5-2013

Improvement in Computational Fluid Dynamics Through Boundary Verification and Preconditioning

David Folkner

IMPROVEMENT IN COMPUTATIONAL FLUID DYNAMICS THROUGH BOUNDARY VERIFICATION AND PRECONDITIONING

by

David E. Folkner

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Mechanical Engineering

Approved:

Dr. Aaron Katz Major Professor	Dr. Robert E. Spall Committee Member
Dr. Heng Ban Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2013

Abstract

Improvement in Computational Fluid Dynamics Through Boundary Verification and Preconditioning

by

David E. Folkner, Master of Science

Utah State University, 2013

Major Professor: Dr. Aaron Katz
Department: Mechanical and Aerospace Engineering

This thesis provides improvements to computational fluid dynamics accuracy and efficiency through two main methods: a new boundary condition verification procedure and preconditioning techniques.

First, a new verification approach that addresses boundary conditions was developed. In order to apply the verification approach to a large range of arbitrary boundary conditions, it was necessary to develop unifying mathematical formulation. A framework was developed that allows for the application of Dirichlet, Neumann, and extrapolation boundary condition, or in some cases the equations of motion directly. Verification of boundary condition techniques was performed using exact solutions from canonical fluid dynamic test cases.

Second, to reduce computation time and improve accuracy, preconditioning algorithms were applied via artificial dissipation schemes. A new convective upwind and split pressure (CUSP) scheme was devised and was shown to be more effective than traditional preconditioning schemes in certain scenarios. The new scheme was compared with traditional schemes for unsteady flows for which both convective and acoustic effects dominated.

Both boundary conditions and preconditioning algorithms were implemented in the context of a "strand grid" solver. While not the focus of this thesis, strand grids provide automatic viscous quality meshing and are suitable for moving mesh overset problems.

(105 pages)

Public Abstract

Improvement in Computational Fluid Dynamics Through Boundary Verification and Preconditioning

This thesis focuses on improvements to numerical simulation of fluid dynamic problems. A study of computation fluid dynamics is done to improve simulation accuracy and efficiency. Accurate and fast results are the desire of simulation techniques. Two main improvements to current CFD methods are implemented and tested: a new boundary condition verification procedure and a new preconditioning technique.

First, mathematical code verification methods are extended to also include boundary conditions that can often drive fluid dynamics problems. A new framework is created to implement a variety of desirable boundaries. With this framework, a robust verification procedure for these boundaries is presented. Testing of the new procedure is completed by using exact analytic solutions.

Second, preconditioning of the numeric waves is used to increase computational efficiency and solution accuracy. A new scheme is introduced and tested with old schemes and is shown to have better accuracy characteristics for low speed unsteady flows.

Both boundary conditions and preconditioning algorithms were implemented in the context of a "strand grid" solver. While not the focus of this thesis, strand grids provide automatic viscous quality meshing and are suitable for moving mesh overset problems.

David E. Folkner, Master of Science

Acknowledgments

Much of the research for this work was supported by the Army Research Office (ARO), under the supervision of Dr. Frederick Ferguson. I would like to thank Dr. Ferguson for his continuing support of this research. A portion of the material presented in this thesis is a product of the CREATE-AV Element of the Computational Research and Engineering for Acquisition Tools and Environments (CREATE) Program sponsored by the U.S. Department of Defense HPC Modernization Program Office. Dr. Robert Meakin is the program manager for CREATE-AV.

The research was largely supported by the aid of several selfless people who took the time to instruct and advise the work. These people are Dr. Aaron Katz and Dr. Venke Sankaran.

