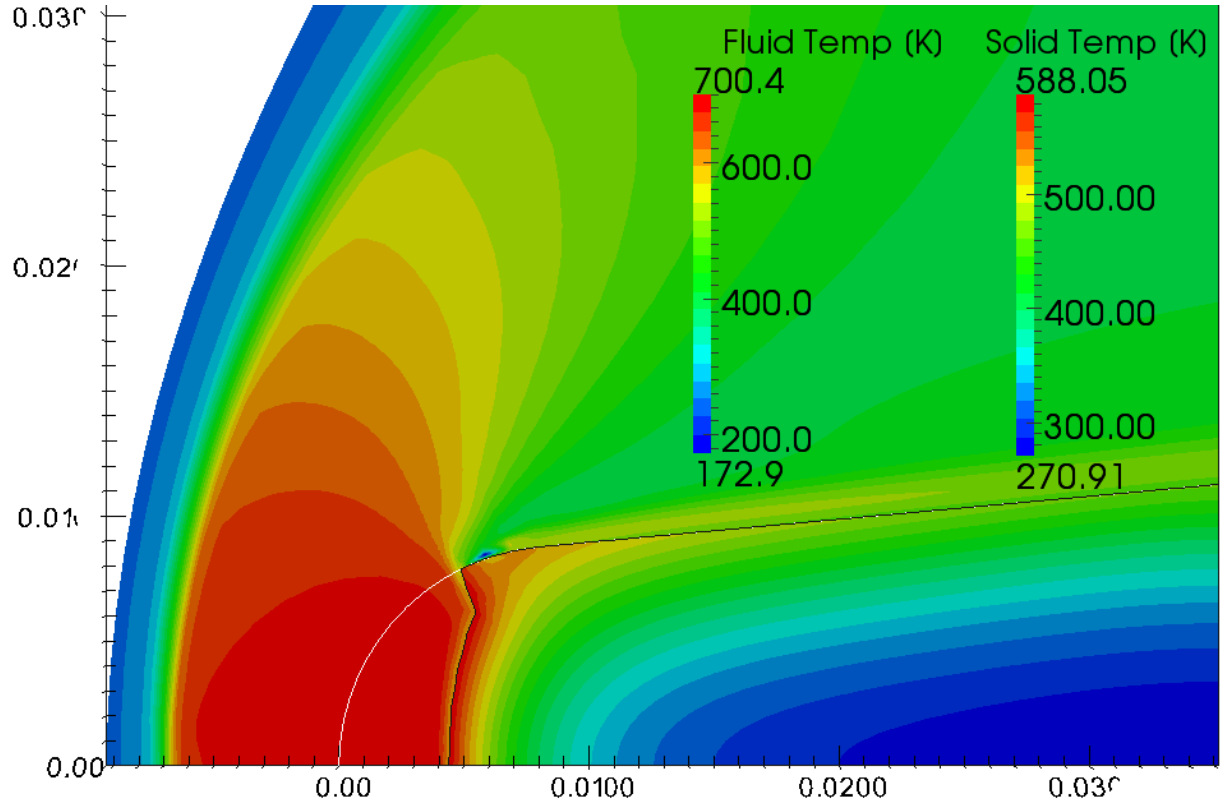



Figure 5.17: 2D nose tip ablation problem Mach 3 flow ablation response at 80 $s$.

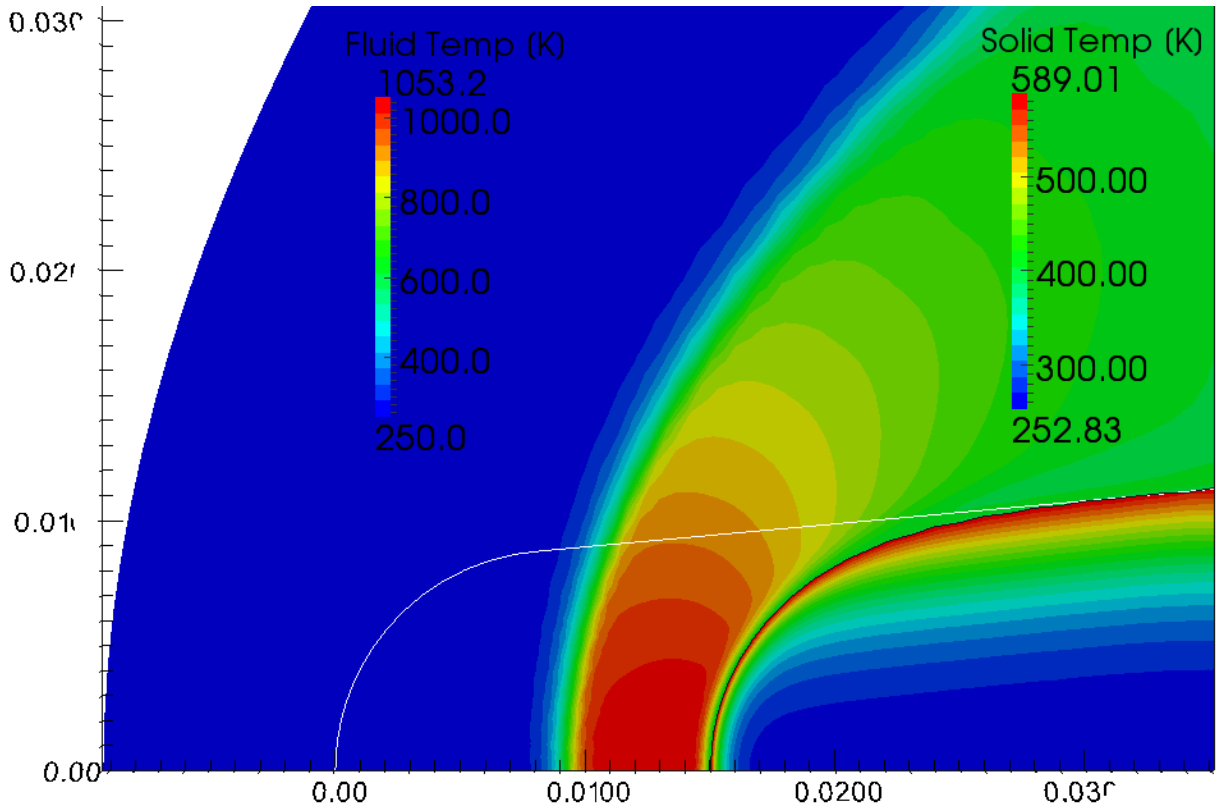



Figure 5.18: 2D nose tip ablation problem Mach 4 flow ablation response at 40 $s$.

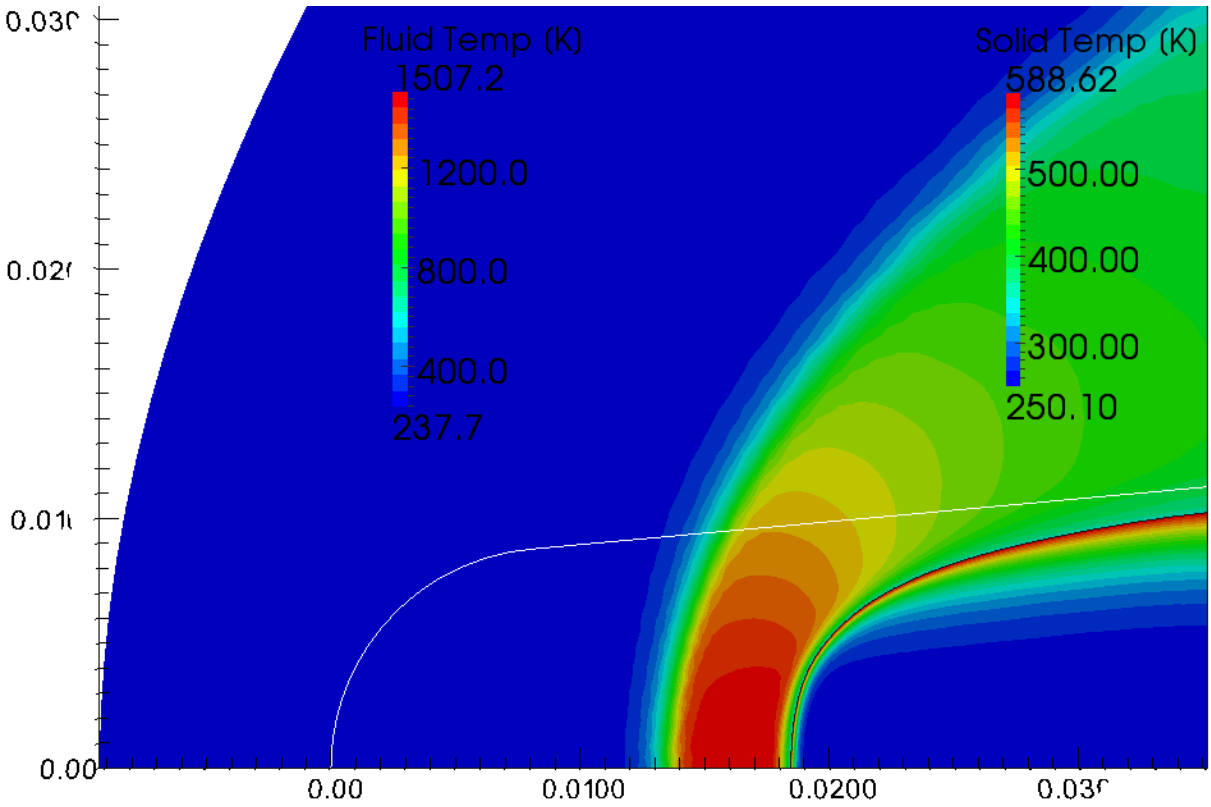



Figure 5.19: 2D nose tip ablation problem Mach 5 flow ablation response at 20 $s$.

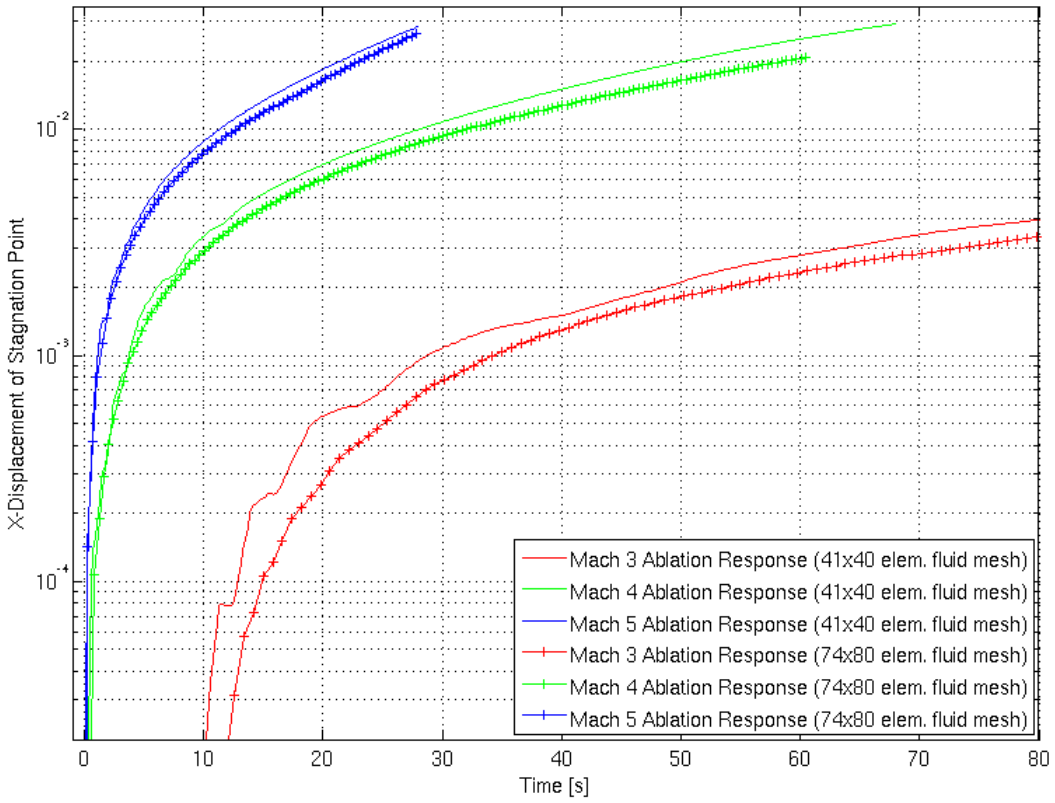



Figure 5.20: 2D nose tip ablation surface recession vs. time for Mach 3, 4, and 5 flows.

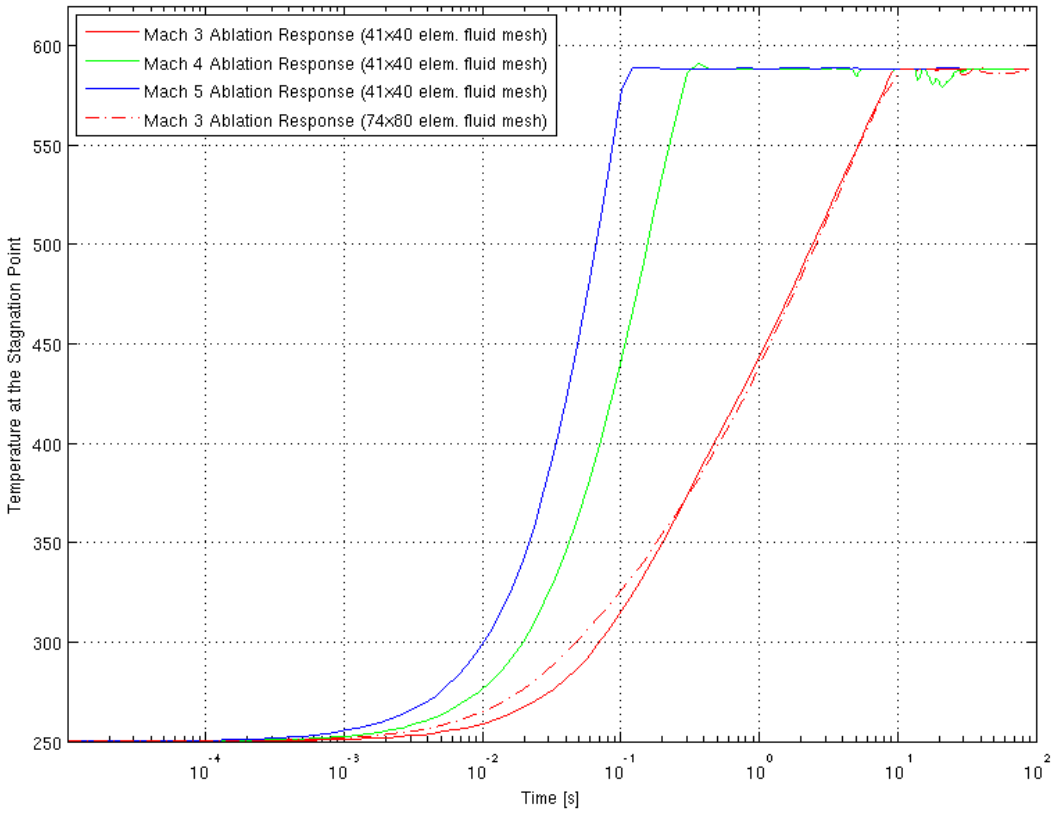



Figure 5.21: 2D nose tip ablation surface temperature vs. time for Mach 3, 4, and 5 flows.

# Chapter 6

# General Design Optimization Methods

## 6.1 Introduction

Design optimization has numerous real-world applications spanning many disciplines of engineering. Every design problem can be cast into the form of an optimization problem. However, the amount of improvement to the design that is realizable and the cost associated with setting up and solving the optimization problem often dictate whether optimization a beneficial step in the design process. Increasingly robust computational methods for engineering analysis, ever increasing computing power, and efficient algorithms for numerical optimization continue to reduce the overhead of design optimization though. The ability to quickly optimize a complex product is becoming a necessity for product development cycles; thus, it is no surprise that a wide variety of areas of design optimization have been the subject of intense research for some time.

The broad subject of design optimization can be divided into three categories. The following list represents the fundamental conceptual approaches to design optimization:

(1) *Sizing optimization* involves changing the parameters of the design such as material properties, thicknesses, cross-sections, or operating conditions as a means of achieving a better design. This method assumes that the shape and topology of the design are predefined, and merely operates on the parameters that define the design. To some extent the following two optimization techniques are variations of sizing optimization but are typically categorized separately as they are specializations of the sizing optimization concept.

(2) *Shape optimization* uses variables controlling the shape of the object to improve the design. Examples of shape optimization include changing the spatial positioning of truss joints to minimize member stresses or changing the shape of an airfoil as a means of increasing the lift/drag ratio. While the shape of the design is the subject of optimization, an initial design must be defined and the topology, or layout of the material and voids that constitute the design, is not subject to change.

(3) *Topology optimization*, also referred to as *material distribution* or *layout optimization*, is concerned with the placement of material within a domain to improve the design. Topology optimization is best thought of as determining the number, size, shape, and location of material and material voids to create an optimum design. This is the most general of the design optimization techniques, as it does not require a baseline design to begin with.

## 6.2 Mathematical Optimization

A design problem may be formulated as an optimization problem with an objective to be minimized or maximized with constraints that limit specific design criteria. The following discusses the general elements of design optimization. The general optimization problem is presented, mathematical optimality is defined, approaches to solving the design optimization problem are briefly laid out, and issues of shape and topology optimization relevant to this thesis are discussed.

### 6.2.1 Constrained Optimization Theory

In a general optimization problem, the objective function, $z$, is defined as the quantity to be minimized or maximized, and the constraints are divided into equality constraints, $h_j$, and inequality constraints, $g_k$. For the design optimization problems contained in this document the governing partial differential equations are discretized by a finite element model. This model is then used to evaluate objective and constraint values. The objective and constraint functions are expressed in terms of real valued abstract design variables, $s_i$, which are bounded by lower limits,

$s_i^L$, and upper limits, $s_i^U$. The general form of the optimization problem is expressed mathematically as

$$\begin{aligned} &\min_{s_i} z(s_i), \quad i = 1, ..., n_x \\ &h_j(s_i) = 0, \quad j = 1, ..., n_h \\ &g_k(s_i) \geq 0, \quad k = 1, ..., n_g \\ &s_i = \{s_i \in \mathbb{R}^{n_x} \quad | \quad s_i^L \leq s_i \leq s_i^U\} \end{aligned} \tag{6.1}$$

where $n_x$, $n_h$, and $n_g$ are the number of optimization variables, number of equality constraints, and the number of inequality constraints, respectively.

The optimization algorithms most often used for wide array of design optimization problems are classified as Lagrangian-based methods. These methods construct a primal-dual Lagrange function from the constrained optimization problem (equation 6.1). The Lagrange function is written as

$$L(s_i, \eta_j, \gamma_k) = z(s_i) + \sum_j^{n_j} \eta_j h_j(s_i) + \sum_k^{n_k} \gamma_k g_k(s_i) \tag{6.2}$$

where the Lagrange multipliers are defined as

$$\begin{aligned} \eta_j &\in \mathbb{R}^{n_h} \\ \gamma_k &\in \mathbb{R}^{n_k}. \end{aligned} \tag{6.3}$$

In this function the primal variables are the design variables $s_i$, and the dual variables are the Lagrange multipliers, $\eta_j$ and $\gamma_k$. Optimality of the primal-dual Lagrange function corresponds to the optimum of the objective function in primal space. Thus,

$$L(s_i^*, \eta_j^*, \gamma_k^*) = z(s_i^*) \tag{6.4}$$

where the $^*$ is used to indicate the optimal solution. This solution corresponds to minimizing the Lagrange function in the primal space and maximizing the Lagrange function in the dual space. Hence, the optimal solution is said to exist at the saddle point of the Lagrange function.

The $Karush - Kuhn - Tucker$ conditions establish criteria for satisfying optimality of equation 6.2 and for the general optimization problem (equation 6.1). The $KKT$ conditions are also known as first-order optimality conditions and are defined as

$$\frac{\partial L}{\partial s_i} \rightarrow \frac{\partial z}{\partial s_i}(s_i^*) + \sum_j^{n_h} \eta_j \frac{\partial h_j}{\partial s_i}(s_i^*) + \sum_k^{n_g} \gamma_k \frac{\partial g_k}{\partial s_i}(s_i^*) = 0 \tag{6.5}$$

$$\frac{\partial L}{\partial \eta_j} \rightarrow h_j(s_i^*) = 0 \tag{6.6}$$

$$\frac{\partial L}{\partial \gamma_k} \rightarrow \gamma_j^* g_j(s_i^*) = 0 \tag{6.7}$$

$$\gamma_j^* \geq 0. \tag{6.8}$$

Equation 6.5 states that the derivative of the objective function evaluated at the optimum should be equal and opposite to the sum the constraints multiplied by their respective Lagrange multipliers. The Lagrange multipliers act as scaling parameters for the constraints and give a sense of how active each constraint is at the optimum.

Equation 6.6 simply states that the equality constraints must be equal to zero at the optimum. This condition is directly seen in the statement of the general optimization problem given by equation 6.1. Note that equations 6.5 - 6.8 do not explicitly include the condition $g_k \geq 0$ in equation 6.1. The inequality conditions are encompassed by the complimentary slackness condition at the optimum (equation 6.7).

Equation 6.7 is the condition necessary to satisfy the inequality constraints, which may be either active or inactive at the optimum. To handle this condition the inequality constraints are rewritten as equality constraints by introducing slack variables $v_k$ such that

$$g_k + v_k^2 = 0 \tag{6.9}$$

where $v_k \in \mathbb{R}^{n_k}$.

Slack variables are additional variables added to the problem which ensure the inequality is always satisfied. Given this modification, the constraints can assume two states: $g_k = 0$, in which case the constraint is active and its associated slack variable $v_k$ must also equal zero, or $g_k < 0$,

in which case the constraint is inactive and its associated slack variable is greater than zero. The Lagrange function originally given by equation 6.2 must now be written as

$$L(s_i, \eta_j, \gamma_k, v_k) = z(s_i) + \sum_{j}^{n_j} \eta_j h_j(s_i) + \sum_{k}^{n_k} \gamma_k (g_k(s_i) + v_j(s_i)^2). \tag{6.10}$$

The first-order optimality condition with respect to $\gamma_j$ becomes

$$\frac{\partial L}{\partial \gamma_k} \rightarrow \gamma_k^* g_k(s_i^*) + v_k^2(s_i^*) = 0. \tag{6.11}$$

Since the slack variables were introduced to the Lagrange function, the partial derivative of the Lagrange function with respect to $v_k$ must be added to the set of first-order optimality conditions. Thus,

$$\frac{\partial L}{\partial v_k} \rightarrow 2\gamma_k v_k = 0. \tag{6.12}$$

Once again, the two states of an inequality constraint can be distinguished:

(1) If the constraint is active, that is $\gamma_k \neq 0$, then $g_k = 0$ and the slack variable $v_k = 0$

(2) If the constraint is inactive, then $\gamma_k = 0$, $g_k < 0$, and the slack variable $v_k > 0$

For both of these cases, the complimentary slackness condition of equation 6.7 is satisfied.

Finally, equation 6.8 must be satisfied, which states that the Lagrange multiplier for the inequality constraints must be greater than or equal to zero. The formal proof for this condition is quite involved and beyond the scope of this thesis. Please refer to Nocedal and Wright [111] for further information on the derivation of the $KKT$ conditions.

### 6.2.2 Design Optimization Methodologies

Several methods are available for solving design optimization problems. The difference between the methods is how the state variables of the governing equations and the optimization variables are handled. The nested analysis and design (NAND) and simultaneous analysis and design (SAND) methods described here are applicable to the single-discipline optimization problems of this thesis as well as multi-disciplinary optimization problems.

**Nested Analysis and Design** The traditional approach for design optimization is the NAND method, which assumes the equilibrium equations are satisfied independently from the optimization process. For each optimization iteration and the current set of optimization variables, $s_i$, the equilibrium state equations must be solved. Consequently, the structural state variables ($\boldsymbol{u}$) are written as an explicit function of the optimization variables, i.e. $\boldsymbol{u}(s_i)$.

Since the state variables are solved independently at each optimization iteration, existing analysis codes can be easily linked to the optimization process with little modification. The benefit of this method can be seen in multidisciplinary optimization problems. In the analysis of multidisciplinary systems, codes of different architectures, for instance finite element, finite volume, or finite difference solvers, may be used for the individual field problems and can be easily nested within the optimization iterations. The work presented in this thesis is based on the NAND optimization methodology.

**Simultaneous Analysis and Design** The SAND approach includes the governing equilibrium equations of the system with the equality constraints of the optimization problem. In this method, $\boldsymbol{u}$ and $s_i$ are both treated as independent variables. The advantage is that the state equations are satisfied only when the optimization problem converges, thus avoiding multiple solutions of the state equations. SAND formulations have been successfully used for optimization problems in structures [97], heat transfer [60], and aerodynamics [112].

This approach leads to a large optimization problem, and the solution process often becomes unwieldy. To mitigate this problem, reduction methods have been applied to the system [113]. The analysis and optimization routines must also be closely integrated, often requiring heavy modification of existing codes.

## 6.3 Topology Optimization

Topology optimization is concerned with optimizing the distribution of material to meet a specific design goal. The following is an overview of the techniques of topology optimization relevant to this thesis. Bendsøe and Sigmund [20] give a comprehensive treatment of the subject

from formulation to applications. Interested readers are referred to this source as well as the literature review by Eschenauer and Olhoff [36] for additional coverage of this topic.

### 6.3.1 Basic Concepts

In the context of finite element methods for structural analysis and optimization, a given design domain, $\Omega$, is discretized into finite elements. The topology optimization procedure assigns material indicator variables, $\chi$, that are most often associated with each element in the domain. The material indicator variables are used to indicate the presence of material or a void. Thus, if $\chi = 1$ this indicates the presence of material and if $\chi = 0$ this marks a material void. In order to ensure that the objective function and constraints predicted by the finite element model are continuous, $\chi$ is allowed to vary continuously between zero and one. By introducing a continuously varying material, the problem now becomes a optimization problem that seeks to find an optimal layout of material properties, and not the optimal layout of a single, homogenous material. Clearly, a material with continuously varying material properties, for instance a varying density, is difficult to produce. This results from the fact that the material becomes the subject of optimization and not the layout of a material with fixed properties. Several solution techniques are used to circumvent these issues and are discussed next.

### 6.3.2 Solution Strategies

#### Material Penalization

An approach known as simple isotropic material with penalization (SIMP) [18, 19] is a common solution to the continuous material parameter problem in topology optimization. This method penalizes intermediate values of the material properties in the optimization problem. Thus, for an arbitrary material parameter, $\eta_0$, for example elastic modulus or coefficient of thermal expansion, the material indicator is raised to a power $\beta$

$$\eta = s_i^{\beta}\eta_0, \quad \epsilon < s_i \leq 1 \tag{6.13}$$

where $\epsilon$ is a small but non-zero value that is used to avoid singularities and the abstract optimization variables $s_i$ are used as the material indicators $\chi$. The SIMP model is usually used in conjunction with a mass constraint. The density of the material, $\rho_0$, is allowed to vary linearly with the abstract variables according to the equation

$$\rho = s_i \rho_0, \quad \epsilon < s_i \leq 1. \tag{6.14}$$

The discrepancy between the property penalization and the density variation forces more clearly defined material and void distributions. With this approach, an intermediate material value adds a considerable negative affect in improving the objective function over a material indicator value near zero or one.

**Filtering**

Problems in topology optimization often do not converge to a unique solution upon successive mesh refinement. The reason is due to the discretization of the design domain and the material penalty formulation used. As the mesh is refined, an increasing number of possibilities for the size and location of material voids becomes possible. Another phenomenon know as checkerboards is also observed in problems solved by topology optimization. Checkerboards are characterized by alternating, adjacent material and void elements. While the checkerboard problem is reduced by the use of higher order elements, these type of numerical instabilities may be mitigated by employing a technique known as filtering. Filtering makes the design gradient of an element dependent on the neighboring elements lying within a specified radius, and effectively smooths out spatial oscillations in the design sensitivities. The derivatives of a function with respect to the design variables, $\partial f / \partial s_i$, are filtered according to the equation

$$\frac{\partial \tilde{f}}{\partial x_k} = (x_k)^{-1} \frac{1}{\sum w_{ik}} \sum w_{ik} x_i \frac{\partial f}{\partial s_i}. \tag{6.15}$$

The weighting function $w_{ik}$ is expressed as

$$w_{ik} = max\,(r - d_{ik}, 0) \tag{6.16}$$

where $r$ is the filter radius and $d_{ik}$ is the distance between the centers of the $i^{th}$ and $k^{th}$ elements.

The reader is referred to the works of Sigmund [127, 128] for more details on numerical instabilities and filtering.

## 6.4 Sensitivity Analysis

The gradient-based optimizers used in this work require derivative information of the optimization criteria. The gradients $\partial z/\partial s_i$, $\partial h/\partial s_i$, and $\partial g/\partial s_i$ must be computed and sent to the numerical optimization routine to solve the optimization problem of equation 6.1. The process of computing the gradient values is known as sensitivity analysis, and is often a very computationally intensive task for structural optimization problems. Three forms of computing the sensitivities are commonly employed: numerical sensitivity analysis, analytical sensitivity analysis, and automatic differentiation,.

### 6.4.1 Numerical Sensitivity Analysis

Numerical sensitivity analysis uses finite difference methods to compute gradient information. While easy to implement, numerical sensitivity analysis has several drawbacks. The first of which is computational cost. If a second-order accurate central difference scheme is used, a forward and backward perturbed analysis of the system model must be performed for each optimization variable. This cost becomes severely limiting if the analysis time is lengthy and the number of optimization variables becomes large. Furthermore, numerical sensitivity analysis is error prone, as the selection of a finite differencing perturbation and the severity of nonlinearities in the criteria may lead to inaccurate gradients. The complex step method of numerical sensitivity analysis provides more accurate results for small perturbation sizes, but its implementation is more involved than classical finite difference methods. Due to the shortcomings of numerical sensitivity analysis analytical methods have become increasingly attractive because they decrease computational time and increase robustness. Nonetheless, numerical sensitivity approaches serve an important purpose, as they are often used for verifying analytical sensitivities and when the overhead related to

developing analytical sensitivity methods is not justified.

### 6.4.2 Analytical Sensitivity Analysis

The analytical sensitivity analysis method determines derivatives of the objective or constraint functions based on analytical derivatives of the discrete equations. As one might imagine, the time involved with developing analytical gradient methods is often extensive. However, for most problems significant returns can be expected in terms of reduced computational cost of the sensitivity procedure.

The aerothermoelastic system is governed by a set of residual equations expressed in generic form as

$$\boldsymbol{R}(\boldsymbol{s}, \boldsymbol{U}(\boldsymbol{s}), \boldsymbol{T}(\boldsymbol{s}), \boldsymbol{u}(\boldsymbol{s})) = 0. \tag{6.17}$$

where $\boldsymbol{s}$ are design variables, $\boldsymbol{U}$ is the vector or conservative state variables for the fluid equations, $\boldsymbol{T}$ is the vector of temperatures for the heat equation, and $\boldsymbol{u}$ is the vector of state variables (nominally displacements) for the elasticity equation.

The residual and state vector may also be expressed of as an aggregation of discipline-specific residual and state vectors in the following way

$$\boldsymbol{R}^T = \left[\boldsymbol{R}^f(\boldsymbol{U}(\boldsymbol{s})) \ , \ \boldsymbol{R}^t(\boldsymbol{T}(\boldsymbol{s})) \ , \ \boldsymbol{R}^s(\boldsymbol{u}(\boldsymbol{s}))\right] = 0 \tag{6.18}$$

and

$$\mathcal{U}^T = [\boldsymbol{U}(\boldsymbol{s}) \ , \ \boldsymbol{T}(\boldsymbol{s}) \ , \ \boldsymbol{u}(\boldsymbol{s})] \tag{6.19}$$

Using the NAND optimization methodology, the design criteria, $q_j$, are defined as the set of objective and constraint functions, and are expressed as

$$q_j = q_j(\boldsymbol{s}, \boldsymbol{U}(\boldsymbol{s}), \boldsymbol{T}(\boldsymbol{s}), \boldsymbol{u}(\boldsymbol{s})). \tag{6.20}$$

In this equation, $\boldsymbol{s}$ are the design variables and $\boldsymbol{U}(\boldsymbol{s})$, $\boldsymbol{T}(\boldsymbol{s})$, $\boldsymbol{u}(\boldsymbol{s})$ indicates the state variables are a function of the design variables.

**DIRECT METHOD** We may then represent the total derivative of the criteria $q_j$ with respect to the optimization variables $\boldsymbol{s}$ by the equation

$$\frac{dq_j}{d\boldsymbol{s}} = \frac{\partial q_j}{\partial \boldsymbol{s}} + \frac{\partial q_j}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{s}} + \frac{\partial q_j}{\partial \boldsymbol{T}}\frac{\partial \boldsymbol{T}}{\partial \boldsymbol{s}} + \frac{\partial q_j}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{s}}. \tag{6.21}$$

From here the partial derivatives of the criteria with respect to the abstract optimization variables $\partial q_j/\partial \boldsymbol{s}$ and the criteria with respect to the state variables (e.g. $\partial q_j/\partial \boldsymbol{U}$) may be evaluated directly by the analysis model. In order to obtain the derivative of the state variables with respect to the design variables (e.g. $\partial \boldsymbol{U}/\partial \boldsymbol{s}$) we must differentiate the governing equations (equation 6.17) with respect to the design variables. Since we require $\boldsymbol{R} = 0$, its derivative must also be equal to zero, hence

$$\frac{d\boldsymbol{R}}{d\boldsymbol{s}} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{s}} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\frac{d\boldsymbol{U}}{d\boldsymbol{s}} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{T}}\frac{d\boldsymbol{T}}{d\boldsymbol{s}} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{u}}\frac{d\boldsymbol{u}}{d\boldsymbol{s}} = 0. \tag{6.22}$$

Using the definition of residual vector given by equation 6.18, the previous equation may be written as

$$\underbrace{\begin{bmatrix} \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{U}} & \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{T}} & \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{u}} \\ \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{U}} & \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{T}} & \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{u}} \\ \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{U}} & \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{T}} & \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{u}} \end{bmatrix}}_{\mathcal{A}} \begin{bmatrix} \frac{d\boldsymbol{U}}{d\boldsymbol{s}} \\ \frac{d\boldsymbol{T}}{d\boldsymbol{s}} \\ \frac{d\boldsymbol{u}}{d\boldsymbol{s}} \end{bmatrix} = - \begin{bmatrix} \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{s}} \\ \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{s}} \\ \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{s}} \end{bmatrix} \tag{6.23}$$

Solving the above equation we can now compute the derivatives of the design criteria by

$$\frac{dq_j}{d\boldsymbol{s}} = \frac{\partial q_j}{\partial \boldsymbol{s}} - \begin{bmatrix} \frac{\partial q_j}{\partial \boldsymbol{U}} \\ \frac{\partial q_j}{\partial \boldsymbol{T}} \\ \frac{\partial q_j}{\partial \boldsymbol{u}} \end{bmatrix}^T \begin{bmatrix} \frac{d\boldsymbol{U}}{d\boldsymbol{s}} \\ \frac{d\boldsymbol{T}}{d\boldsymbol{s}} \\ \frac{d\boldsymbol{u}}{d\boldsymbol{s}} \end{bmatrix} \tag{6.24}$$

**ADJOINT METHOD** The adjoint method also begins from the derivatives of the optimization criteria

$$\frac{dq_j}{d\boldsymbol{s}} = \frac{\partial q_j}{\partial \boldsymbol{s}} + \frac{\partial q_j}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{U}}{\partial \boldsymbol{s}} + \frac{\partial q_j}{\partial \boldsymbol{T}}\frac{\partial \boldsymbol{T}}{\partial \boldsymbol{s}} + \frac{\partial q_j}{\partial \boldsymbol{u}}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{s}}. \tag{6.25}$$

The adjoint problem may then be defined as

$$\underbrace{\begin{bmatrix} \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{U}} & \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{U}} & \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{U}} \\ \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{T}} & \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{T}} & \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{T}} \\ \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{u}} & \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{u}} & \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{u}} \end{bmatrix}}_{\mathcal{A}^T} \begin{bmatrix} \boldsymbol{a}^f \\ \boldsymbol{a}^t \\ \boldsymbol{a}^s \end{bmatrix} = \begin{bmatrix} \frac{\partial q_j}{\partial \boldsymbol{U}} \\ \frac{\partial q_j}{\partial \boldsymbol{T}} \\ \frac{\partial q_j}{\partial \boldsymbol{u}} \end{bmatrix} \tag{6.26}$$

Solving the above equation we can now compute the adjoint vectors and compute the derivatives of the criteria with respect to the design variables according to the equation

$$\frac{dq_j}{d\boldsymbol{s}} = \frac{\partial q_j}{\partial \boldsymbol{s}} - \begin{bmatrix} \boldsymbol{a}^f \\ \boldsymbol{a}^t \\ \boldsymbol{a}^s \end{bmatrix}^T \begin{bmatrix} \frac{\partial \boldsymbol{R}^f}{\partial \boldsymbol{s}} \\ \frac{\partial \boldsymbol{R}^t}{\partial \boldsymbol{s}} \\ \frac{\partial \boldsymbol{R}^s}{\partial \boldsymbol{s}} \end{bmatrix} \tag{6.27}$$

There are two approaches to solving the total sensitivity equation and the choice of which is dictated by the number of optimization criteria, $q_j$, and optimization variables, $s_i$.

**Direct Method**

If the number of optimization variables is less than the number of optimization criteria the direct approach is more efficient. This method proceeds according to the following two steps:

(1) Compute the derivatives of the state variables with respect to the optimization variables to obtain the direct sensitivity solution.

(2) Compute the total sensitivity by evaluating equation 6.21.

This approach requires the solution of a linear system of equations for each optimization variable, however the evaluation direct sensitivity equations requires the same order of computational time as the evaluation of the system equations (equation 6.17). As such, the direct method becomes very costly with an increasing number of optimization variables.

**Adjoint Method**

If the number of optimization criteria is less than the number of optimization variables the adjoint approach is more efficient. The adjoint method progresses as follows:

(1) Compute the derivatives of the optimization criteria with respect to the state variables to obtain the adjoint sensitivity solution. This is accomplished by evaluating the following equation

$$\boldsymbol{a}_j = \left[\frac{\partial \boldsymbol{R}}{\partial(\boldsymbol{U}, \boldsymbol{T}, \boldsymbol{u})}\right]^{-T} \frac{\partial q_j}{\partial(\boldsymbol{U}, \boldsymbol{T}, \boldsymbol{u})} \tag{6.28}$$

(2) Compute the total sensitivity by evaluating

$$\frac{dq_j}{ds_i} = \frac{\partial q_j}{\partial s_i} - \boldsymbol{a}_j \frac{\partial \boldsymbol{R}}{\partial s_i} \tag{6.29}$$

This approach requires the solution of a linear system of equations for each optimization criteria, $q_j$. The adjoint method eliminates the dependency on the number of optimization variables and thus is the preferred method if the number of optimization variables exceeds the number of criteria.

The problems solved in this thesis, and for topology optimization problems in general, contain a large number of optimization variables and a limited number of criteria. As such, the adjoint method of sensitivity analysis is used here.

### 6.4.3 Automatic Differentiation

Automatic differentiation is a computational tool for performing analytical sensitivity analysis based on discretized equation forms. An automatic differentiator operates on an existing discrete equation routine, or set of routines, and produces a code that computes the derivative of the original routine's output. Automatic differentiation applies the chain rule to the base code's commands with respect to the input variables. For instance, if a routine computes the stiffness matrix and body force vector of an element, and is automatically differentiated, the output code will compute the derivatives of the stiffness matrix and body force vector. It is a quick way of generating derivative functions, especially if the computational routines that must be differentiated are complex. Well-refined and tested implementations for the Fortran 77 (ADIFOR [21]) and C/C++ (ADIC [22] and ADOL-C [50]) programming languages exist. While automatic differentiation is a quick and easy tool for computing derivatives of functions, it removes the user from any physical insight into the problem and often decreases computational efficiency.

## 6.5 Numerical Optimization

Numerical optimization algorithms are primarily classified according to two types: discrete and continuous optimization routines. Discrete optimizers seek to find the combination of variables, often integer values, to minimize an objective. Popular methods of discrete-based optimization are branch-and-bound methods, genetic algorithms, and simulated annealing. In contrast, continuous optimization allows the variables to assume real values within their bounds and assumes a continuous objective function. A subset of continuous optimization is gradient-based optimization where the derivatives of the criteria are assumed to exist and these derivatives are required by the optimizer.

Within continuous optimization, constrained and unconstrained methods exist. Constrained optimization is widely used for engineering applications. This is due to the fact that most engineering problems are formulated with bounds on the design, for example constraints on mass, stress, or eigenfrequencies.

The partial differential equations that govern most engineering systems result in nonlinear objective functions and constraints. Consequently, this work employs gradient-based algorithms for nonlinearly constrained optimization. The two major branches for general nonlinear constrained optimization algorithms include interior point methods and sequential quadratic programming (SQP). The method of moving asymptotes (MMA) and a follow on method, known as sequential convex programming (SCP), have appeared recently and were formulated in the context of structural optimization problems. These two methods have also been shown to work well for general optimization problems [145]. This thesis uses the popular SQP optimizers by Gill ([48]) and Schittkowski ([122]) and the structural optimization inspired MMA and SCP algorithms due to Svanberg ([132]) and Zillober ([144]).

### 6.5.1 Sequential Quadratic Programming

The sequential quadratic programming method constructs and solves a series of quadratic approximations to the objective and constraint functions. SQP is appropriate for large problems with significant nonlinearities, and is very effective for structural optimization problems.