David Folkner

Contents

List of Tables

List of Figures

Notation

General

ρ	Density
u	Particle velocity in X direction
v	Particle velocity in Y direction
p	Pressure
h	Enthalpy
E	Total or stagnation energy
h^0	Total or stagnation enthalpy
T	Temperature
s	Entropy
t	Physical Time
τ	Pseudo Time
c	Acoustic sound speed
A	Face area
A_i	Directed area (A_x,A_y)
n_i	Face normal vector (n_x,n_y)
V	Volume
M	Mach number
Q	Conserved variable vector
Q_v	Primitive variable vector
F	Inviscid flux vector (in X)
G	Inviscid flux vector (in Y)
F^v	Viscous flux vector

Chapter 2

$\mathcal{F}$	Area wighted inviscid flux
$\hat{\mathcal{F}}$	Numeric inviscid flux
D	Artificial dissipation
R	Discretization residual
R_b	Boundary discretization residual
s	Limiter (0 to 1)
Q_L	Left State
Q_R	Right State
Q_E	Extrapolated Boundary state
k	Cell face

Chapter 3

u_n	Normal velocity
u_t	Tangential velocity
B	Boundary Equations
b	Boundary specified values
N	Lagrange specification matrix
$\tilde{Q}$	Some set of variables (entropy, primitive...)
S	Manufactured source terms
S_b	Manufactured boundary source terms
λ	Lagrange multiplier
Ω	Boundary equations or dimensionless frequency parameter
Ω_D	Dirichlet boundary equations
Ω_N	Neumann boundary equations
Ω_E	Extrapolation boundary equations
L	Boundary selection matrix
R	Rotation matrix
M_n	Left Eigenvectors
M_n^{-1}	Right Eigenvectors
S_p	Quasi-1D Euler source terms

Chapter 4

Γ	Primitive variable Jacobian
Γ_p	Primitive variable preconditioner
$\mathcal{A}$	Flux Jacobian
q_a	Area weighted velocity
ρ_p	Derivative of variable with respect to pressure
ρ'_p	Substitute preconditioned value
V_p	Preconditioned sound speed
V_p^s	Steady preconditioned sound speed
V_p^u	Unsteady preconditioned sound speed
Str	Strouhal Number
X	Left Eigenvectors
X^{-1}	Right Eigenvectors
Δ	Right - Left state or other parameter
α	CUSP parameter
β	CUSP parameter
M_a	Area weighted Mach number
M_e	Effective Mach number
f_p	Pressure vector
CFL	Dimensional time parameter
γ	Modified CUSP term or Ideal gas constant
Ω	Frequency control parameter
ω	Vorticity or frequency
ϵ	Magnitude of oscillations
T_w	Time period
L_w	Length period

Acronyms

AUSM	Advection Upstream Splitting Method
CFD	Computational Fluid Dynamics
CREATE	Computational Research and Engineering Acquisition Tools and Environments
CUSP	Convective Upwind and Split Pressure
DoD	Department of Defense
LED	Local Extremum Diminishing
LHS	Left Hand Side
MMS	Method of Manufactured Solutions
RHS	Right Hand Side

Chapter 1

Introduction

As opposed to theoretical or experimental approaches, computational fluid dynamics (CFD) seeks to numerically simulate and analyze fluid flow. CFD uses algorithms and iterative numerical methods to solve for flow behavior in a variety of applications. Research in the other branches of fluid dynamics–theory and experimentation–provide further perspective for complex flows. All three perspectives provide tools for design and increased understanding of flow physics.

CFD is the newest and fastest growing sub-discipline of fluid dynamics. Though CFD is a relatively new field, it has revolutionized many areas of engineering. With the advent of faster and cheaper computing power that is available to more engineers, CFD has grown in its usefulness in the design process. Its ability to provide economic analysis has aided its recent rise. For example, CFD is used extensively for aerodynamic analysis to predict airloads for drag reduction and improved control. Biomedical engineers use CFD to model fluid processes within the human body, leading to new devices and treatments. CFD is used to analyze combustion flows in order to reduce expensive and difficult experimentation.

While CFD has already provided enormous impact in science and engineering, many research issues remain to improve simulation capabilities for complex flows. The aim of this thesis is to explore some of these issues - to improve computational efficiency, accuracy of results, and automation of problem setup. Specifically, boundary condition formulation and verification, as well as preconditioning techniques are examined at a fundamental level. Algorithms are demonstrated using strand grids, which are designed to automate mesh generation and grid assembly for complex overset simulations. Strand grids are discussed in detail in sections 1.1.1 and 2.1. The studies conducted here provide fundamental research to support a fully three-dimensional strand grid solver used within the Department of De-

fense (DoD) Computational Research and Engineering Acquisition Tools and Environments (CREATE) program. However, the methods are general, and can be extended to a variety of other CFD applications.

The following sections outline a few current challenges in CFD and provide context for the improvements made in this research.

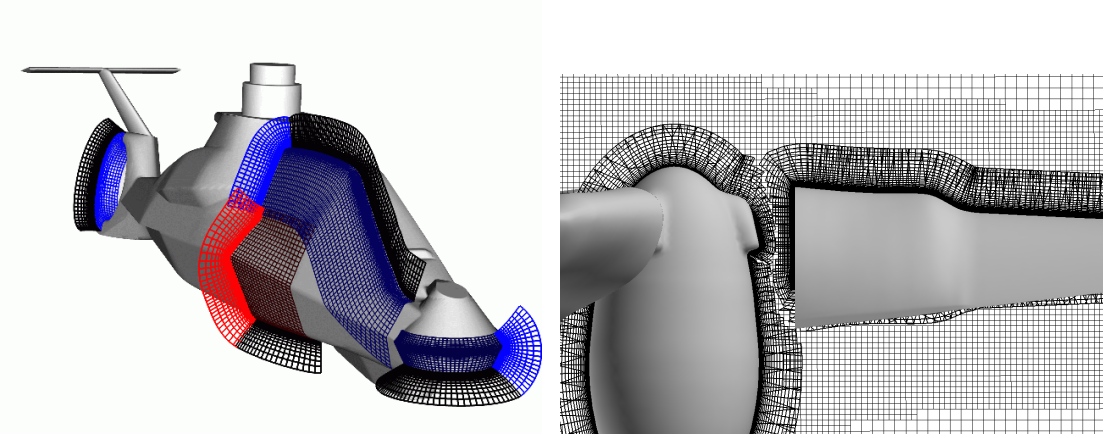
1.1 Challenges in CFD

To motivate the research performed in this thesis, a discussion of three challenges in CFD – automation, efficiency, and accuracy – is given below in the context of rotorcraft simulation. High-fidelity CFD for rotorcraft is an emerging application that faces these challenges and is a critical focus area of the DoD CREATE program. Obtaining detailed flow fields of rotorcraft using CFD encounters many difficulties. First, the complex geometry of rotorcraft demands a high level of automation in grid generation and problem configuration since manual setup, as is currently done, is very time consuming and expensive. Second, widely varying length scales and complex vortex interactions intensify the need for increased accuracy and efficiency. Third, the unsteady nature of rotorcraft flow fields requires extremely efficient algorithms, especially for complex overset mesh systems that move with time. These challenges are discussed below in greater detail in the context of rotorcraft flows.

1.1.1 Automation

As computational power increases, a larger percentage of CFD solution time is being spent by engineers configuring the problem. One of the major configuration tasks is mesh generation. In CFD the mesh refers to the physical splitting of the domain into cells or nodes. Discretized conservation laws derived from first principles are then applied to each cell. To resolve the flow dynamics fully, the mesh must be able to capture the physical geometry as well as the flow interactions at critical regions.

Sample meshes for rotorcraft are shown in Fig. 1.1. Traditionally, creating meshes of high quality takes time and significant expertise. Fig. 1.1(a) shows a portion of an overset



(a) A sample of a Comanche rotorcraft body with chimera mesh as provided by Meakin [3].

(b) Strand/Cartesian grid for TRAM rotor [4].

Fig. 1.1: Sample rotorcraft meshes for various parts of a typical set up.

mesh that was created using Chimera Grid Tools developed at NASA Ames Research Center by William Chan et al. [3]. The process in which these overset meshes are created is robust, but time intensive. Each block of the overset grid system must be manually input into the global mesh description. Sufficient resolution and overlap between grid components must be maintained. Experts using Chimera Grid Tools may spend several weeks generating complex mesh systems for rotorcraft.