The algorithm applies Newton's method to the system of KKT conditions. The step size of the optimization variables can be computed by using the familiar Newton linearization approach discussed in section 2.4.1 but applied to the KKT system. The first order $KKT$ conditions are defined in vector form as

$$\boldsymbol{R} = \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{s}} \\ \boldsymbol{h} \\ \boldsymbol{\gamma^T g} \end{bmatrix} \tag{6.30}$$

and the Jacobian of the $KKT$ vector is expressed as

$$\mathbf{J} = \begin{bmatrix} \frac{\partial^2 L}{\partial \boldsymbol{s}^2} & \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s}} & \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{s}} \\ \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s}} & 0 & 0 \\ \boldsymbol{\gamma^T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{s}} & 0 & \boldsymbol{g} \end{bmatrix}. \tag{6.31}$$

Thus, the Newton step can be computed by solving the system of $KKT$ equations given by

$$\begin{bmatrix} \frac{\partial^2 L}{\partial \boldsymbol{s}^2}^n & \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s}}^n & \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{s}}^n \\ \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s}}^n & 0 & 0 \\ \boldsymbol{\gamma^T} \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{s}}^n & 0 & \boldsymbol{g}^n \end{bmatrix} \begin{bmatrix} \triangle \boldsymbol{s}^n \\ \triangle \boldsymbol{\eta}^n \\ \triangle \boldsymbol{\gamma}^n \end{bmatrix} = \begin{bmatrix} \frac{\partial L}{\partial \boldsymbol{s}}^n \\ \boldsymbol{h}^n \\ \boldsymbol{\gamma^T g}^n \end{bmatrix} \tag{6.32}$$

and the new optimization iterate $(s^{n+1}, \eta^{n+1}, \gamma^{n+1})$ is given by

$$\begin{bmatrix} s^{n+1} \\ \eta^{n+1} \\ \gamma^{n+1} \end{bmatrix} = \begin{bmatrix} s^n \\ \eta^n \\ \gamma^n \end{bmatrix} + \begin{bmatrix} \triangle s^n \\ \triangle \eta^n \\ \triangle \gamma^n \end{bmatrix} \tag{6.33}$$

Equation 6.32 is well-posed if the constraint derivatives $\partial \boldsymbol{h}/\partial \boldsymbol{x}$ and $\partial \boldsymbol{g}/\partial \boldsymbol{x}$ are linearly independent and if the Hessian of the Lagrange function $\partial^2 L/\partial \boldsymbol{x}^2$ is positive definite.

A unique property of the KKT system is that the iterate step sizes ($\triangle \boldsymbol{x}^n, \triangle \boldsymbol{\eta}^n, \triangle \boldsymbol{\gamma}^n$) can be computed by either solving the system of equations (equation 6.32) or by finding the solution to an equivalent quadratic minimization problem. This characteristic of the problem requires either the use of Newton's method or a method for quadratic minimization to compute the next iterate. The corresponding quadratic minimization problem is posed as

$$\min_s \frac{1}{2} (\triangle \boldsymbol{s}^n)^T \frac{\partial^2 L}{\partial \boldsymbol{s}^2}^n \triangle \boldsymbol{s}^n + \left(\frac{\partial z}{\partial \boldsymbol{s}}^n\right)^T \triangle \boldsymbol{x}^n \tag{6.34}$$

$$\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{s}}^n \triangle \boldsymbol{s}^n + \boldsymbol{h}^n \geq \boldsymbol{0} \tag{6.35}$$

$$\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{s}}^n \triangle \boldsymbol{s}^n + \boldsymbol{g}^n \geq \boldsymbol{0} \tag{6.36}$$

Computation of the Hessian $\partial^2 L / \partial \boldsymbol{s}^2$ is a computationally expensive procedure and may not always yield a positive definite matrix. The exact Hessian is often replaced by a quasi-Newton approximation, which uses first order updates to a approximate the Hessian. The Broydon-Fletch-Goldfarb-Shanno (BFGS) update is the most popular of all Hessian approximation methods. For more information on the solution of the quadratic minimization problem (equations 6.34 - 6.36) and quasi-Newton methods including the BFGS update, refer to Nocedal and Wright [111].

### 6.5.2 Method of Moving Asymptotes

The Method of Moving Asymptotes (MMA) was first introduced by Svanberg [132] and has since become a mainstay of structural optimization and topology optimization in particular. The method is similar to sequential quadratic programming (SQP) methods in that it seeks to solve the optimization problem by constructing and solving a sequence of easier to solve subproblems or local models at each iterate. The difference between MMA and SQP is how the subproblem is defined. MMA takes advantage of a special characteristic of structural problems. Many structural constraints, such as material stresses, are directly dependent on structural displacements and are exact linearizations of the original functions with respect to inverse variables. It should be noted that the primary output of the finite element model, which determines the objective and constraint values, are structural displacements. As such, nearly any quantity that might be subject

to constraint, such as stresses, strain energy, mass, or displacements, will be dependent on the displacement solution.

The MMA algorithm constructs a strictly convex separable approximation to the objective and inequality constraints by linearizing with respect to transformed inverse variables defined as

$$\frac{1}{s_i^n - L_i^n}, \quad \frac{1}{U_i^n - s_i^n}. \tag{6.37}$$

The choice of which inverse variable type to linearize against is determined by the sign of the function derivative at the current iterate. The values of the lower and upper asymptotes $L_i$ and $U_i$, which represent bounds for the local model, are permitted to change between iterations and are often referred to as "moving asymptotes." Selection of the $L_i$ and $U_i$ parameters is an important part of MMA and is discussed below.

At each iteration the objective function is approximated for the current set of design variables $s_i^n$ by linearizing over the inverse variables according to the equation

$$\tilde{z}\left(s_i^n\right) = z\left(s_i^n\right) + \sum_{i,+}\left[\frac{\partial z}{\partial s_i}|_{x^n}\left(\frac{(U_i - s_i^n)^2}{U_i - s_i} - (U_i - s_i^n)\right) + \tau_i \frac{(s_i - s_i^n)}{U_i - s_i}\right] - \sum_{i,-}\left[\frac{\partial z}{\partial s_i}|_{x^n}\left(\frac{(s_i^n - L_i)^2}{s_i - L_i} - (s_i - L_i)\right) - \tau_i \frac{(s_i - s_i^n)}{s_i - L_i}\right]. \tag{6.38}$$

The original form of the MMA algorithm cannot explicitly handle equality constraints so they will not be considered here. The inequality constraints are linearized with respect to the inverse variables as

$$\tilde{g_k}\left(x\right) = g_k(x^n) + \sum_{i,+}\left.\frac{\partial g_k}{\partial s_i}\right|_{x^n}\left(\frac{(U_i - s_i^n)^2}{U_i - s_i} - (U_i - s_i^n)\right) - \sum_{i,-}\left.\frac{\partial g_k}{\partial s_i}\right|_{x^n}\left(\frac{(s_i^n - L_i)^2}{s_i - L_i} - (s_i^n - L_i)\right) \tag{6.39}$$

The term $\sum_{i,+}$ indicates the linearization with respect to the inverse variables $1/(U_i^n - s_i)$ and the summation of the resulting components for non-negative partial derivatives evaluated at $x^n$. The term $\sum_{j,-}$ indicates the linearization with respect to the inverse variables $1/(s_i - L_j^n)$ and the summation of the resulting terms for negative partial derivatives evaluated at $x^n$. The $\tau_i$ terms in the objective function are positive parameters chosen to ensure that the approximation is strictly convex. Note that values of the moving asymptotes are chosen such that $L_i^n < s_i < U_i^n$.

The subproblem for the current iterate can now be defined in terms of the linearized functions

$$\min_{s_i} \tilde{z}(s_i), \quad i = 1, ..., n_x \tag{6.40}$$

$$\tilde{g}_k(s_i) \geq 0, \quad k = 1, ..., n_g$$

$$s_i^{'L} \leq s_i \leq s_i^{'U}$$

where the box constraint bounds are defined as

$$s_i^{'L} = max\left(s_i^L, a \cdot s_i^n + b \cdot L_i^n\right) \tag{6.41}$$

$$s_i^{'U} = min\left(s_i^U, a \cdot s_i^n + b \cdot U_i^n\right) \tag{6.42}$$

and the values of $a$ and $b$ are chosen and fixed between 0 and 1. The purpose of the "move limits" $a$ and $b$ is to keep the variables away from the asymptotes to avoid computing a very large objective and gradient value.

The subproblem defined by equation 6.40 has several desirable characteristics which result from the convex approximation made by the method of moving asymptotes. Specifically, the objective $\tilde{z}$ and constraints $\tilde{g}_k$ are first-order approximations of the original objectives and constraints at $x^n$. The objective $\tilde{z}$ is strictly convex and inequality constraints $\tilde{j_k}$ are also convex, but most importantly all of the functions are now separable.

The moving asymptotes $L_i$ and $U_i$ have been shown to effectively control the behavior of the subproblem functions [132]. When $L_i^n$ and $U_i^n$ are chosen close to the variable $s_i^n$, the second derivative will increase and thus so will the curvature of the function approximation. If $L_i^n$ and $U_i^n$ are chosen far from the variables $s_i^n$, then the second derivative decreases, the curvature decreases, and the function approximation becomes close to linear. As $L_i^n \to -\infty$ and $U_i^n \to \infty$ the objective and inequality constraint approximations become linear. For example, the objective becomes

$$\tilde{z}(s) = z(s^n) + \sum_i \left.\frac{\partial z}{\partial s_i}\right|_{s^n} (s_i - s_i^n) \tag{6.43}$$

which is recognized as the sequential linear programming (SLP) method. Rules for selecting the moving asymptotes are largely heuristic. The rules given by Svanberg in his original MMA paper [132] for choosing the asymptotes are:

(1) If the iterates oscillate, the problem is likely to become more stable if the asymptotes are moved closer to $x^n$. Oscillation is detected if the sign of the quantities $s_i^n - s_i^{n-1}$ and $s_i^{n-1} - s_i^{n-2}$ have opposite signs. The asymptotes should take on values defined by

$$L_i^n = s_i^n - s(s_i^{n-1} - L_i^{n-1}) \tag{6.44}$$
$$U_i^n = s_i^n + s(U_i^{n-1} - s_i^{n-1})$$

(2) If the process is moving slowly, the problem needs to be relaxed by spreading the asymptotes. A slow convergence rate is identified if the signs of the difference in variables $s_i^n - s_i^{n-1}$ and $s_i^{n-1} - s_i^{n-2}$ is equal. If this is so, the asymptotes should take on values defined by

$$L_i^n = s_i^n - (s_i^{n-1} - L_i^{n-1})/s \tag{6.45}$$
$$U_i^n = s_i^n + (U_i^{n-1} - s_i^{n-1})/s$$

The value of $s$ is taken between zero and one, and typically ranges from 0.5 to 0.9 according to Svanberg [132].

The convex subproblem generated by the MMA algorithm has traditionally been solved by a dual method. For large nonlinear programming problems the dual problem is generally more difficult to solve, however, two characteristics of MMA subproblems make the dual approach more attractive:

(1) The convexity of the subproblem guarantees that the optimal Lagrange multipliers in the dual problem directly correspond to the optimal variables in the primal problem.

(2) The separability of the objective and constraints allows the $n$-dimensional minimization problem to be separated into $n$ one-dimensional minimizations.

The dual of the subproblem reduces to a maximization of the dual objective function over the dual variables (Lagrange multipliers of the primal problem) with the only constraint enforcing the dual variables to be non-negative. This maximization problem can typically be solved efficiently with a

gradient projection method. Most implementations of the MMA algorithm use the Fletcher-Reeves conjugate gradient method to solve the dual subproblem.

Another attractive method for solving the subproblem is the primal-dual interior point method. This method relaxes the KKT conditions via slack variables and solves the KKT system by Newton's method. This approach is best for problems involving a large number of constraints where solving a linear system proportional in size to the number of constraints becomes costly.

### 6.5.3 Sequential Convex Programming

One of the hindrances of the original MMA algorithm is that it will not converge for some problems with known solutions. Svanberg [133] and Zillober [145] refer to this characteristic of MMA as a lack of global convergence. Zillober remedied this situation [144] by introducing a merit function and line search for solving the MMA subproblem. The MMA algorithm with these additions subsequently received the names sequential convex programming (SCP) and globally convergent MMA (GCMMA) [133]. Zillober then extended the SCP method from structural optimization problems to general nonlinear programming problems [145]. This section presents the merit function and line search additions to MMA that make up the SCP algorithm.

Simply taking the solution of the MMA subproblem as the next iterate does not guarantee convergence of the algorithm. Similar to SQP, a merit function with the choice of a penalty parameter that ensures the search direction is a descent direction of the merit function is required for global convergence. The MMA subproblem of equation 6.40 can be rewritten with the box constraints included in the inequality constraints as

$$\begin{aligned} &\min_{s_i} \tilde{z}(s_i), \quad i = 1, ..., n_x \\ &\tilde{h}_j(s_i) = 0, \quad j = 1, ..., n_h \\ &\tilde{g}_k(s_i) \geq 0, \quad k = 1, ..., n_g + 2n_x. \end{aligned} \tag{6.46}$$

The augmented Lagrangian merit function is used for the SCP method to assure an objective

decrease

$$\Phi_r(x,u) = \tilde{z}(x) + \sum_{j=1}^{n_h} \left( \eta_j h_j(s) + \frac{1}{2} r_j h_j^2(x) \right) + \sum_{k=1}^{n_g+2n_x} \left( \gamma_k g_k(x) + \frac{1}{2} r_k g_k^2(x) \right) \tag{6.47}$$

where $r_i$ are penalty parameters. The gradient of the augmented Lagrangian is then written as:

$$\nabla \Phi_r(x,\gamma) = \left[ \nabla \tilde{z}(x) + A(x) \left( \bar{\gamma} + R\bar{h}(x) \right) \hat{h}(x) \right] \tag{6.48}$$

where $\bar{\gamma}$ is a vector containing the Lagrange multipliers for the active constraints and zeros for the inactive constraints, $\bar{h}$ is a vector containing the constraint function values for the active constraints and zeros the inactive constraints, $\hat{h}(x)$ is a vector containing the function values for active constraints or the quantity $-\gamma_j / r_j$ for inactive constraints, $A(x)$ is a vector containing the derivatives of the constraints, and $R$ is diagonal matrix of penalty parameters.

If the search direction causes an increase in the merit function, the penalty parameters $r_i$ are updated until a decrease in the merit function results. A line search is then performed with respect to the merit function given in equation 6.47 to determine the new iterate.

The inclusion of the merit function and linear search are the essential differences between the MMA and SCP algorithms. Zillober [145] proves the global convergence of SCP by showing the line search method satisfies the descent property for the merit function and that the penalty parameters $r_i$ are indeed bounded.

# Chapter 7

# Transient Aerothermoelastic Optimization

## 7.1 Introduction

This chapter develops the necessary techniques for optimizing transient aerothermoelastic problems. The ultimate goal is to be able to find an optimal solution to a problem that changes rapidly through time and for multi-disciplinary design criteria that may be competing with each other. Such situations are common for aerothermoelastic problems so a sound mathematical and numerical framework for navigating the design space is very valuable.

The adjoint sensitivity analysis procedure shown in chapter 6 is extended to transient problems. This formulation will be used to solve optimization problems relating to the internal design of a structure subject to transient pressure and heat loading. The basic concept pursued herein is to use topology optimization to determine the material layout that will mitigate heating while maintaining structural integrity. Transient topology optimization is not a new development with notable work by Li [99], however, transient topology optimization has not yet been explored as a design tool for aerothermoelastic problems. This work will hopefully open doors regarding the usefulness of such an approach for this class of problems.

## 7.2 Transient Adjoint Sensitivity Analysis

In this section we generalize the adjoint sensitivity analysis shown in section 6.4.2 to time-dependent problems. This procedure applies to all design criteria $q$ which are a function of the state variables $\boldsymbol{\mathcal{U}}$. The state vector $\boldsymbol{\mathcal{U}}$ may be comprised of state values from several different equation

types. For aerothermoelastic problems, $\boldsymbol{\mathcal{U}} = [\boldsymbol{U}, \boldsymbol{u}, \boldsymbol{T}]$, which is composed of the state variables from the Navier-Stokes equations, elastodynamics equation, and the transient heat transfer equation.

We begin by combining all criteria $q$, total (dynamic + static contribution) residuals $\boldsymbol{\mathfrak{R}}$ and state vectors $\boldsymbol{\mathcal{U}}$ through time as follows.

$$\begin{aligned} \tilde{\boldsymbol{q}} &= \left[q^0, \ldots, q^{n_t}\right]^T, \\ \tilde{\boldsymbol{\mathfrak{R}}} &= \left[\boldsymbol{\mathfrak{R}}^0, \ldots, \boldsymbol{\mathfrak{R}}^{n_t}\right]^T, \\ \tilde{\boldsymbol{\mathcal{U}}} &= \left[\boldsymbol{\mathcal{U}}^0, \ldots, \boldsymbol{\mathcal{U}}^{n_t}\right]^T. \end{aligned} \tag{7.1}$$

Using this compact form, the derivative of the transient design criteria $Q$ with respect to the design variables $s_k$ can be written as

$$\frac{dQ}{ds_k} = \frac{\partial \mathcal{Q}}{\partial \tilde{\boldsymbol{q}}}^T \frac{d\tilde{\boldsymbol{q}}}{ds_k}, \tag{7.2}$$

The derivative of the contributions to the objectives at all time steps, $\tilde{\boldsymbol{q}}$, with respect to the design variable $s_k$, is given by

$$\frac{d\tilde{\boldsymbol{q}}}{ds_k} = \frac{\partial \tilde{\boldsymbol{q}}}{\partial s_k} + \frac{\partial \tilde{\boldsymbol{q}}}{\partial \tilde{\boldsymbol{\mathcal{U}}}}^T \frac{d\tilde{\boldsymbol{\mathcal{U}}}}{ds_k}. \tag{7.3}$$

Then, the derivative of the state vector with respect of the design variables, $d\tilde{\boldsymbol{\mathcal{U}}}/ds_k$, is computed from the total derivative of the residual equations

$$\frac{d\tilde{\boldsymbol{\mathfrak{R}}}}{ds_k} = \frac{\partial \tilde{\boldsymbol{\mathfrak{R}}}}{\partial s_k} + \frac{\partial \tilde{\boldsymbol{\mathfrak{R}}}}{\partial \tilde{\boldsymbol{\mathcal{U}}}} \frac{d\tilde{\boldsymbol{\mathcal{U}}}}{ds_k} = \mathbf{0}. \tag{7.4}$$

Solving Eq. (7.4) for $d\tilde{\boldsymbol{\mathcal{U}}}/ds_k$ and substituting the result into Eq. (7.3) yields

$$\frac{d\tilde{\boldsymbol{q}}}{ds_k} = \frac{\partial \tilde{\boldsymbol{q}}}{\partial s_k} - \frac{\partial \tilde{\boldsymbol{q}}}{\partial \tilde{\boldsymbol{\mathcal{U}}}}^T \left[\frac{\partial \tilde{\boldsymbol{\mathfrak{R}}}}{\partial \tilde{\boldsymbol{\mathcal{U}}}}\right]^{-1} \frac{\partial \tilde{\boldsymbol{\mathfrak{R}}}}{\partial s_k}, \tag{7.5}$$

The adjoint subproblem is then defined as

$$\tilde{\boldsymbol{a}} = -\left[\frac{\partial \tilde{\boldsymbol{\mathfrak{R}}}}{\partial \tilde{\boldsymbol{\mathcal{U}}}}\right]^{-T} \frac{\partial \tilde{\boldsymbol{q}}}{\partial \tilde{\boldsymbol{\mathcal{U}}}}. \tag{7.6}$$

For the first step (when $n = 0$) the derivatives of the dynamic residual vector with respect to the state vectors $\partial \boldsymbol{\mathfrak{R}}^0 / \partial \boldsymbol{\mathcal{U}}^j$ are given by:

$$\frac{\partial \boldsymbol{\mathfrak{R}}^0}{\partial \boldsymbol{\mathcal{U}}^j} = \begin{cases} \mathbf{I} & \forall\ j = 0, \\ \mathbf{0} & \forall\ j = 1, \ldots, n_t. \end{cases} \tag{7.7}$$

For steps $n > 0$, the derivatives $\partial \mathfrak{R}^n / \partial \boldsymbol{\mathcal{U}}^j$ are given by

$$\frac{\partial \mathfrak{R}^n}{\partial \boldsymbol{\mathcal{U}}^j} = \begin{cases} -\frac{\mathbf{M}^n}{\Delta t} \equiv \boldsymbol{\mathcal{J}}^n & \forall\ j = n-1, \\ \frac{\mathbf{M}^n}{\Delta t} + \frac{\partial \mathbf{R}_s^n}{\partial \boldsymbol{\mathcal{U}}^n} \equiv \boldsymbol{\mathfrak{J}}^n & \forall\ j = n, \\ \mathbf{0} & \forall\ j \in \{1, \ldots, N_t\} \setminus \{n-1, n\}\,. \end{cases} \tag{7.8}$$

Rewriting Eqs. (7.6)-(7.8) leads to the following adjoint system of equations

$$\underbrace{\begin{bmatrix} \mathbf{I} & \left[\boldsymbol{\mathcal{J}}^0\right]^T & & & \\ & \left[\boldsymbol{\mathfrak{J}}^1\right]^T & \left[\boldsymbol{\mathcal{J}}^1\right]^T & & \\ & & \left[\boldsymbol{\mathfrak{J}}^2\right]^T & \ddots & \\ & & & \ddots & \left[\boldsymbol{\mathfrak{J}}^{n_t-1}\right]^T \\ & & & & \left[\boldsymbol{\mathfrak{J}}^{n_t}\right]^T \end{bmatrix}}_{\mathbf{A}^T} \underbrace{\begin{pmatrix} \boldsymbol{a}^0 \\ \boldsymbol{a}^1 \\ \boldsymbol{a}^2 \\ \vdots \\ \boldsymbol{a}^{N_t} \end{pmatrix}}_{\tilde{\boldsymbol{a}}} = - \begin{pmatrix} \left(\partial \mathcal{Q} / \partial z^0\right)\left(\partial z^0 / \partial \boldsymbol{\mathcal{U}}^0\right) \\ \left(\partial \mathcal{Q} / \partial z^1\right)\left(\partial z^1 / \partial \boldsymbol{\mathcal{U}}^1\right) \\ \left(\partial \mathcal{Q} / \partial z^2\right)\left(\partial z^2 / \partial \boldsymbol{\mathcal{U}}^2\right) \\ \vdots \\ \left(\partial \mathcal{Q} / \partial z^{N_t}\right)\left(\partial z^{N_t} / \partial \boldsymbol{\mathcal{U}}^{n_t}\right) \end{pmatrix}. \tag{7.9}$$

The structure of the matrix $\mathbf{A}^T$ is banded and contains a single block at the end of the diagonal. It is the structure of the adjoint problem that lends this system to an efficient backward time integration starting from time step $n_t$. Given the adjoint solution $\tilde{\boldsymbol{a}}$, the derivative of the objective with respect to the design variable $s_k$ can then be determined as

$$\frac{dZ}{ds_k} = \frac{\partial \mathcal{Q}^T}{\partial \tilde{\boldsymbol{q}}} \left( \frac{\partial \tilde{\boldsymbol{q}}}{\partial s_k} + \tilde{\boldsymbol{a}}^T \frac{\partial \tilde{\mathfrak{R}}}{\partial s_k} \right). \tag{7.10}$$

For solving the optimization problems presented in Section 6.2 we write the fluid states to disk at all time steps when solving the forward problem. The block matrices $\boldsymbol{\mathcal{J}}^n$ and $\boldsymbol{\mathfrak{J}}^n$ are recomputed for every time step in the adjoint sensitivity analysis. Strategies for reducing the storage requirements and for reducing the computational costs for the adjoint sensitivity analysis have been explored by Rumpfkeil and Zingg [120, 121] and Hinze [55].