In light of this example, methods that simplify and speed up the mesh generation process are a critical need in CFD. Along these lines, one goal of this thesis is to provide fundamental research to support the development of a fully functional, three-dimensional strand grid solver for complex, dynamic geometry. Unlike traditional approaches, strand grids provide fully automated volume mesh generation. Fig. 1.1(b) shows an automatically generated near-body strand mesh with an overset Cartesian off body mesh. Strand grids are automatically generated projections of the surface mesh along normal directions of the physical surface that extend outward a short distance. The strands have a uniform structured 1D node distribution along the length of each strand. Strands are organized in an unstructured manner parallel to the surface boundary. Fig. 1.2 shows a sample strand grid around a 2D NACA 0012 airfoil. The strands protrude normal to the airfoil surface. Each rectangle represents a prismatic cell created by the strand technique.

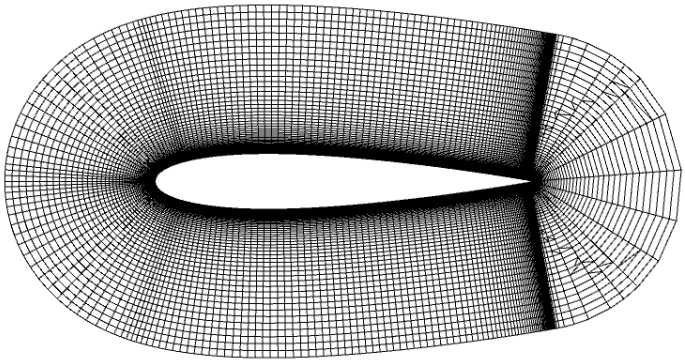


Fig. 1.2: A sample 2D strand mesh around a NACA 0012 airfoil.

Along with full automation of volume mesh generation, strand grids provide other advantages. Strands have a benefit in computer storage. Traditional unstructured meshes must store the locations and connectivity information for every node in the domain. For problems with many nodes, such as a fully modeled rotorcraft, this storage can become a bottleneck in the solution process. Due to strands having the same 1D node distribution, the amount of storage is reduced dramatically. The benefits of smaller mesh storage can be easily observed in parallel computing. Many parallel applications have large communication overheads when sharing information from different mesh locations, whereas strand grids can be stored so efficiently that each processor is capable of storing the entire domain. This reduces the need for expensive inter-domain communication. A more thorough description of strand grids is given in section 2.1.

1.1.2 Efficiency

Efficiency in CFD is used to describe the rate at which computations can be completed. Despite the increasing availability of cheap computational resources, maintaining efficiency

is critically important for CFD analysis. Unsteady rotorcraft simulation requires that the entire domain fully converge at each time step, otherwise errors accrue and grow. For simulations that require a large number of time steps, it is essential to optimize the convergence for each time step.

Rotorcraft have complex flow fields that include many different flow regions. It is possible for a single rotorcraft simulation to have transonic regions, viscous boundary layers, stagnation points, low Mach number flows, etc. Traditional time-marching CFD algorithms are capable of efficiently capturing many of these different flow regions at mid range Mach numbers, but struggles when Mach numbers are low. Low Mach numbers cause the numeric error propagation to be stiff, drastically hindering the efficiency of the methods.

The issue of numeric stiffness can extend from a difference in velocity scales through the domain. These velocity scales represent the physical motion of the fluid as well as the acoustic propagation that travel at the sound speed. At low Mach numbers, these scales can vary by large orders of magnitude. By sufficiently resolving the faster waves, the slower waves propagate inefficiently through the domain. This effect causes the convergence to suffer. Ensuring the varying wave speeds are of the same magnitude can be accomplished through a method called preconditioning. Preconditioners introduce new terms into the numerics that effect the wave speeds while not changing the actual dynamics of the flow field.

Implementing preconditioners for steady state problems is a straightforward process. Time accuracy is not needed and the adjustments made to the wave speeds have little to no effect on the accuracy of the problem. Unsteady rotorcraft simulations, however, require time accuracy in the solution, which can be a struggle for traditional preconditioning schemes. For these unsteady cases, a more robust preconditioning algorithm must be implemented to increase efficiency for stiff problems whilst maintaining the time accuracy.

1.1.3 Accuracy

Although very tightly coupled with efficiency and automation, accuracy is a current focus in CFD research. Increasing the accuracy of CFD methods can reduce the number of

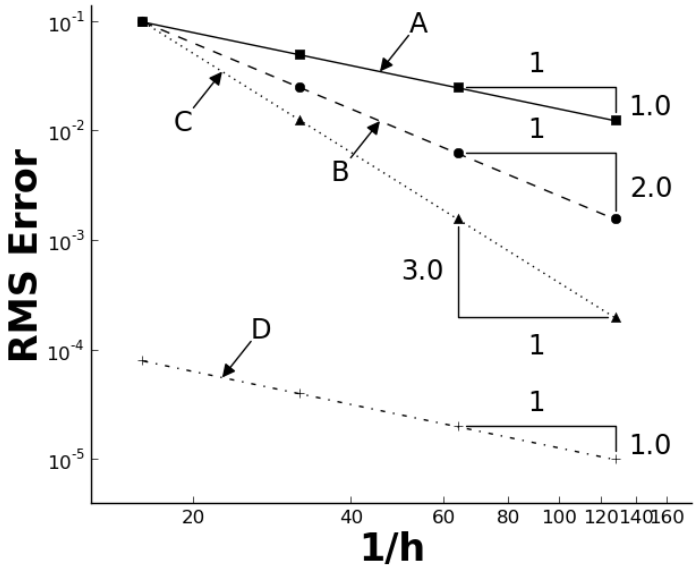


Fig. 1.3: An example of order of accuracy plots with different slopes and starting points.

grid points required to fully resolve the problem, increasing efficiency of both computation and setup. Increased accuracy gives the user greater insight to the behavior of the fluid dynamics. Higher-order methods are currently being researched heavily in literature. They are able to accomplish reduction in error, but it is possible to reduce the error without changing the order of a scheme. Fig. 1.3 shows several sample order of accuracy plots. Although line C is third-order accurate and line D is only first-order, line D clearly has less error. The preconditioning scheme discussed in the previous section is capable of increasing the accuracy in this manner for a standard second-order finite volume scheme.

In simple scenarios, it is often possible to measure the exact error with an analytic solution. Rotorcraft simulations have enough complexity that measuring the model's accuracy becomes challenging. Inaccurate CFD results give no benefit to the user and can often mislead the user to draw incorrect conclusions. There is a need to ensure accuracy of CFD simulations. One such method is mathematical verification of the algorithms. Verification is done by imposing known functions as the solution. With a known solution, the error can be measured and the order of accuracy determined. This is known as the Method of

Manufactured Solutions (MMS).

With all of the research that exists for improved accuracy and verification on the interior domain of a problem, there is a lack of work on the boundaries. Boundaries are often the critical regions in many CFD applications. Several examples include viscous flow regions around a rotor blade, heat transfer problems near surfaces, and many mass flux problems. It is imperative to implement a robust generalized method to verify the proper implementation and mathematical correctness of arbitrary boundary conditions. This work proposes a new method of boundary verification as well as a general framework to unify the many boundary implementation methods that exist.

1.2 Objectives

To address the issues presented in the previous section, this section outlines the overall objectives of this thesis.

1. Determine a robust and general method for boundary condition verification.

2. Devise a general method to unify boundary implementation for arbitrary formulations with arbitrary spacial discretization methods.

3. Formulate a preconditioning scheme with improved accuracy for acoustic and convective unsteady problems.

1.3 Summary of previous work done in the literature

The challenges in CFD are numerous and much work has been done in preparation for the objectives outlined above. Summarizing the work others have accomplished sheds light on the research topics covered in this thesis. This section focuses on this previously done work to give background to the improvement proposed here.

1.3.1 Strand Grids

When approaching meshing with strand grids, the problem is typically divided into two main parts: near-body domain and off-body domain. Strands grid technology is used

to automatically generate the near-body mesh where Cartesian grids are used in the off-body domain. The different grid technologies interact through an over-set Chimera-like system [5, 6]. The near-body strands are useful for resolving boundary layers as well as providing an automated mesh generation [7]. A more detailed summary of strand grid technology details can be found in section 2.1.