Given the transient adjoint procedure for solving the sensitivity analysis, we are then able to feed a numerical optimization method (covered in chapter 6.5) with the design criteria values (computed during the forward analysis) and the design sensitivities (computed via the sensitivity analysis). The optimizer then computes a new set of design variables and we iterate the process until we have reached a satisfactory solution.

## 7.3 Numerical Example Problems

The numerical examples contained in the remainder of this chapter pull together the many of the developments in the previous chapters up to this point in order to perform coupled, transient design optimization. These problems highlight the multi-disciplinary analysis and optimization capabilities researched and developed for this dissertation.

### 7.3.1 2-Dimensional Thermoelastic Topology Optimization

One purpose of this problem is demonstrate the use of a phase changing material to absorb energy through the latent heat effect. In this manner, the interior of the structure is able to act as a thermal protection system similar to what is commonly done with surface ablator materials. A thermal protection system that doesn't alter the external shape of the body is desirable from a design standpoint, but a standing questions is how much of a design benefit does this offer in terms of heat mitigation? In this example we use transient topology optimization to answer that question.

#### 7.3.1.1 Problem Description

Figure 7.1 shows the geometry, material parameters, and boundary conditions for this problem. The block is heated at its left edge with a uniform heat load that ramps up from zero at $t = 0\ s$ to 800 $W$ at $t = 0.5\ s$ at which point the heat load is abruptly removed. The upper and lower corners on the left edge of the block are are pinned and a force of 1000 $N$ acts at the center of the block. Given these thermoelastic boundary conditions, the objective of the optimization problem is to simultaneously minimize the temperature at the center of the block and to minimize the structural compliance (maximize stiffness).

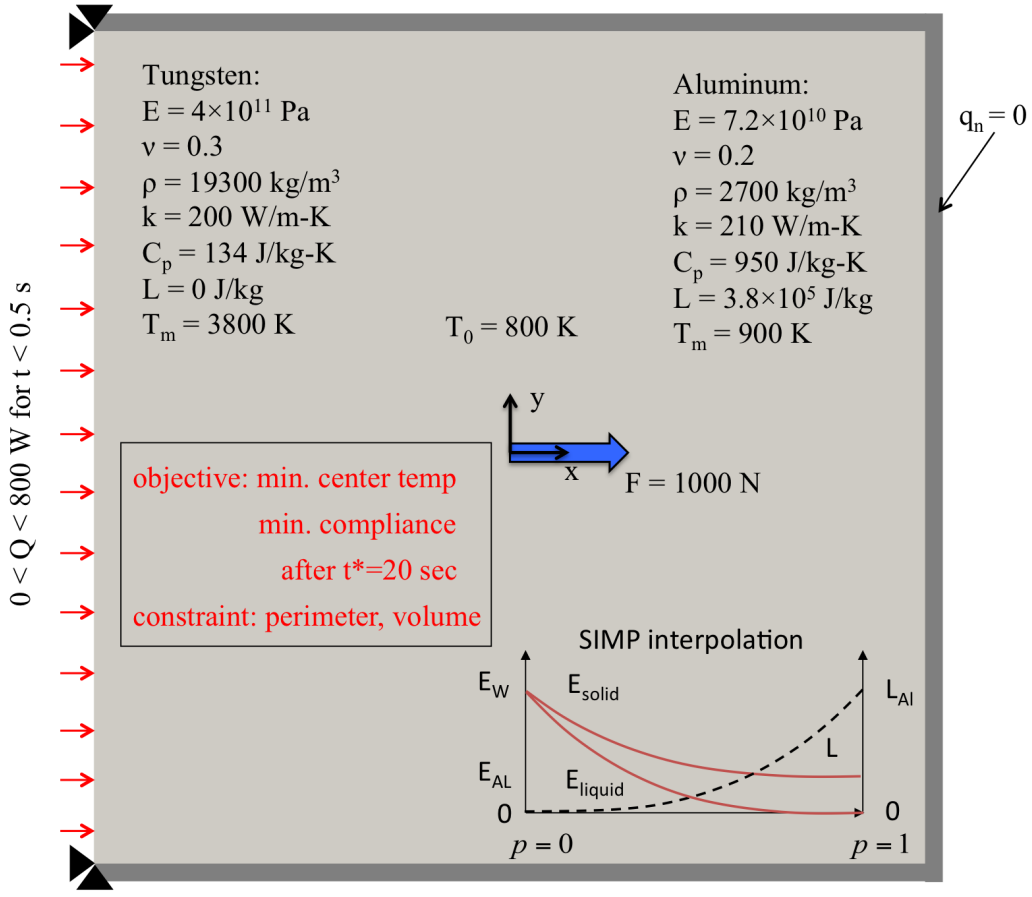



Figure 7.1: Two-dimensional thermoelastic topology optimization geometry and boundary conditions.

#### 7.3.1.2 Computational Mesh and Model

Figure 7.2 shows the 30x30 element mesh used for this problem. The thermoelastic forward analysis is solved using the BDF time integrator, a constant time step of $\Delta t = 0.5\ s$, and is run for 40 time iterations.

All problems were run with the SIMP material model acting on both the latent heat and elastic modulus material properties with a penalization factor of $\beta = 3$. A "perimeter" constraint is imposed on the volume of tungsten material such that the variation in the distribution of tungsten is held below a predefined value. This effectively groups the tungsten together.

#### 7.3.1.3 Results and Discussion

Figure 7.3 shows the Pareto front for this problem as the weighting of the objective shifts from heavily weighting minimum compliance towards heavily weighting minimum temperature. These figures show that the prevailing theme behind the optimized designs is to put phase changing aluminum between the applied heat flux and center of the structure in order to create a "shield" of material the mitigates the heat load. When the stiffness of the structure is weighted more heavily, the optimizer clearly shifts it emphasis towards placing tungsten between the pinned supports and the load point to create a solid load path.

### 7.3.2 Axisymmetric Aerothermoelastic Internal Material Design

This numerical example was constructed to show the use of transient optimization for an aerothermoelastic problem. The nose tip studied in sections 2.6.3 and 5.5.1 is used for this purpose.

#### 7.3.2.1 Problem Description

This problem is very similar in boundary conditions to that of the aeroheating nose tip example in section 5.5.1 however the fluid/structure interface includes aeroelastic and aerothermal boundary conditions. Figure 7.4 shows the problem specifications. The objective of the optimization

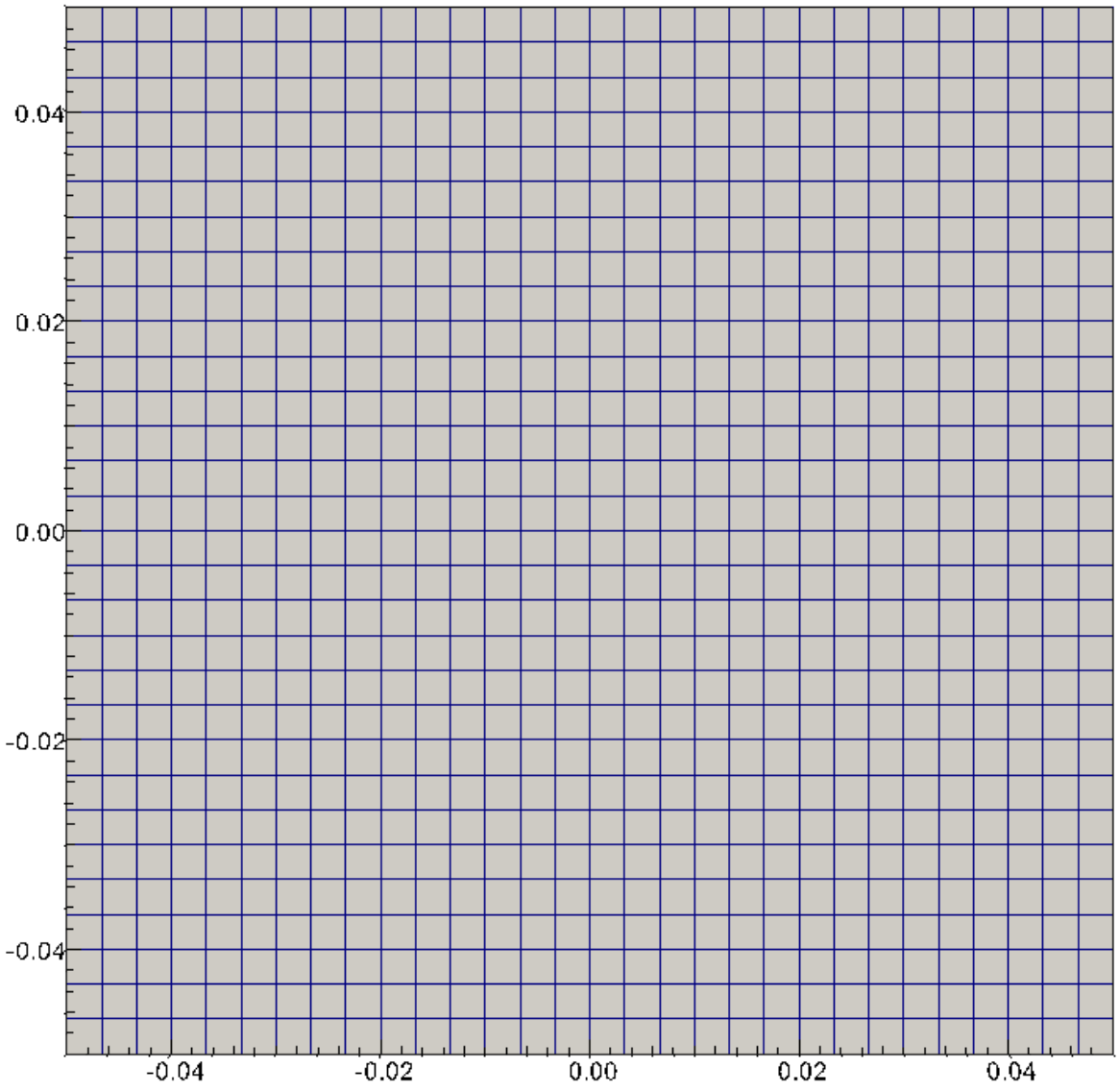



Figure 7.2: Two-dimensional thermoelastic topology optimization computational mesh.

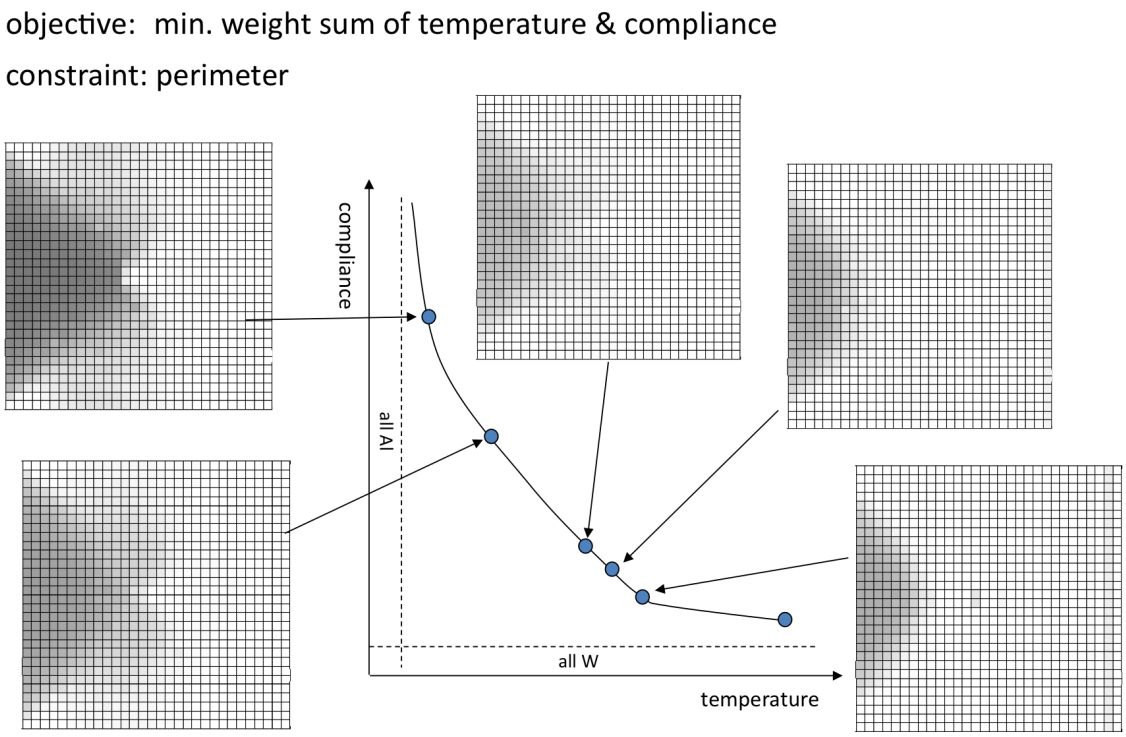



Figure 7.3: Pareto front showing the change in designs as the weighting of the objective shifts between minimum temperature and minimum compliance. The dark regions represent phase changing aluminum and the light regions represent tungsten.

problem is to design the optimal material layout within the design domain that minimizes the temperature at a specified region and at the end of a specified time window.

As noted in section 2.4.1.1, an approximate Jacobian have been implemented and is used for the forward solve. The "backward in time" adjoint problem defined by equation 7.9, however, necessitates the use of a consistent Jacobian in order to guarantee accurate sensitivities. Generation of a consistent, hand-coded analytic Jacobian for aerothermoelastic problems is undoubtedly the most efficient way to solve the optimization problem, however was outside the scope of this work and is subject of future endeavors in this area. Hence, a consistent, analytic Jacobian has not been implemented during the coarse of this work and a consistent Jacobian computed via finite differences is far too costly to be useful for this problem. In light of this, the decision was made to move forward with solving the adjoint problem with the available approximate Jacobian but with the understanding that the computed sensitivities may not be accurate enough to drive the optimizer towards the true optimum.

#### 7.3.2.2 Computational Mesh and Model

Figure 7.5 shows the mesh for this problem. The numerical model as fully aerothermoelastic one and is time integrated via a BDF scheme for the fluid and heat equations and via a Newmark method for the elasticity equations. Time is advanced adaptively to a maximum time of 100.0 $s$. The transient simulation is then run repeated within the framework of the transient adjoint sensitivity analysis (section 7.2) and optimization (chapter 6) processes.

#### 7.3.2.3 Results and Discussion

Figure 7.5 shows an intermediate design from a early design iteration during the optimization processes. Though not exciting from a visual perspective, what this figure shows is that the optimizer is beginning to place low conductivity material in the upper left corner of the design domain. This is precisely the location of the highest temperature for the block of element making up the design region. This means that the optimizer is shifting the low conductivity material to

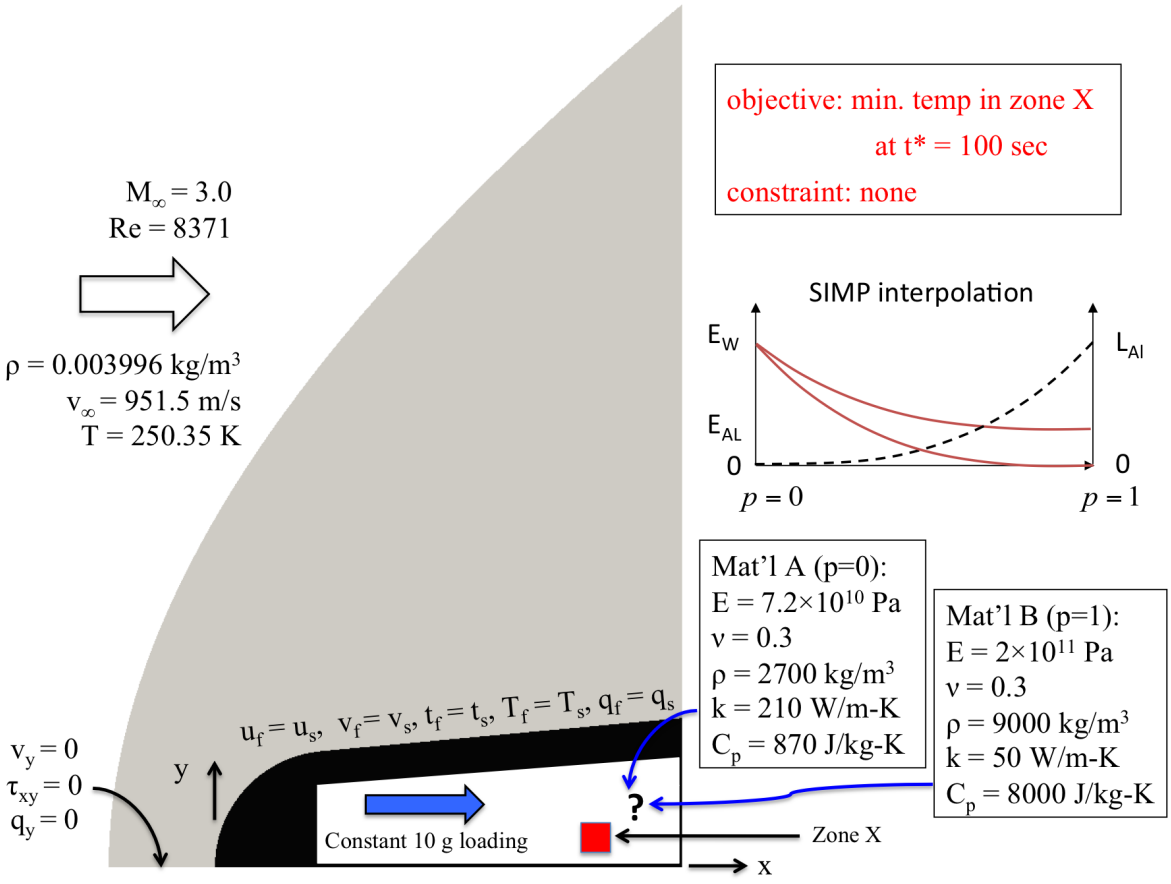



Figure 7.4: Axisymmetric nose tip topology optimization geometry and boundary conditions.

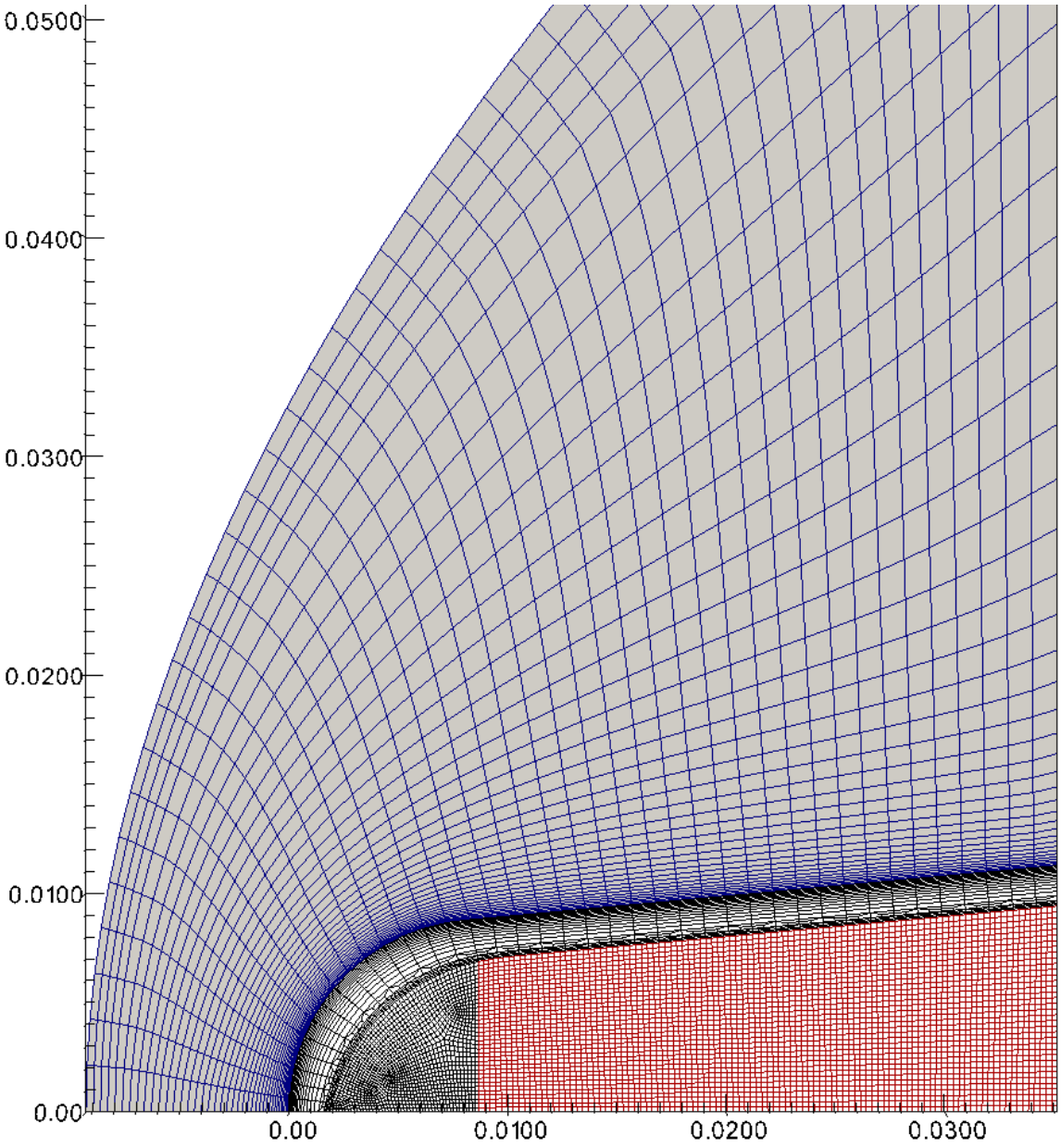



Figure 7.5: Axisymmetric nose tip computational mesh.

an area where it will have the most impact for minimizing the objective (temperature). After 10 design iterations the temperature at the objective zone is reduced by approximately 7 $K$.

This example confirms that the approximate Jacobians used for the transient adjoint sensitivity analysis are at least accurate enough to drive the optimization problem in a meaningful direction. Beyond this it is difficult to conclude if the end results generated with the approximate sensitivities are the true optimum.

This example and the previous one, however, confirm that one of the primary goals of this effort in transient optimization for aerothermoelastic problems was reached – the development and demonstration of a transient adjoint-based sensitivity analysis and optimization framework. From this perspective, the developments contained in this chapter are a success.

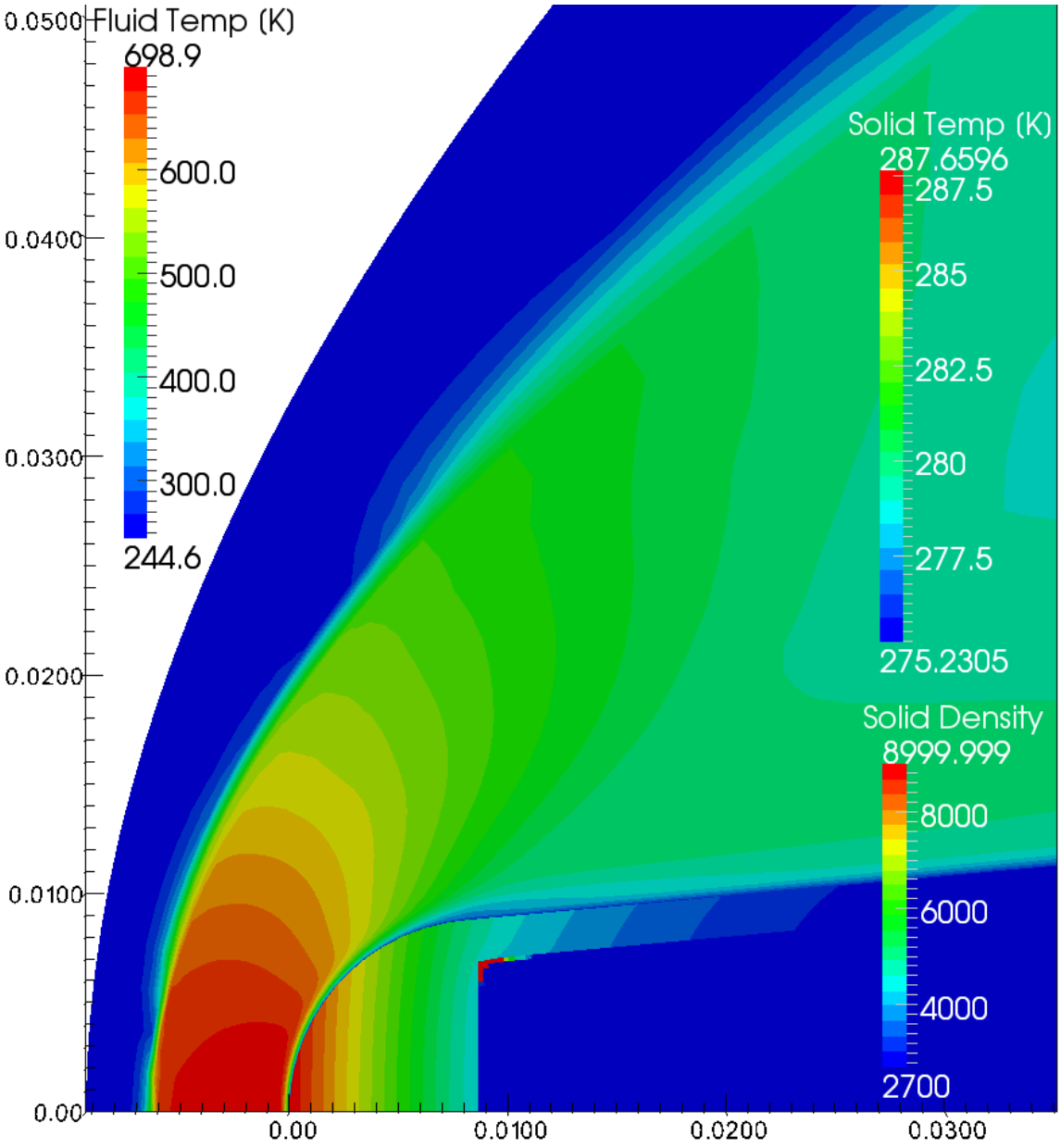



Figure 7.6: Axisymmetric nose tip design iteration from early in the optimization process.

# Chapter 8

# Conclusions

The objective of this dissertation was to develop, implement, and demonstrate a multi-disciplinary analysis and optimization methodology for high-speed aerothermoelastic problems. In the preceding chapters each of the these three disciplines were considered along with a means of interdisciplinary coupling and transient sensitivity analysis and optimization. In the remaining pages of this document the entire dissertation is summarized, the primary contributions of this effort are highlighted, and suggestions for future work are made.

## 8.1 Summary

As mentioned in the introduction of this thesis, aerothermoelastic problems are inherently coupled and often to a degree that warrants very careful consideration of the interactions which link them. Given the complexity of each individual field by itself (aerodynamics, heat transfer, and elasticity), the coupled responses due to the interplay of all three fields is often unwieldy. Designing with such an intricate system is difficult at best, even with considerable knowledge and experience. It is this fact that directed this dissertation effort down the path of numerical design optimization for aerothermoelastic systems. Here, the goal at hand was to develop and test a numerical sensitivity analysis approach that allows a design engineer to much more easily navigate the vast design space spanned by aerothermoelastic problems. The remainder of this section briefly summarizes the purpose and outcome of each chapter in this thesis.

Chapter 1 served as the introduction to this thesis as well as an overview of the computational

software platform used for completing this work. The code developed for this dissertation was written from the ground-up to perform tightly coupled multi-disciplinary analysis and optimization. It draws heavily on well established external libraries such as Exodus, Trilinos, and BLAS for mesh representation, parallel linear equation solving, and low-level matrix-vector operations, respectively.

Chapter 2 presented the stabilized finite element formulation used for solving compressible fluid dynamics problems. Several enhancements to the basic scheme where discussed in this chapter such as nodally averaged stabilization and discontinuity capturing fields [23] and an arbitrary Lagrangian-Eulerian frame of reference for moving mesh problems. Multiple code verification problems were shown for both subsonic and supersonic test cases.

Chapter 3 introduced the equation for transient heat transfer and its solution via the finite element method. Multiple extensions pertaining to the thermophysical aspects of aerothermoelastic problems were introduced in this section. An arbitrary Lagrangian-Eulerian frame of reference was implemented and tested to accommodate moving mesh problems. A phase change formulation was presented as it allowed us to model material behavior when the melting point of the material is exceeded. A stabilized finite element method was discussed and implemented that is well suited for handling problems involving high heat flux and low thermal conductivity – which often leads to non-physical oscillations in the solution with a standard Galerkin approach. Finally, a "heat of ablation" or Q* ablation model was developed and tested for predicting ablator response and heat flow mitigation in the structure, which is of great significance to atmospheric re-entry problems. Several numerical examples were presented and solved which highlighted each of these unique modeling capabilities.

Chapter 4 showed the governing equations for elastodynamics and presented the scheme used to solve thermoelasticly coupled structural dynamics problems. Both an elastodynamic finite element method and a transient heat transfer finite element method are assembled into a coupled formulation and solved via a Newmark time integrator for second-order-in-time structural dynamics problems. A numerical example is presented which solves the dynamic response of cantilevered beam that is excited by a time dependent convective heat flux.

Chapter 5 developed a coupling formulation for linking the Navier-Stokes equations of compressible fluid dynamics, the transient heat equation, and the elastodynamic equations into a tightly coupled, monolithic set of nonlinear equations which are solved simultaneously. The approach is based on the balance of solution variables and fluxes/traction at the fluid/solid interface and is achieved by manipulating the residual equations. This method has the advantage of not introducing additional variables such as Lagrange multipliers as is commonly done with other coupling schemes such as the mortar method based approaches. In addition, all equations are solved simultaneously as opposed to a staggered coupling scheme which solves each governing equation set separately. Numerical examples were shown for aeroheating, aerothermoelastic response, and ablation response problems.

Chapter 6 was largely an introduction to the theory of mathematical and numerical optimization that is intended to familiarize the reader with basic concepts and algorithms in use for computational design optimization.

Chapter 7 presented the theory and development of a transient, adjoint-based sensitivity analysis procedure for performing design optimization of aerothermoelastic systems. At present, this optimization capability has been used to solve thermoelasticly coupled material layout problems. Given the power of the transient adjoint sensitivity analysis, the full capabilities of this method have not been fully realized and its continued extension is a significant area for future work.

## 8.2 Primary Contributions

This section describes, in the author's view, the contributions of this dissertation towards advancing the state-of-the-art in aerothermoelastic modeling, simulation, and optimization.

The first contribution is the development and verification of the finite element based code written by the author for solving compressible gas dynamics problems, transient heat transfer problems, and structural dynamics problems. Given that many codes used for solving computational mechanics problems are written for a single discipline, one that is written with the intent for solving multi-disciplinary problems all within the same code base is certainly a unique and powerful re-

search tool. Several non-standard enhancements to the basic finite element formulations have either been developed or implemented for each of the disciplines. An ALE approach for moving meshes has been written for use in fluid and heat transfer problems. A stabilized formulation (GGLS) for heat transfer has been implemented and its usefulness has been demonstrated for aeroheating problems which involve high heating and low thermal conductivity materials. A phase change formulation was implemented that later facilitated the application of phase changing materials for topology optimization. Finally, a Q* ablation model was implemented that allowed ablation problems to be solved in a monolithically coupled fashion. A contribution of this effort that is not to be overlooked is the extensive verification that has been performed for this code. This helps to establish credibility of the code as an analysis tool and a worthy platform that can be trusted to deliver accurate results.

Another contribution of this work is the development of a flexible, extensible software platform for doing research on multi-disciplinary computational mechanics and design optimization. The code developed for this dissertation leverages many external software projects that greatly enhance its capabilities, especially in terms of linear equation solving. One of the primary objectives of the code itself was for it to be written as a scalable platform for high-performance parallel computing. Several of the numerical examples presented in the previous chapters confirm this objective has been met. In terms of design optimization, this software platform represents a unique capability for solving both forward analyses *and* sensitivity analyses in parallel.

The aerothermoelastic coupling technique presented in chapter 5 represents a major achievement of this dissertation. The basic idea behind the residual based coupling method was originally used in the GOMA computational fluid dynamics code at Sandia National Labs, and later extended to aeroheating problems. It was extended to fully coupled aerothermoelastic problems in this thesis and has shown great promise as an alternative to weak formulation coupling approaches such as discontinuous Galerkin methods or mortar methods.

The development and implementation of the transient adjoint sensitivity analysis solver also comprises a major achievement of this dissertation. Such a capability is quite uncommon, with a

notable exception being the adjoint solver contained in NASA's FUN3D computational fluid dynamics code. It is this adjoint solver that makes the optimization of transient problems, especially ones with a large number of design variables, feasible. Much effort has been spent to make the adjoint solver efficient and robust from a computational perspective since the backward solve procedure requires the re-initialization of the state vector at each time step and the re-computation of the Jacobian's with respect to multiple time states.

The use of design optimization techniques such as topology optimization for solving problems in aerothermoelasticity is the final contribution of this dissertation. At least to the author's knowledge, the combination of these two fields is unexplored territory. The work contained herein has only begun to explore this area and many challenges remain to effectively apply computational design optimization to aerothermoelastic problems.

## 8.3 Future Work

Aerothermoelastic problems are most relevant in the context of high-speed gas dynamics, especially hypersonic high-temperature gas dynamics, where the coupling is strong. Given the complex physical environment of hypersonic aerothermoelastic problems it is difficult to include all necessary physics in the computational model. This is compounded when the development of the computational framework begins from the ground-up and within the time-frame and scope of a Ph.D. thesis. The coupling method developed herein has proven to be quite efficient and robust from a numerical standpoint, yet much can be done to fully generalize it. The optimization capability developed for this thesis is also in a fledgling state, with much work to be done in order to fully realize the promise of design optimization for aerothermoelastic problems. This section briefly lists recommendations for future work in light of these three observations.

**Physics Modeling** Several additions can be made to improve the physics modeling capabilities of this work to make it a viable platform hypersonic flow simulation.

- *Turbulence Modeling* The inclusion of a turbulence model is needed for accurately account-

ing for high Reynolds number turbulent flows.

- *Thermochemical Real Gas Models* Hypersonic flows often necessitate that real gas effects are appropriately accounted for, as mentioned in section 2.2. The implementation of a reacting gas capability is needed to extend this code to applications in hypersonic gas dynamics.

**Physics Coupling** The coupling scheme presented in Chapter 5 can benefit from the additions listed below.

- *Extension to Non-Matching Meshes* The coupling approach used in this dissertation require the meshes be contiguous (connected) at the fluid/structure interface. Clearly, this is a limitation in terms of mesh generation (where, due to complexity, it may be difficult to generate the fluid and solid mesh as one) and efficiency of the meshes as the amount of mesh refinement needed for the solving the fluid and solid problems may vary greatly. This coupling method may be readily applied to non-matching meshes and would greatly strengthen it applicability to general problems.

- *Weak Formulation Coupling* A coupling method based on a weak variational formulation such as a mortar method would be useful in assessing the performance of the residual based coupling approach. Mortar methods are widely used coupling schemes and being able to compare the strengths of weaknesses of both approaches is needed to make a definitive statement about the favorable performance of one over the other.

- *Staggered Coupling* The monolithic coupling scheme used in this thesis has proven to work quite well. Being able to speak objectively about the superiority of a monolithic coupling scheme over a staggered coupling scheme requires that one be able to meaningfully compare the performance of each. Additionally, the ability to test a staggered coupling method is needed to fully understand the performance and accuracy gains one hopefully achieves with a fully coupled monolithic method.

**Design Optimization** Much has been done in the transient sensitivity analysis area for aerothermoelastic problems, yet much work is still required to make this a truly mature research tool.

- *Transient Direct Sensitivity Analysis* A transient direct sensitivity analysis solver is needed to efficiently handle the cases where there are few optimization variables but many design criteria. In such cases, the direct approach is preferred over the adjoint approach developed and used in this document.

- *Shape Optimization Mesh Manipulation Module* A module for controlling shape changes is need to realize the benefits of this work for transient aerothermoelastic shape optimization problems.

## Nomenclature

**Italic Letters**

| | |
|---|---|
| $a$ | acceleration or spectral radius |
| $B'$ | nondimensional ablation mass transfer rate |
| $C$ | specific heat or coefficient |
| $c$ | speed of sound or specific heat |
| $D$ | drag force, diffusion coefficient, or constitutive tensor |
| $E$ | total energy per unit mass or elastic modulus |
| $e$ | internal energy per unit mass |
| $G$ | shear modulus |
| $g$ | inequality constraint set or metric tensor |
| $H$ | total enthalpy per unit mass |
| $h$ | equality constraint set or enthalpy per unit of mass |
| $K$ | stiffness |
| $k$ | thermal conductivity |
| $L$ | Lagrange function or lower bound |
| $N$ | finite element shape function |
| $n$ | normal component |
| $p$ | fluid static pressure |
| $Q$ | volumetric heating term |
| $q$ | heat flux |
| $R$ | gas constant |
| $r$ | "r-switch" exponent |
| $S$ | source term |
| $s$ | design variable or recession distance |
| $T$ | temperature |
| $t$ | time variable |
| $u$ | displacement or upper bound |
| $v$ | velocity or slack variable |
| $w$ | weighting function |

$x$ spatial coordinate

$z$ objective function

**Bold Italic Letters**

$\boldsymbol{a}$ adjoint vector

$\boldsymbol{b}$ boundary condition function or right-hand side vector

$\boldsymbol{c}$ vector of "slack variables

$\boldsymbol{F}$ vector of inviscid fluxes

$\boldsymbol{f}$ force or flux vector

$\boldsymbol{G}$ vector of viscous fluxes

$\boldsymbol{N}$ vector of shape function

$\boldsymbol{n}$ normal vector

$\boldsymbol{p}$ Krylov vector or search direction

$\boldsymbol{q}$ design criterion or heat flux vector

$\boldsymbol{R}$ static or steady residual vector

$\boldsymbol{r}$ unit vector

$\boldsymbol{S}$ fluid source vector

$\boldsymbol{s}$ vector of design variables

$\boldsymbol{U}$ vector of fluid conervative state variables

$\boldsymbol{u}$ vector of structural displacements

$\boldsymbol{v}$ fluid velocity vector

$\boldsymbol{W}$ vector of test functions

$\boldsymbol{x}$ vector of mesh coordinates

**Bold Roman Letters**

$\mathbf{A}$ inviscid flux Jacobian matrix

$\mathbf{B}$ differential operator matrix or strain-displacement matrix

$\mathbf{C}$ structural damping matrix

$\mathbf{D}$ viscous flux Jacobian matrix or material constitutive matrix

$\mathbf{H}$ Hessian matrix

$\mathbf{I}$ identity matrix

$\mathbf{J}$ static or steady Jacobian matrix

$\mathbf{K}$ diffusivity matrix or structural stiffness matrix

$\mathbf{M}$ structural mass matrix

**Caligraphy Letters**

$\boldsymbol{\mathcal{L}}$ Navier-Stokes or transient heat transfer operator

$\boldsymbol{\mathcal{R}}$ dynamic or unsteady residual vector

$\boldsymbol{\mathcal{S}}$ trial solution space

$\boldsymbol{\mathcal{U}}$ combined multi-disciplinary state vector

$\boldsymbol{\mathcal{V}}$ test function space

$\mathcal{L}_h$ latent heat

**Bold "Mathfrak" Letters**

$\mathfrak{R}$ space of real numbers

**Lowercase Greek Letters**

$\alpha$ generalized-$\alpha$ parameter, ALE parameter, or coefficient of thermal exp.

$\alpha_d$ Rayleigh damping coefficient

$\beta$ Nemark algorithmic parameter or ALE parameter

$\beta_d$ Rayleigh damping coefficient

$\chi$ species mass fracion

$\delta$ shock capturing parameter

$\epsilon$ finite difference perturbation, convergence tolerance or strain

$\eta$ second natural coordinate direction or Lagrange multiplier

$\gamma$ ratio of specific heats, Newmark parameter or Lagrange multiplier

$\kappa$ fluid thermal conductivity

$\lambda$ bulk viscosity or Lagrange multiplier

$\mu$ dynamic viscosity

$\nu$ shock capturing parameter or Poisson's ratio

$\omega$ reaction rate

$\phi$ damping parameter

$\rho$ density

$\sigma$ stress

$\tau$ stabilization parameter or viscous shear stress

$\theta$ relaxation factor or ALE parameter

$\xi$ first natural coordinate direction

$\zeta$ third natural coordinate direction

**Uppercase Greek Letters**

$\Delta$ incremental value

$\Gamma$ boundary of the domain

$\Omega$ volume of the domain

$\Phi$ shape function set

$\Psi$ generic aeroelastic equations

$\Sigma$ summation symbol

$\Theta$ $\Theta$-scheme time integration parameter

**Bold Lowercase Greek Letters**

$\boldsymbol{\epsilon}$ strain tensor

$\boldsymbol{\lambda}$ vector of Lagrange equality multipliers

$\boldsymbol{\mu}$ vector of Lagrange inequality multipliers

$\boldsymbol{\sigma}$ stress tensor

$\boldsymbol{\tau}$ viscous stress tensor

**Superscripts**

$( )^*$ optimum solution or heat of ablation
$( )^c$ continuity equation
$( )^e$ element or energy equation
$( )^h$ discrete approximation
$( )^k$ optimization iteration
$( )^L$ lower bound
$( )^m$ momentum equation or nonlinear iterate
$( )^n$ time iterate or node
$( )^T$ transpose
$( )^U$ upper bound
$( )^{tc}$ thermochemical

**Subscripts**

$( )_\Gamma$ surface or mesh boundary
$( )_\Omega$ volume or mesh interior
$( )_a$ ambient
$( )_c$ continuity equation or convection
$( )_d$ drag
$( )_e$ energy equation
$( )_f$ fluid, force, or skin friction
$( )_g$ inequality constraint
$( )_h$ equality constraint or heat flux
$( )_i$ vector component index
$( )_j$ vector component index
$( )_k$ vector component index
$( )_l$ vector component index or liquid
$( )_m$ momentum equation
$( )_p$ under constant pressure
$( )_s$ structural, solid, specie, or design variable
$( )_T$ temperature
$( )_t$ traction, thermal, or temporal
$( )_u$ displacement
$( )_v$ under constant volume
$( )_w$ wall
$( )_0$ initial guess or total/stagnation value
$( )_\infty$ far field
$( )_{abl}$ ablation
$( )_{adv}$ advection
$( )_{dco}$ discontinuity capturing operator
$( )_{diff}$ diffusion

$(\,)_{dof}$ degree of freedom
$(\,)_{fsi}$ fluid-structure interface
$(\,)_{galerkin}$ Galerkin
$(\,)_{ggls}$ Galerkin gradient least squares
$(\,)_{ref}$ reference
$(\,)_{supg}$ streamline-upwind Petrov-Galerkin

**Symbols Over The Letter**

$\bar{(\,)}$ time-average average or prescribed quantity
$\ddot{(\,)}$ second time derivative
$\dot{(\,)}$ first time derivative
$\hat{(\,)}$ unit vector representation
$\tilde{(\,)}$ approximate quantity

**Other Symbols**

$\nabla(\,)$ gradient
$d(\,)$ differential

**Dimensionless Numbers**

$M$ Mach number
$Pe$ Peclet number
$Pr$ Prandtl number
$Re$ Reynolds number

**Acronyms**

ALE Arbitrary Lagrangian-Eulerian
BFGS Broydon-Fletcher-Goldfarb-Shanno
CFD Computational Fluid Dynamic
CFL Courant-Friedrichs-Lewy
CPU Central Processing Unit
dcp discontinuity capturing parameter
FE Finite Element
FSI Fluid-Structure Interaction
FV Finite Volume
GMRES Generalized Minimal Residual
I**LU** Incomplete **LU**
KKT Karush-Kuhn-Tucker
MMA Method of Moving Asymptotes
NAND Nested Analysis and Design
ODE Ordinary Differential Equation
PDE Partial Differential Equation
SA Sensitivity Analysis
SAND Simultaneous Analysis and Design

| | |
|---|---|
| SCP | Sequential Convex Programming |
| SIMP | Simple Isotropic Material with Penalization |
| SQP | Sequential Quadratic Programming |
| SUPG | Streamline-Upwind Petrov-Galerkin |

## Bibliography


[1] National Aeronautics and Space Administration. Pieter g. buning's home page, http://aaac.larc.nasa.gov/ buning/.

[2] AIAA. 4th aiaa cfd drag prediction workshop, http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/.

[3] S.K. Aliabadi, S.E. Ray, and T.E. Tezduyar. Supg finite element computation of viscous compressible flows based on the conservation and entropy variables formulations. Computational Mechanics, 11:300–312, 1993.

[4] S.K. Aliabadi and T.E. Tezduyar. Space-time finite element computation of compressible flows involving moving boundaries and interfaces. Computer Methods in Applied Mechanics and Engineering, 107:209–223, 1993.

[5] S.K. Aliabadi and T.E. Tezduyar. Parallel fluid dynamics computations in aerospace applications. International Journal for Numerical Methods in Fluids, 21:783–805, 1995.

[6] S. Allmaras. Lagrange multiplier implementation of dirichlet boundary conditions in compressible navier-stokes finite element methods. In Proceedings of the 17th AIAA Computational Fluid Dynamics Conference, Toronto, Ontario Canada, June 6–, 2005.

[7] A.J. Amar, B.F. Blackwell, and J.R. Edwards. One-dimensional ablation using a full newton's method and finite control volume procedure. Journal of Thermophysics and Heat Transfer, 22:71–82, 2008.

[8] A.J. Amar, B.F. Blackwell, and J.R. Edwards. Development and verification of a one-dimensional ablation code including pyrolysis gas flow. Journal of Thermophysics and Heat Transfer, 23:59–71, 2009.

[9] J.D. Anderson. Fundamentals of Aerodynamics. McGraw Hill Higher Education, 3rd edition edition, 2001.

[10] A.K. Aziz and P. Monk. Continuous finite elements in space and time for the heat equation. AMS Mathematics of Computation, 52(186):255–274, 1989.

[11] G.E. Barter and D.L. Darmofal. Shock capturing with higher-order, pde-based artificial viscosity. In Proceedings of the 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, June 25–28, 2007.

[12] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier-stokes equations. Journal of Computational Physics, 131:267–279, 1997.

[13] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2d euler equations. Journal of Computational Physics, 138:251–285, 1997.

[14] C.E. Baumann and J.T. Oden. A discontinuous hp finite element method for the euler and navier-stokes equations. International Journal for Numerical Methods in Fluids, 31:79–95, 1999.

[15] G.J. Le Beau, S.E. Ray, S.K. Aliabadi, and T.E. Tezduyar. Supg finite element computation of compressible flows with the entropy and conservation variables formulations. Computer Methods in Applied Mechanics and Engineering, 104:397–422, 1993.

[16] G.J. Le Beau and T.E. Tezduyar. Finite element computation of compressible flows with the supg formulation. In Advances in Finite Element Analysis in Fluid Dynamics, pages 21–27, 1991.

[17] T. Belytschko, R. Gracie, and G. Ventura. A review of extended/generalized finite element methods for material modeling. Modelling and Simulation in Materials Science and Engineering, 17:043001, 2009.

[18] M.P. Bendsoe. Optimal shape design as a material distribution system. Structural Optimization, 1:193–202, 1989.

[19] M.P. Bendsoe and O. Sigmund. Material interpolation schemes in topology optimization. Archive of Applied Mechanics, 69:635–654, 1999.

[20] M.P. Bendsoe and O. Sigmund. Topology Optimization: Theory, Methods and Applications. Springer, 2003.

[21] C. Bischof, A. Carle, G. Corliss, A. Griewank, and P. Hovland. Adifor: Generating derivative codes from fortran programs. Scientific Computing, 1(1):11–29, 1992.

[22] C. Bischof, L. Roh, and A. Mauer-Oats. Adic: An extensible automatic differentiation tool for ansi-c. Scientific – Practice and Experience, 27(12):1427–1456, 1997.

[23] S.W. Bova and R.B. Bond. Stabilized finite element scheme for high speed flows with chemical non-equilibrium. In preparation for the Proceedings of the 48th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, FL, January 4–7, 2010.

[24] S.W. Bova, D.R. Noble, P.K. Notz, and M.A. Howard. How to perform tightly coupled aerothermal analysis in aria. Technical report, Sandia National Labs Internal Memo, 2009.

[25] A.N. Brooks and T.J.R. Hughes. Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. Computer Methods in Applied Mechanics and Engineering, 32:199–259, 1982.

[26] G.V. Candler, M.D. Barnhardt, T.W. Drayna, I. Nompelis, D.M. Peterson, and P. Subbareddy. Unstructured grid approaches for accurate aeroheating simulations. In Proceedings of the 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, June 25–28, 2007.

[27] J.E. Carter. Numerical solutions of the navier-stokes equations for the supersonic laminar flow over a two-dimensional compression corner. Technical report, NASA Technical Report NASA-TR-R-385, 1972.

[28] F. Chalot and T.J.R. Hughes. A consistent equilibrium chemistry algorithm for hypersonic flows. Computer Methods in Applied Mechanics and Engineering, 112:25–40, 1994.

[29] Y.-K. Chen, F.S. Milos, and T. Gokcen. Validation of a three-dimensional ablation and thermal response simulation code. In Proceedings of the 10th AIAA/ASME Joint Thermophysics and Heat Transfer Conference, Chicago, IL, June 23–July 1, 2010.

[30] J. Chung and G.M. Hulbert. A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-alpha method. Journal of Applied Mechanics, 60:371–376, 1993.

[31] MacNeal-Schwendler Corporation. Nastran Theoretical Manual, 1972.

[32] J. Dolbow, N. Moes, and T. Belytschko. Discontinuous enrichment in finite elements with a partition of unity method. Finite Elements in Analysis and Design, 36:235–260, 2000.

[33] V. Dolejsi. On the discontinuous galerkin method for the numerical solution of the navier-stokes equations. International Journal for Numerical Methods in Fluids, 45:1083–1106, 2004.

[34] J. Donea. A taylor-galerkin method for convective transport problems. International Journal for Numerical Methods in Engineering, 20:101–119, 1984.

[35] J. Donea and A. Huerta. Finite Element Methods for Flow Problems. John Wiley & Sons, 2003.

[36] H.A. Eschenauer and N. Olhoff. Topology optimization of continuum structures: a review. Applied Mechanics Review, 54:331–390, 2001.

[37] V.D. Fachinotti and M. Bellet. Linear tetrahedra finite elements for thermal shock problems. International Journal for Numerical Methods for Heat and Fluid Flow, 16:590–601, 2006.

[38] C. Farhat and P. Geuzaine. The discrete geometric conservation law and its effects on nonlinear stability and accuracy. In Proceedings of the 15th AIAA Computational Fluid Dynamics Conference, Anaheim, CA, June 11–14, 2001.

[39] C. Farhat and P. Geuzaine. Design and analysis of robust ale time-integrators for the solution of unsteady flow problems on moving grids. Computer Methods in Applied Mechanics and Engineering, 193:4073–4095, 2004.

[40] C. Farhat, P. Geuzaine, and C. Grandmont. Mixed explicit/implicit time integration of coupled aeroelastic problems: Three-field formulation, geometric conservation and distributed solution. International Journal for Numerical Methods in Fluids, 21:807–835, 1995.

[41] C. Farhat, P. Geuzaine, and C. Grandmont. The discrete geometric conservation law and the nonlinear stability of ale schemes for the solution of flow problems on moving grids. Journal of Computational Physics, 174:669–694, 2001.

[42] C. Farhat, P. Geuzaine, T. Lieu, and A. Rajasekharan. AERO-F Manual, 2006.

[43] C. Farhat and M. Lesoinne. Enhanced partitioned procedures for solving nonlinear transient aeroelastic problems. In Proceedings of the 39th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Long Beach, CA, April 20–23, 1998.

[44] C. Farhat and M. Lesoinne. Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems. Computer Methods in Applied Mechanics and Engineering, 182:499–515, 2000.

[45] C.A. Felippa, K.C. Park, and C. Farhat. Partitioned analysis of coupled mechanical systems. Computer Methods in Applied Mechanics and Engineering, 190:3247–3270, 2001.

[46] L.P. Franca and E.G. Dutra Do Carmo. The galerkin gradient least-squares method. Computer Methods in Applied Mechanics and Engineering, 74:41–54, 1989.

[47] P. Geuzaine, C. Grandmont, and C. Farhat. Design and analysis of ale schemes with provable second-order time-accuracy for inviscid and viscous flow simulations. Journal of Computational Physics, 191:206–227, 2003.

[48] P.E. Gill, M.A. Saunders, and W. Murray. SNOPT: An SQP algorithm for large scale constrained optimization, 1997.

[49] P. Gresho, R. Lee, R. Sani, and T. Stullich. Recent Advances in Numerical Methods in Fluids, chapter On the time-dependent FEM solution of the incompressible Navier-Stokes equations in two and three dimensions. Pineridge Press, Swansea, UK, 1980.

[50] A. Griewank, D. Juedes, and J. Utke. Algorithm 755: Adol-c: A package for the automatic differentiation of algorithms written in c/c++. ACM Transactions on Mathematical Software, 22(2):131–167, 1996.

[51] B. Hassan, D.W. Kuntz, and D.L. Potter. Coupled fluid/thermal prediction of ablating hypersonic vehicles. In Proceedings of the 36th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, June 12–15, 1998.

[52] B. Hassan, D.W. Kuntz, D.E. Salguero, and D.L. Potter. A coupled fluid/thermal/flight dynamics approach for predicting hypersonic vehicle performance. In Proceedings of the 35th AIAA Thermophysics Conference, Anaheim, CA, June 11–14, 2001.

[53] G. Hauke and T.J.R. Hughes. A comparative study of different sets of variables for solving compressible and incompressible flows. Computer Methods in Applied Mechanics and Engineering, 153:1–44, 1998.

[54] H.M. Hilber, T.J.R. Hughes, and R.L. Taylor. Improved numerical dissipation for time integration algorithms in structural dynamics. Earthquake Engineering and Structural Dynamics, 5:283–292, 1977.

[55] M. Hinze, A. Walther, and J. Sternberg. An optimal memory-reduced procedure for calculating adjoints of the instationary Navier-Stokes equations. Optimal Control Applications & Methods, 27(1):19–40, 2006.

[56] C.W. Hirt, A.A. Amsden, and J.L. Cook. An arbitrary lagrangian-eulerian computing method for all flow speeds. Journal of Computational Physics, 14:227–253, 1974.

[57] M. Hogge and P. Gerrekens. Two-dimensional deforming finite element methods for surface ablation. AIAA Journal, 23(3):465–472, 1985.

[58] M.S. Holden. A study of flow separation in regions of shock wave-boundary later interaction in hypersonic flow. In Proceedings of the 11th AIAA Fluid and Plasma Dynamics Conference, Seattle, WA, 1978.

[59] M.A. Howard and S.W. Bova. Coupling methods for aerodynamic heating problems. In Proceedings of the 49th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 2010.

[60] A.N. Hrymak, G.J. McRae, and A.W. Westerberg. Combined analysis and optimization of heat transfer surfaces. Journal of Heat Transfer, 107:527–532, 1985.

[61] T.J.R. Hughes and L.P. Franca. A new finite element formulation for computational fluid dynamics: Vii. the stokes problem with various well-posed boundary conditions: Symmetric formulations that converge for all velocity/pressure spaces. Computer Methods in Applied Mechanics and Engineering, 65:85–96, 1987.

[62] T.J.R. Hughes, L.P. Franca, and M. Balestra. A new finite element formulation for computational fluid dynamics: V. circumventing the babuka-brezzi condition: a stable petrov-galerkin formulation of the stokes problem accommodating equal-order interpolations. Computer Methods in Applied Mechanics and Engineering, 59:85–99, 1986.

[63] T.J.R. Hughes, L.P. Franca, and G.M. Hulbert. A new finite element formulation for computational fluid dynamics: Viii. the galerkin/least-squares method for advective-diffusive equations. Computer Methods in Applied Mechanics and Engineering, 73:173–189, 1989.

[64] T.J.R. Hughes, L.P. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: I. symmetric forms of the compressible euler and navier-stokes equations and the second law of thermodynamics. Computer Methods in Applied Mechanics and Engineering, 54:223–234, 1986.

[65] T.J.R. Hughes, L.P. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: Vi. convergence analysis of the generalized supg formulation for linear time-dependent multidimensional advective-diffusive systems. Computer Methods in Applied Mechanics and Engineering, 59:85–99, 1987.

[66] T.J.R. Hughes, W.K. Liu, and T.K. Zimmerman. Lagrangian-eulerian finite element formulation for incompressible viscous flows. Computer Methods in Applied Mechanics and Engineering, 29:329–349, 1981.

[67] T.J.R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: Iii. the generalized streamline operator for multidimensional advective-diffusive systems. Computer Methods in Applied Mechanics and Engineering, 58:305–328, 1986.

[68] T.J.R. Hughes and M. Mallet. A new finite element formulation for computational fluid dynamics: Iv. a discontinuity-capturing operator for multidimensional advective-diffusive systems. Computer Methods in Applied Mechanics and Engineering, 58:329–336, 1986.

[69] T.J.R. Hughes, M. Mallet, and A. Mizukami. A new finite element formulation for computational fluid dynamics: Ii. beyond supg. Computer Methods in Applied Mechanics and Engineering, 54:341–355, 1986.

[70] T.J.R. Hughes, G. Scovazzi, and T.E. Tezduyar. Stabilized methods for compressible flows. Journal of Scientific Computing, 2008. Published online 20 August 2008.

[71] T.J.R. Hughes and T.E. Tezduyar. Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible euler equations. Computer Methods in Applied Mechanics and Engineering, 45:217–284, 1984.

[72] F. Ilinca and J.-F. Hetu. Galerkin gradient least-squares formulations for transient conduction heat transfer. Computer Methods in Applied Mechanics and Engineering, 191:3073–3097, 2002.

[73] ABAQUS Inc. ABAQUS 6.4 Theory Manual, 2003.

[74] ANSYS Inc. Ansys icem cfd, http://www.ansys.com/products/icemcfd.asp.

[75] ANSYS Inc. ANSYS Verification Manual, Release 11.0, 2007.

[76] ANSYS Inc. Theory Reference for ANSYS and ANSYS Workbench, Release 11.0, 2007.

[77] Fluent Inc. Fluent 6.3 User's Guide, 2006.

[78] Kitware Inc. Paraview: Open source scientific visualization, http://www.paraview.org.

[79] K. James and J.R.R.A. Martins. Three-dimensional structural topology optimization of an aircraft wing using level set methods. In Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, BC Canada, June 10–12, 2008.

[80] A. Jameson. Cfd for aerodynamic design and optimization: Its evolution over the last three decades. In Proceedings of the 16th AIAA Computational Fluid Dynamics Conference, Orlando, FL, June 23–26, 2003.

[81] K.E. Jansen and T.J.R. Hughes. A better consistency for stabilized finite element methods. Computer Methods in Applied Mechanics and Engineering, 174:153–170, 1999.

[82] B.S. Kirk. Adaptive finite element simulation of flow and transport applications on parallel computers. PhD thesis, University of Texas at Austin, 2007.

[83] B.S. Kirk. Adiabatic shock capturing in perfect gas hypersonic flows. International Journal for Numerical Methods in Fluids, 2009.

[84] B.S. Kirk and A.J. Amar. Orion re-entry: modeling the aerothermodynamic environment and thermal protection system response. Seminar Presentation, May 2008.

[85] B.S. Kirk and G.F. Carey. Development and validation of a supg finite element scheme for the compressible navier-stokes equations using a modified inviscid flux discretization. International Journal for Numerical Methods in Fluids, 57:265–293, 2008.

[86] D.W. Kuntz, B. Hassan, and D.L. Potter. Predictions of ablating hypersonic vehicles using an iterative coupled fluid/thermal approach. AIAA Journal of Thermophysics and Heat Transfer, 15(2):129–139, 2001.

[87] Sandia National Laboratories. Aztec iterative solver package, http://trilinos.sandia.gov/packages/aztecoo/.

[88] Sandia National Laboratories. The cubit geometry and mesh generation toolkit, http://cubit.sandia.gov/cubitprogram.html.

[89] Sandia National Laboratories. Direct sparse solver package, http://trilinos.sandia.gov/packages/amesos/.

[90] Sandia National Laboratories. Epetra linear algebra services package, http://trilinos.sandia.gov/packages/epetra/.

[91] Sandia National Laboratories. Exodus ii project, http://sourceforge.net/projects/exodusii/.

[92] Sandia National Laboratories. Nox nonlinear solver package, http://trilinos.sandia.gov/packages/nox/.

[93] Sandia National Laboratories. Rythmos transient integration for differential equations package, http://trilinos.sandia.gov/packages/rythmos/.

[94] Sandia National Laboratories. Seacas project, http://sourceforge.net/projects/seacas/.

[95] Sandia National Laboratories. The trilinos project, http://trilinos.sandia.gov.

[96] Argonne National Laboratory. The portable, extensible toolkit for scientific computation, http://www.mcs.anl.gov/petsc/petsc-as/.

[97] T. Larsson and M. Ronnqvist. Simultaneous structural analysis and design based on augmented lagrangian duality. Structural Optimization, 9:1–9, 1995.

[98] P.D. Lax and B. Wendroff. Systems of conservation laws. Communications in Pure and Applied Mathematics, 1960.

[99] Y. Li, K. Saitou, and N. Kikuchi. Topology optimization of thermally actuated compliant mechanisms considering time-transient effect. Finite Elements in Analysis and Design, 40:1317–1331, 2004.

[100] R. Lohner, K. Morgan, and O.C. Zienkiewicz. The solution of non-linear hyperbolic equation systems by the finite element method. International Journal for Numerical Methods in Fluids, 4:1043–1063, 1984.

[101] R.W. MacCormack. The effect of viscosity in hypervelocity impact cratering (reprint of aiaa paper 69-354, 1969). AIAA Journal of Spacecraft and Rockets, 40(5), 2003.

[102] R.W. MacCormack and A.J. Paullay. Computational efficiency achieved by time splitting of finite difference operators. In 10th AIAA Aerospace Sciences Meeting, 1972.

[103] A. Martin and I.D. Boyd. Strongly coupled computation of material response and nonequilibrium ow for hypersonic ablation. In Proceedings of the 41st AIAA Thermophysics Conference, San Antonio, TX, June 22–June 25, 2009.

[104] J.R.R.A. Martins, J. Reuther, and J. Alonso. High-fidelity aerostructural design optimization of supersonic business jet. AIAA Journal of Aircraft, 41(3):523–530, 2004.

[105] K. Maute, M. Nikbay, and C. Farhat. Coupled analytical sensitivity analysis and optimization of three-dimensional nonlinear aeroelastic systems. AIAA Journal, 39(11):2051–2061, 2001.

[106] K. Maute, M. Nikbay, and C. Farhat. Conceptual layout of aeroelastic wing structures by topology optimization. In Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Denver, CO, April 22–25, 2002.

[107] F.S. Milos and Y.-K. Chen. Ablation, thermal response, and chemistry program for analysis of thermal protection systems. In Proceedings of the 10th AIAA/ASME Joint Thermophysics and Heat Transfer Conference, Chicago, IL, June 23–July 1, 2010.

[108] R. Mittal and G. Iaccarino. Immersed boundary methods. Annual Review of Fluid Mechanics, 37:239–261, 2005.

[109] N.M. Newmark. A method of computation for structural dynamics. Journal of Engineering Mechanics Division ASCE, 85:67–94, 1959.

[110] E.J. Nielsen, B. Diskin, and N.K. Yamaleev. Discrete adjoint-based design optimization of unsteady turbulent flows on dynamic unstructured grids. In Proceedings of the 19th AIAA Computational Fluid Dynamics Conference, San Antonio, TX, June 22–25, 2009.

[111] J. Nocedal and S.J. Wright. Numerical Optimization. Springer, 1999.

[112] C.E. Orozco and O.N. Ghattas. Massively parallel aerodynamic shape optimization. Computing Systems in Engineering, 3(1-4):311–320, 1992.

[113] C.E. Orozco and O.N. Ghattas. A reduced sand method for nonlinear design problems in mechanics. In 37th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference and Exhibit, Salt Lake City, UT, April 15–17 1996.

[114] C. Park. Review of chemical-kinetic problems for future nasa missions, i: Earth entries. Journal of Thermophysics and Heat Transfer, 7(3):385–398, 1993.

[115] C. Park, J.T. Howe, R.L. Jaffe, and G.V. Candler. Review of chemical-kinetic problems for future nasa missions, ii: Mars entries. Journal of Thermophysics and Heat Transfer, 8(1):9–23, 1994.

[116] C.S Peskin. Numerical analysis of blood flow in the heart. Journal of Computational Physics, 25:220–252, 1977.

[117] C.S. Peskin. The immersed boundary method. Acta Numerica, 11:479–517, 2002.

[118] C.A. Powars and R.M. Kendall. User's Manual for Aerotherm Chemical Equilibrium (ACE) Computer Program. Aerotherm Corp., Mountain View, CA, May 1969.

[119] J.N. Reddy and D.K. Gartling. The Finite Element Method in Heat Transfer and Fluid Dynamics. CRC Press, 2nd edition edition, 2001.

[120] M. P. Rumpfkeil and D. W. Zingg. Unsteady optimization using a discrete adjoint approach applied to aeroacoustic shape design. In 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008.

[121] M. P. Rumpfkeil and D. W. Zingg. The optimal control of unsteady flows with a discrete adjoint method. Optimization and Engineering, 11(1):5–22, 2010.

[122] K. Schittkowski. Nlpql: A fortran subroutine for solving constrained nonlinear programming problems. Annals of Operations Research, 5:485–500, 1997.

[123] H. Schlichting and K. Gersten. Boundary-Layer Theory, volume 8th Edition. Spring-Verlag, 2003.

[124] F. Shakib and T.J.R. Hughes. A new finite element formulation for computational fluid dynamics: Ix. fourier analysis of space-time galerkin/least-squares algorithms. Computer Methods in Applied Mechanics and Engineering, 87:35–58, 1991.

[125] F. Shakib, T.J.R. Hughes, and Z. Johan. A new finite element formulation for computational fluid dynamics: X. the compressible euler and navier-stokes equations. Computer Methods in Applied Mechanics and Engineering, 89:141–219, 1991.

[126] Lawrence Berkeley National Laboratory Sherry Li. Superlu sparse nonsymmetric solver package, http://crd.lbl.gov/ xiaoye/superlu/.

[127] O. Sigmund. Design of material structures using topology optimization. PhD thesis, Department of Solid Mechanics, Technical University of Denmark, 1994.

[128] O. Sigmund and J. Petersson. Numerical instabilities in topology optimization: A survey on procedures dealing with checkerboards, mesh-dependencies and local minima. Structural Optimization, 16:68–75, 1998.

[129] S.I. Silton. Navier-stokes computations for a spinning projectile from subsonic to supersonic speeds. Technical report, U.S. Army Research Laboratory, ARL-TR-2850, Sept 2000.

[130] T.M. Smith, C.C. Ober, and A.A. Lorber. Sierra/premo-a new general purpose compressible flow simulation code. In 32nd AIAA Fluid Dynamics Conference, St. Louis, MO, June 24–27, 2002 2002.

[131] NASA Ames Research Staff. DPLR Package Users Guide, 2006.

[132] K. Svanberg. The method of moving asymptotes - a new method for structural optimization. International Journal of Numerica Methods in Engineering, 24:359–373, 1987.

[133] K. Svanberg. The mma for modeling and solving structural optimization problems. Technical report, Department of Mathematics, KTH, Stockholm, 2000.

[134] M. Tabata. A finite element approximation corresponding to the upwind finite differencing. Memoirs of Numerical Mathematics, 4:47–63, 1977.

[135] J.C. Tannehill and D.A. Pletcher. Computational Fluid Mechanics and Heat Transfer. Taylor and Francis, 2nd edition edition, 1997.

[136] T.E. Tezduyar and T.J.R. Hughes. Development of time-accurate finite element techniques for first-order hyperbolic systems with particular emphasis on the compressible euler equations. Technical report, NASA Technical Report NASA-CR-204772, 1982.

[137] T.E. Tezduyar and T.J.R. Hughes. Finite element formulations for convection dominated flows with particular emphasis on the compressible euler equationsfinite element formulations for convection dominated flows with particular emphasis on the compressible euler equations. In Proceedings of the 21st AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 1983.

[138] T.E. Tezduyar and M. Senga. Supg finite element computation of inviscid supersonic flows with yz beta shock-capturing. Computers and Fluids, 36:147–159, 2007.

[139] University of Florida Tim Davis. Umfpack: Unsymmetric multifrontal sparse lu factorization package, http://www.cise.ufl.edu/research/sparse/umfpack.

[140] H. Tran and C. Farhat. An integrated platform for the simulation of fluid-structure-thermal interaction problems. In Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Denver, CO, April 22–25, 2002.

[141] M.J. Turner, R.W. Clough, H.C. Martin, and L.J. Topp. Stiffness and deflection analysis of complex structures. Journal of Aeronautical Sciences, 23:805–824, 1956.

[142] C.C. Wong. Pinca: A scalable parallel program for compressible gas dynamics with non-equilibrium chemistry. Technical report, Sandia National Laboratories, Technical Report, SAND 94-2436, 1995.

[143] W.L. Wood, M. Bossak, and O.C. Zienkiewicz. An alpha modification of newmark's method. International Journal for Numerical Methods in Engineering, pages 1562–1566, 1981.

[144] C. Zillober. A globally convergent version of the method of moving asymptotes. Structural Optimization, 6:166–174, 1993.

[145] C. Zillober. Global convergence of a nonlinear programming method using convex approximations. Numerical Algorithms, 27:256–259, 2001.

# Appendix A

## Two-Dimensional Euler Flux Jacobians and Viscous Tensor

The two-dimensional Euler fluxes $\boldsymbol{F}_i$ are defined as:

$$\boldsymbol{F}_1 = \begin{pmatrix} \rho u_1 \\ \rho u_1^2 + p \\ \rho u_1 u_2 \\ u_1(\rho E + p) \end{pmatrix}, \quad \boldsymbol{F}_2 = \begin{pmatrix} \rho u_2 \\ \rho u_1 u_2 \\ \rho u_2^2 + p \\ u_2(\rho E + p) \end{pmatrix} \tag{A.1}$$

The components of the Euler flux Jacobian matrices, $\boldsymbol{A}_i$, can be derived by the following definition:

$$\frac{\partial \boldsymbol{F}_1}{\partial x_1} = \boldsymbol{A}_1 \frac{\partial \boldsymbol{U}}{\partial x_1} \Rightarrow \begin{pmatrix} \frac{\partial(\rho u_1)}{\partial x_1} \\ \frac{\partial(\rho u_1^2 + p)}{\partial x_1} \\ \frac{\partial(\rho u_1 u2)}{\partial x_1} \\ \frac{\partial(\rho u_1 E + u_1 p)}{\partial x_1} \end{pmatrix} = \boldsymbol{A}_1 \begin{pmatrix} \frac{\partial \rho}{\partial x_1} \\ \frac{\partial \rho u_1}{\partial x_1} \\ \frac{\partial \rho u_2}{\partial x_1} \\ \frac{\partial \rho E}{\partial x_1} \end{pmatrix} \tag{A.2}$$

$$\frac{\partial \boldsymbol{F}_2}{\partial x_2} = \boldsymbol{A}_2 \frac{\partial \boldsymbol{U}}{\partial x_2} \Rightarrow \begin{pmatrix} \frac{\partial(\rho u_2)}{\partial x_1} \\ \frac{\partial(\rho u_1 u2)}{\partial x_1} \\ \frac{\partial(\rho u_2^2 + p)}{\partial x_1} \\ \frac{\partial(\rho u_2 E + u_2 p)}{\partial x_1} \end{pmatrix} = \boldsymbol{A}_2 \begin{pmatrix} \frac{\partial \rho}{\partial x_2} \\ \frac{\partial \rho u_1}{\partial x_2} \\ \frac{\partial \rho u_2}{\partial x_2} \\ \frac{\partial \rho E}{\partial x_2} \end{pmatrix} \tag{A.3}$$

The rows of the $\boldsymbol{A}_1$ Jacobian are derived as follows:

$$\frac{\partial F_1(1)}{\partial U} = \left[\frac{\partial(\rho u_1)}{\partial \rho}, \frac{\partial(\rho u_1)}{\partial \rho u_1}, \frac{\partial(\rho u_1)}{\partial \rho u_2}, \frac{\partial(\rho u_1)}{\partial \rho E}\right] \tag{A.4}$$

$$= [0, 1, 0, 0] \tag{A.5}$$

$$\frac{\partial F_1(2)}{\partial U} = \frac{\partial(\rho u_1^2 + p)}{\partial U} \tag{A.6}$$

$$= \frac{\partial\left(\rho u_1^2 + [(\gamma-1)\rho e]\right)}{\partial U} \tag{A.7}$$

$$= \frac{\partial\left(\rho u_1^2 + \left[(\gamma-1)(\rho E - \frac{1}{2}\rho(u_1^2+u_2^2))\right]\right)}{\partial U} \tag{A.8}$$

$$= \frac{\partial\left(\rho u_1^2 + (\gamma-1)\rho E - \frac{1}{2}(\gamma-1)\rho(u_1^2+u_2^2)\right)}{\partial U} \tag{A.9}$$

$$= \frac{\partial(\rho u_1^2)}{\partial \rho}\frac{\partial \rho}{\partial x_1} + \frac{\partial \rho u_1^2}{\partial u_1}\frac{\partial u_1}{\partial x_1} + (\gamma-1)\frac{\partial \rho E}{\partial x_1} \tag{A.10}$$

$$-\frac{1}{2}(\gamma-1)\left[\frac{\partial \rho(u_1^2+u_2^2)}{\partial \rho}\frac{\partial \rho}{\partial x_1} + \frac{\partial \rho(u_1^2+u_2^2)}{\partial u_1}\frac{\partial u_1}{\partial x_1} + \frac{\partial \rho(u_1^2+u_2^2)}{\partial u_2}\frac{\partial u_2}{\partial x_1}\right] \tag{A.11}$$

$$= u_1^2\frac{\partial \rho}{\partial x_1} + 2\rho u_1\frac{\partial u_1}{\partial x_1} + (\gamma-1)\frac{\partial \rho E}{\partial x_1} \tag{A.12}$$

$$-\frac{1}{2}(\gamma-1)\left[(u_1^2+u_2^2)\frac{\partial \rho}{\partial x_1} + 2\rho u_1\frac{\partial u_1}{\partial x_1} + 2\rho u_2\frac{\partial u_2}{\partial x_1}\right] \tag{A.13}$$

$$= u_1^2\frac{\partial \rho}{\partial x_1} + 2\rho u_1\left[\frac{1}{\rho}\left(\frac{\partial \rho u_1}{\partial x_1} - u_1\frac{\partial \rho}{\partial x_1}\right)\right] + (\gamma-1)\frac{\partial \rho E}{\partial x_1} \tag{A.14}$$

$$-\frac{1}{2}(\gamma-1)\left(u_1^2+u_2^2\right)\frac{\partial \rho}{\partial x_1} - \frac{1}{2}(\gamma-1)2\rho u_1\left[\frac{1}{\rho}\left(\frac{\partial \rho u_1}{\partial x_1} - u_1\frac{\partial \rho}{\partial x_1}\right)\right] \tag{A.15}$$

$$-\frac{1}{2}(\gamma-1)2\rho u_2\left[\frac{1}{\rho}\left(\frac{\partial \rho u_2}{\partial x_1} - u_2\frac{\partial \rho}{\partial x_1}\right)\right] \tag{A.16}$$

$$= u_1^2\frac{\partial \rho}{\partial x_1} + 2u_1\frac{\partial \rho u_1}{\partial x_1} - 2u_1^2\frac{\partial \rho}{\partial x_1} + (\gamma-1)\frac{\partial \rho E}{\partial x_1} \tag{A.17}$$

$$-\frac{1}{2}(\gamma-1)\left(u_1^2+u_2^2\right)\frac{\partial \rho}{\partial x_1} - (\gamma-1)u_1\frac{\partial \rho u_1}{\partial x_1} + (\gamma-1)u_1^2\frac{\partial \rho}{\partial x_1} \tag{A.18}$$

$$-(\gamma-1)u_2\frac{\partial \rho u_2}{\partial x_1} + (\gamma-1)u_2^2\frac{\partial \rho}{\partial x_1} \tag{A.19}$$

$$= \left[\frac{1}{2}(\gamma-1)\left(u_1^2+u_2^2\right) - u_1^2\right]\frac{\partial \rho}{\partial x_1} + [3u_1 - \gamma u_1]\frac{\partial \rho u_1}{\partial x_1} \tag{A.20}$$

$$+[u_2 - \gamma u_2]\frac{\partial \rho u_2}{\partial x_1} + [\gamma - 1]\frac{\partial \rho E}{\partial x_1} \tag{A.21}$$

$$\tag{A.22}$$

$$
\begin{aligned}
\frac{\partial F_1}{\partial x_1}(3) &= \frac{\partial(\rho u_1 u_2)}{\partial x_1} \\
&= \frac{\partial \rho u_1 u_2}{\partial \rho}\frac{\partial \rho}{\partial x_1} + \frac{\partial \rho u_1 u_2}{\partial u_1}\frac{\partial u_1}{\partial x_1} + \frac{\partial \rho u_1 u_2}{\partial u_2}\frac{\partial u_2}{\partial x_1} \\
&= u_1 u_2 \frac{\partial \rho}{\partial x_1} + \rho u_2 \left[\frac{1}{\rho}\left(\frac{\partial \rho u_1}{\partial x_1} - u_1 \frac{\partial \rho}{\partial x_1}\right)\right] + \rho u_1 \left[\frac{1}{\rho}\left(\frac{\partial \rho u_2}{\partial x_1} - u_2 \frac{\partial \rho}{\partial x_1}\right)\right] \qquad \text{(A.23)} \\
&= u_1 u_2 \frac{\partial \rho}{\partial x_1} + u_2 \frac{\partial \rho u_1}{\partial x_1} - u_1 u_2 \frac{\partial \rho}{\partial x_1} + u_1 \frac{\partial \rho u_2}{\partial x_1} - u_1 u_2 \frac{\partial \rho}{\partial x_1} \\
&= -u_1 u_2 \frac{\partial \rho}{\partial x_1} + u_2 \frac{\partial \rho u_1}{\partial x_1} + u_1 \frac{\partial \rho u_2}{\partial x_1}
\end{aligned}
$$

$$
\begin{aligned}
\tfrac{\partial F_1}{\partial x_1}(4) &= \tfrac{\partial(\rho u_1 E + u_1 p)}{\partial x_1} \\
&= \tfrac{\partial \rho u_1 E}{\partial x_1} + \tfrac{\partial u_1 p}{\partial x_1} \\
&= \tfrac{\partial \rho u_1 E}{\partial x_1} + \tfrac{\partial u_1 \left[(\gamma-1)\left(\rho E - \frac{1}{2}\rho(u_1^2+u_2^2)\right)\right]}{\partial x_1} \\
&= \tfrac{\partial \rho u_1 E}{\partial x_1} + (\gamma-1)\tfrac{\partial u_1 \rho E}{\partial x_1} - \tfrac{1}{2}(\gamma-1)\tfrac{\partial \rho(u_1^3+u_1u_2^2)}{\partial x_1} \\
&= \gamma\left(\tfrac{\partial \rho u_1 E}{\partial x_1}\right) - \tfrac{1}{2}(\gamma-1)\tfrac{\partial \rho(u_1^3+u_1u_2^2)}{\partial x_1} \\
&= \gamma\left[\tfrac{\partial \rho u_1 E}{\partial \rho}\tfrac{\partial \rho}{\partial x_1} + \tfrac{\partial \rho u_1 E}{\partial u_1}\tfrac{\partial u_1}{\partial x_1} + \tfrac{\partial \rho u_1 E}{\partial E}\tfrac{\partial E}{\partial x_1}\right] \\
&\quad -\tfrac{1}{2}(\gamma-1)\left[\tfrac{\partial \rho(u_1^3+u_1u_2^2)}{\partial \rho}\tfrac{\partial \rho}{\partial x_1} + \tfrac{\partial \rho(u_1^3+u_1u_2^2)}{\partial u_1}\tfrac{\partial u_1}{\partial x_1} + \tfrac{\partial \rho(u_1^3+u_1u_2^2)}{\partial u_2}\tfrac{\partial u_2}{\partial x_1}\right] \\
&= \gamma\left[u_1 E\tfrac{\partial \rho}{\partial x_1} + \rho E\tfrac{\partial u_1}{\partial x_1} + \rho u_1\tfrac{\partial E}{\partial x_1}\right] \\
&\quad -\tfrac{1}{2}(\gamma-1)\left[(u_1^3+u_1u_2^2)\tfrac{\partial \rho}{\partial x_1} + (3\rho u_1^2+\rho u_2^2)\tfrac{\partial u_1}{\partial x_1} + (2\rho u_1 u_2)\tfrac{\partial u_2}{\partial x_1}\right] \\
&\quad \text{now substitute in } \tfrac{\partial u_1}{\partial x_1}\text{ , }\tfrac{\partial u_2}{\partial x_1}\text{ , and }\tfrac{\partial E}{\partial x_1} \\
&= \gamma u_1 E\tfrac{\partial \rho}{\partial x_1} + \gamma E\left(\tfrac{\partial \rho u_1}{\partial x_1} - u_1\tfrac{\partial \rho}{\partial x_1}\right) + \gamma u_1\left(\tfrac{\partial \rho E}{\partial x_1} - E\tfrac{\partial \rho}{\partial x_1}\right) \\
&\quad -\tfrac{1}{2}(\gamma-1)\left(u_1^3+u_1u_2^2\right)\tfrac{\partial \rho}{\partial x_1} - \tfrac{1}{2}(\gamma-1)\left(3u_1^2+u_2^2\right)\left(\tfrac{\partial \rho u_1}{\partial x_1} - u_1\tfrac{\partial \rho}{\partial x_1}\right) \\
&\quad -(\gamma-1)u_1u_2\left(\tfrac{\partial \rho u_2}{\partial x_1} - u_2\tfrac{\partial \rho}{\partial x_1}\right) \\
&= \gamma u_1 E\tfrac{\partial \rho}{\partial x_1} + \gamma E\tfrac{\partial \rho u_1}{\partial x_1} - \gamma u_1 E\tfrac{\partial \rho}{\partial x_1} + \gamma u_1\tfrac{\partial \rho E}{\partial x_1} - \gamma u_1 E\tfrac{\partial \rho}{\partial x_1} \\
&\quad -\left[\tfrac{1}{2}(\gamma-1)\left(u_1^3+u_1u_2^2\right)\right]\tfrac{\partial \rho}{\partial x_1} - \left[\tfrac{1}{2}(\gamma-1)3u_1^2\right]\tfrac{\partial \rho u_1}{\partial x_1} - \left[\tfrac{1}{2}(\gamma-1)u_2^2\right]\tfrac{\partial \rho u_1}{\partial x_1} \\
&\quad +\left[\tfrac{1}{2}(\gamma-1)3u_1^2u_2\right]\tfrac{\partial \rho}{\partial x_1} + \left[\tfrac{1}{2}(\gamma-1)u_1u_2^2\right]\tfrac{\partial \rho}{\partial x_1} \\
&\quad -\left[(\gamma-1)u_1u_2\right]\tfrac{\partial \rho u_2}{\partial x_1} + \left[(\gamma-1)u_1u_2^2\right]\tfrac{\partial \rho}{\partial x_1} \\
&= \left[-\gamma u_1 E - \tfrac{1}{2}(\gamma-1)\left(u_1^3+u_1u_2^2\right) + \tfrac{3}{2}(\gamma-1)\left(u_1^3+u_1u_2^2\right)\right]\tfrac{\partial \rho}{\partial x_1} \\
&\quad +\left[\gamma E - \tfrac{3}{2}(\gamma-1)u_1^2 - \tfrac{1}{2}(\gamma-1)u_2^2\right]\tfrac{\partial \rho u_1}{\partial x_1} \\
&\quad +\left[-(\gamma-1)u_1u_2\right]\tfrac{\partial \rho u_2}{\partial x_1} + \gamma u_1\tfrac{\partial \rho E}{\partial x_1} \\
&= \left[(\gamma-1)\left(u_1^3+u_1u_2^2\right) - \gamma u_1 E\right]\tfrac{\partial \rho}{\partial x_1} \\
&\quad +\left[\gamma E - (\gamma-1)u_1^2 - \tfrac{1}{2}(\gamma-1)\left(u_1^2+u_2^2\right)\right]\tfrac{\partial \rho u_1}{\partial x_1} \\
&\quad +\left[-(\gamma-1)u_1u_2\right]\tfrac{\partial \rho u_2}{\partial x_1} + \gamma u_1\tfrac{\partial \rho E}{\partial x_1}
\end{aligned}
\tag{A.24}
$$

Thus, the two-dimensional form of the Euler flux Jacobian matrix $\boldsymbol{A}_1$ is defined as:

$$\boldsymbol{A}_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ (\gamma-1)\frac{u^2}{2} - u_1^2 & 3u_1 - \gamma u_1 & u_2 - \gamma u_2 & \gamma - 1 \\ -u_1 u_2 & u_2 & u_1 & 0 \\ u_1\left[(\gamma-1)u^2 - \gamma E\right] & \gamma E - (\gamma-1)\left(u_1^2 + \frac{u^2}{2}\right) & -(\gamma-1)u_1 u_2 & \gamma u_1 \end{bmatrix} \tag{A.25}$$

The Euler flux Jacobian matrix $\boldsymbol{A}_2$ can be derived in the same way and the full derivation of this matrix has not be included here. The final form of the $\boldsymbol{A}_2$ Jacobian is defined as:

$$\boldsymbol{A}_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -u_1 u_2 & u_2 & u_1 & 0 \\ (\gamma-1)\frac{u^2}{2} - u_2^2 & u_1 - \gamma u_1 & 3u_2 - \gamma u_2 & \gamma - 1 \\ u_2\left[(\gamma-1)u^2 - \gamma E\right] & -(\gamma-1)u_1 u_2 & \gamma E - (\gamma-1)\left(u_2^2 + \frac{u^2}{2}\right) & \gamma u_2 \end{bmatrix} \tag{A.26}$$

where $u^2 = u_1^2 + u_2^2$

The two-dimensional viscous fluxes are defined as:

$$\boldsymbol{G}_1 = \begin{pmatrix} 0 \\ \boldsymbol{\tau}_{11} \\ \boldsymbol{\tau}_{12} \\ u_1\boldsymbol{\tau}_{11} + u_2\boldsymbol{\tau}_{12} - q_1 \end{pmatrix}, \quad \boldsymbol{G}_2 = \begin{pmatrix} 0 \\ \boldsymbol{\tau}_{21} \\ \boldsymbol{\tau}_{22} \\ u_1\boldsymbol{\tau}_{21} + u_2\boldsymbol{\tau}_{22} - q_2 \end{pmatrix} \tag{A.27}$$

The components of the viscous tensor $\boldsymbol{K}_{ij}$ can be derived starting from the following definitions:

$$\boldsymbol{G}_1 = \boldsymbol{K}_{11}\frac{\partial \boldsymbol{U}}{\partial x_1} + \boldsymbol{K}_{12}\frac{\partial \boldsymbol{U}}{\partial x_2} \tag{A.28}$$

$$\boldsymbol{G}_2 = \boldsymbol{K}_{21}\frac{\partial \boldsymbol{U}}{\partial x_1} + \boldsymbol{K}_{22}\frac{\partial \boldsymbol{U}}{\partial x_2} \tag{A.29}$$

The rows of the $\boldsymbol{K}_{11}$ and $\boldsymbol{K}_{12}$ tensors are derived as follows:

$$\boldsymbol{G}_1(1) = 0 \tag{A.30}$$

$$\begin{aligned} \boldsymbol{G}_1(2) &= \tfrac{4}{3}\mu\tfrac{\partial u_1}{\partial x_1} - \tfrac{2}{3}\mu\tfrac{\partial u_2}{\partial x_2} \\ &= \tfrac{4}{3}\tfrac{\mu}{\rho}\left(\tfrac{\partial \rho u_1}{\partial x_1} - u_1\tfrac{\partial \rho}{\partial x_1}\right) - \tfrac{2}{3}\tfrac{\mu}{\rho}\left(\tfrac{\partial \rho u_2}{\partial x_2} - u_2\tfrac{\partial \rho}{\partial x_2}\right) \\ &= \left(-\tfrac{4}{3}\tfrac{\mu u_1}{\rho}\right)\tfrac{\partial \rho}{\partial x_1} + \left(\tfrac{4}{3}\tfrac{\mu}{\rho}\right)\tfrac{\partial \rho u_1}{\partial x_1} + \left(\tfrac{2}{3}\tfrac{\mu u_2}{\rho}\right)\tfrac{\partial \rho}{\partial x_2} + \left(-\tfrac{2}{3}\tfrac{\mu}{\rho}\right)\tfrac{\partial \rho u_2}{\partial x_2} \end{aligned} \tag{A.31}$$

$$\begin{aligned}
\boldsymbol{G}_1(3) &= \mu\frac{\partial u_1}{\partial x_2} + \mu\frac{\partial u_2}{\partial x_1} \\
&= \frac{\mu}{\rho}\left(\frac{\partial \rho u_1}{\partial x_2} - u_1\frac{\partial \rho}{\partial x_2}\right) + \frac{\mu}{\rho}\left(\frac{\partial \rho u_2}{\partial x_1} - u_2\frac{\partial \rho}{\partial x_1}\right) \\
&= \left(\frac{\mu u_2}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(\frac{\mu}{\rho}\right)\frac{\partial \rho u_2}{\partial x_1} + \left(-\frac{\mu u_1}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(\frac{\mu}{\rho}\right)\frac{\partial \rho u_1}{\partial x_2}
\end{aligned} \tag{A.32}$$

$$\boldsymbol{G}_1(4) = \left(\tfrac{4}{3}\mu\frac{\partial u_1}{\partial x_1} - \tfrac{2}{3}\mu\frac{\partial u_2}{\partial x_2}\right)u_1 + \left(\mu\frac{\partial u_1}{\partial x_2} + \mu\frac{\partial u_2}{\partial x_1}\right)u_2 - \kappa\frac{\partial T}{\partial x_1} \tag{A.33}$$

now define $\frac{\partial T}{\partial x_1}$ noting that $T = e/c_{\mathrm{v}}$

$$\begin{aligned}
\frac{\partial T}{\partial x_1} &= \frac{1}{c_{\mathrm{v}}}\frac{\partial e}{\partial x_1} \\
&= \frac{1}{c_{\mathrm{v}}}\frac{\partial\left[E - \frac{1}{2}\left(u_1^2+u_2^2\right)\right]}{\partial x_1} \\
&= \frac{1}{c_{\mathrm{v}}}\frac{\partial E}{\partial x_1} - \frac{1}{2c_{\mathrm{v}}}\frac{\partial\left(u_1^2+u_2^2\right)}{\partial x_1} \\
&= \frac{1}{c_{\mathrm{v}}}\frac{\partial E}{\partial x_1} - \frac{1}{2c_{\mathrm{v}}}\left[\frac{\partial\left(u_1^2+u_2^2\right)}{\partial u_1}\frac{\partial u_1}{\partial x_1} + \frac{\partial\left(u_1^2+u_2^2\right)}{\partial u_2}\frac{\partial u_2}{\partial x_1}\right] \\
&= \frac{1}{c_{\mathrm{v}}}\left[\frac{\partial E}{\partial x_1} - u_1\frac{\partial u_1}{\partial x_1} - u_2\frac{\partial u_2}{\partial x_1}\right] \\
&= \frac{1}{c_{\mathrm{v}}}\left[\frac{\partial \rho E}{\partial x_1} - E\frac{\partial \rho}{\partial x_1} - u_1\frac{\partial \rho u_1}{\partial x_1} + u_1^2\frac{\partial \rho}{\partial x_1} - u_2\frac{\partial \rho u_2}{\partial x_1} + u_2^2\frac{\partial \rho}{\partial x_1}\right] \\
&= \left[\frac{1}{c_{\mathrm{v}}\rho}\left(u_1^2+u_2^2-E\right)\right]\frac{\partial \rho}{\partial x_1} + \left[\frac{-u_1}{c_{\mathrm{v}}\rho}\right]\frac{\partial \rho}{\partial x_1} + \left[\frac{-u_2}{c_{\mathrm{v}}\rho}\right]\frac{\partial \rho}{\partial x_1} + \left[\frac{1}{c_{\mathrm{v}}\rho}\right]\frac{\partial \rho}{\partial x_1}
\end{aligned} \tag{A.34}$$

then the fourth element of the viscous flux vector $\boldsymbol{G}_1$ may be defined as

$$\begin{aligned}
\boldsymbol{G}_1(4) &= \left(-\tfrac{4}{3}\frac{\mu u_1^2}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(\tfrac{4}{3}\frac{\mu u_1}{\rho}\right)\frac{\partial \rho u_1}{\partial x_1} + \left(\tfrac{2}{3}\frac{\mu u_1 u_2}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(-\tfrac{2}{3}\frac{\mu u_1}{\rho}\right)\frac{\partial \rho u_2}{\partial x_2} \\
&\quad + \left(\frac{\mu u_2^2}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(\frac{\mu u_2}{\rho}\right)\frac{\partial \rho u_2}{\partial x_1} + \left(-\frac{\mu u_1 u_2}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(\frac{\mu u_2}{\rho}\right)\frac{\partial \rho u_1}{\partial x_2} \\
&\quad + \left[\frac{-\kappa}{c_{\mathrm{v}}\rho}\left(u_1^2+u_2^2-E\right)\right]\frac{\partial \rho}{\partial x_1} + \left[\frac{\kappa u_1}{c_{\mathrm{v}}\rho}\right]\frac{\partial \rho}{\partial x_1} + \left[\frac{\kappa u_2}{c_{\mathrm{v}}\rho}\right]\frac{\partial \rho}{\partial x_1} + \left[\frac{-\kappa}{c_{\mathrm{v}}\rho}\right]\frac{\partial \rho}{\partial x_1}
\end{aligned} \tag{A.35}$$

Thus, the two-dimensional form of the viscous tensor $\boldsymbol{K}_{11}$ and $\boldsymbol{K}_{12}$ are defined as:

$$\boldsymbol{K}_{11} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{4}{3}\frac{\mu u_1}{\rho} & \frac{4}{3}\frac{\mu}{\rho} & 0 & 0 \\ -\frac{\mu u_2}{\rho} & 0 & \frac{\mu}{\rho} & 0 \\ \boldsymbol{K}_{11}(4,1) & \frac{4}{3}\frac{\mu u_1}{\rho} + \frac{\kappa u_1}{c_{\mathrm{v}}\rho} & \frac{\mu u_2}{\rho} + \frac{\kappa u_2}{c_{\mathrm{v}}\rho} & -\frac{\kappa}{c_{\mathrm{v}}\rho} \end{bmatrix} \tag{A.36}$$

where

$$\boldsymbol{K}_{11}(4,1) = -\frac{4}{3}\frac{\mu u_1^2}{\rho} - \frac{\mu u_2^2}{\rho} - \frac{\kappa}{c_{\mathrm{v}}\rho}\left(u_1^2+u_2^2-E\right)$$

$$\boldsymbol{K}_{12} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \frac{2}{3}\frac{\mu u_2}{\rho} & 0 & -\frac{2}{3}\frac{\mu}{\rho} & 0 \\ -\frac{\mu u_1}{\rho} & \frac{\mu}{\rho} & 0 & 0 \\ -\frac{1}{3}\frac{\mu u_1 u_2}{\rho} & \frac{\mu u_2}{\rho} & -\frac{2}{3}\frac{\mu u_1}{\rho} & 0 \end{bmatrix} \tag{A.37}$$

The rows of the $\boldsymbol{K}_{21}$ and $\boldsymbol{K}_{22}$ tensors are derived as follows:

$$\boldsymbol{G}_2(1) = 0 \tag{A.38}$$

$$\begin{aligned} \boldsymbol{G}_2(3) &= \mu\frac{\partial u_1}{\partial x_2} + \mu\frac{\partial u_2}{\partial x_1} \\ &= \frac{\mu}{\rho}\left(\frac{\partial \rho u_1}{\partial x_2} - u_1\frac{\partial \rho}{\partial x_2}\right) + \frac{\mu}{\rho}\left(\frac{\partial \rho u_2}{\partial x_1} - u_2\frac{\partial \rho}{\partial x_1}\right) \\ &= \left(\frac{\mu u_2}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(\frac{\mu}{\rho}\right)\frac{\partial \rho u_2}{\partial x_1} + \left(-\frac{\mu u_1}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(\frac{\mu}{\rho}\right)\frac{\partial \rho u_1}{\partial x_2} \end{aligned} \tag{A.39}$$

$$\begin{aligned} \boldsymbol{G}_2(2) &= \tfrac{4}{3}\mu\frac{\partial u_2}{\partial x_2} - \tfrac{2}{3}\mu\frac{\partial u_1}{\partial x_1} \\ &= \tfrac{4}{3}\frac{\mu}{\rho}\left(\frac{\partial \rho u_2}{\partial x_2} - u_2\frac{\partial \rho}{\partial x_2}\right) - \tfrac{2}{3}\frac{\mu}{\rho}\left(\frac{\partial \rho u_1}{\partial x_1} - u_1\frac{\partial \rho}{\partial x_1}\right) \\ &= \left(\tfrac{2}{3}\frac{\mu u_1}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(-\tfrac{2}{3}\frac{\mu}{\rho}\right)\frac{\partial \rho u_1}{\partial x_1} + \left(-\tfrac{4}{3}\frac{\mu u_2}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(\tfrac{4}{3}\frac{\mu}{\rho}\right)\frac{\partial \rho u_2}{\partial x_2} \end{aligned} \tag{A.40}$$

$$\boldsymbol{G}_2(4) = \left(\mu\frac{\partial u_1}{\partial x_2} + \mu\frac{\partial u_2}{\partial x_1}\right)u_1 + \left(\tfrac{4}{3}\mu\frac{\partial u_2}{\partial x_2} - \tfrac{2}{3}\mu\frac{\partial u_1}{\partial x_1}\right)u_2 - \kappa\frac{\partial T}{\partial x_2} \tag{A.41}$$

where the temperature gradient $\frac{\partial T}{\partial x_2}$ is defined as

$$\frac{\partial T}{\partial x_2} = \left[\frac{1}{c_v\rho}\left(u_1^2 + u_2^2 - E\right)\right]\frac{\partial \rho}{\partial x_2} + \left[\frac{-u_1}{c_v\rho}\right]\frac{\partial \rho}{\partial x_2} + \left[\frac{-u_2}{c_v\rho}\right]\frac{\partial \rho}{\partial x_2} + \left[\frac{1}{c_v\rho}\right]\frac{\partial \rho}{\partial x_2} \tag{A.42}$$

then the fourth element of the viscous flux vector $\boldsymbol{G}_2$ may be defined as

$$\begin{aligned} \boldsymbol{G}_2(4) &= \left(\frac{\mu u_1 u_2}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(\frac{\mu u_1}{\rho}\right)\frac{\partial \rho u_2}{\partial x_1} + \left(\frac{-\mu u_1^2}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(\frac{\mu u_1}{\rho}\right)\frac{\partial \rho u_1}{\partial x_2} \\ &+ \left(\tfrac{2}{3}\frac{\mu u_1 u_2}{\rho}\right)\frac{\partial \rho}{\partial x_1} + \left(-\tfrac{2}{3}\frac{\mu u_2}{\rho}\right)\frac{\partial \rho u_1}{\partial x_1} + \left(-\tfrac{4}{3}\frac{\mu u_2^2}{\rho}\right)\frac{\partial \rho}{\partial x_2} + \left(\tfrac{4}{3}\frac{\mu u_2}{\rho}\right)\frac{\partial \rho u_2}{\partial x_2} \\ &+ \left[\frac{-\kappa}{c_v\rho}\left(u_1^2 + u_2^2 - E\right)\right]\frac{\partial \rho}{\partial x_2} + \left[\frac{\kappa u_1}{c_v\rho}\right]\frac{\partial \rho}{\partial x_2} + \left[\frac{\kappa u_2}{c_v\rho}\right]\frac{\partial \rho}{\partial x_2} + \left[\frac{-\kappa}{c_v\rho}\right]\frac{\partial \rho}{\partial x_2} \end{aligned} \tag{A.43}$$

Thus, the two-dimensional form of the viscous tensor $\boldsymbol{K}_{21}$ and $\boldsymbol{K}_{22}$ are defined as:

$$\boldsymbol{K}_{21} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{\mu u_2}{\rho} & 0 & \frac{\mu}{\rho} & 0 \\ \frac{2}{3}\frac{\mu u_1}{\rho} & -\frac{2}{3}\frac{\mu}{\rho} & 0 & 0 \\ -\frac{1}{3}\frac{\mu u_1 u_2}{\rho} & -\frac{2}{3}\frac{\mu u_2}{\rho} & \frac{\mu u_1}{\rho} & 0 \end{bmatrix} \tag{A.44}$$

$$\boldsymbol{K}_{22} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -\frac{\mu u_1}{\rho} & \frac{\mu}{\rho} & 0 & 0 \\ -\frac{4}{3}\frac{\mu u_2}{\rho} & 0 & \frac{4}{3}\frac{\mu}{\rho} & 0 \\ \boldsymbol{K}_{11}(4,1) & \frac{\mu u_1}{\rho} + \frac{\kappa u_1}{c_{\mathrm{v}}\rho} & \frac{4}{3}\frac{\mu u_2}{\rho} + \frac{\kappa u_2}{c_{\mathrm{v}}\rho} & \frac{-\kappa}{c_{\mathrm{v}}\rho} \end{bmatrix} \tag{A.45}$$

where

$$\boldsymbol{K}_{22}(4,1) = -\frac{\mu u_1^2}{\rho} - \frac{4}{3}\frac{\mu u_2^2}{\rho} - \frac{\kappa}{c_{\mathrm{v}}\rho}\left(u_1^2 + u_2^2 - E\right)$$