Despite the advantages in automation and near-body resolution, the novelty of strand grids requires in depth validation of their capabilities. Wissink et al. [8,9] have run a number of validation cases including Reynolds averaged and detached eddy simulation turbulence with a focus on robust grid refinement. Katz et al. [2] and Work et al. [4] have focused on allowing the strands to capture more complex geometries such as sharp corners through strand clipping and smoothing. Recently, Work proposed a new solver specifically designed to better handle these more complex features of strand grids. This new solver is the base on which the strand solvers for this thesis were developed. Further detail on the numerics and discretization of this chapter can be found in Chapter 2.

1.3.2 Verification

In CFD research, many new numerics routines and algorithms are constantly being implemented. In order to ensure their accuracy and test the code's integrity, verification is required. Roache [10], Roy [11], and Veluri [12] showed how MMS can be used to verify the desired order of accuracy of the scheme with the actual order of accuracy the code produces. They demonstrated that verification using MMS is capable of identifying malformed algorithms and general coding "bugs."

Many researchers have extended MMS to verify complex problems. Diskin et al. [13–15] have employed MMS extensively to compare and test different second order schemes as well as study the effect irregular grids have on accuracy. Many others have developed MMS methodologies for the interior solution only [16–21].

Though verification is necessary for the interior domain, many problems are dominated by the effect of the boundary conditions. In these simulations it is equally important to verify boundary implementation. Addressing this issue of boundary verification, Choudhary

et al. [22] recently proposed a boundary verification procedure using MMS. Their approach requires carefully constructed manufactured solutions that already satisfy the boundary conditions. Thus a new manufactured solution needs to be constructed for each boundary condition and geometry.

Despite the need for high accuracy near boundaries (or perhaps because of it), numerous boundary procedures have been proposed in literature. For example, many treatments have been proposed for simple inviscid wall boundaries [23–28]. Before introducing a generalized verification procedure for boundary conditions, it is first necessary to create a framework in which these widely varying boundary procedures can be implemented. Allmaras [28] demonstrated a method for combining the equations of motion with boundary equations for finite-element Dirichlet boundaries. This thesis extends Allmaras's work to create a more robust boundary implementation framework that works with node- and cell-centered finite volume methods. Chapter 3 outlines this new framework as well as a new generalized boundary verification technique.

1.3.3 Preconditioning

Nonlinearities in the governing equations of fluid dynamics make all CFD methods require iterative solvers. Time-marching CFD solvers rely on the unsteady time term in the governing equations. For steady-state, or time-independent solutions these converge by marching through a false time often referred to as "pseudo-time," τ. Errors are reduced through convective waves and dampening. When convergence is reached, these pseudo-time terms go to zero and the remaining solution represents the steady-state solution [29]. This can be seen in the simplified case of the inviscid 1D Euler equation presented in equations 1.1 - 1.3.

$$\frac{\partial Q}{\partial \tau} + \frac{\partial F}{\partial x} = 0 \tag{1.1}$$

$$\frac{\partial Q}{\partial \tau} + A\frac{\partial Q}{\partial x} = 0 \tag{1.2}$$

$$A = \frac{\partial F}{\partial Q}, \quad \lambda = \begin{Bmatrix} u \\ u + c \\ u - c \end{Bmatrix} \tag{1.3}$$

Here, A represents the Jacobian of flux and time terms, λ is a vector of the Eigenvalues of A, u is the particle velocity, and c represents the sound of speed or acoustic velocity. The waves that convect the errors out of the domain travel with velocities equal to the Eigenvalues found in λ. Since time-marching CFD algorithms rely on these convective waves to remove transient errors out of the domain, the wave behavior is important. If these different wave speeds vary greatly in magnitude, time-marching algorithms converge slowly.

In low speed flows (or low Mach number flows), particle waves can travel orders of magnitude slower than the acoustic sound waves. While properly capturing the acoustic wave, the particle-based waves travel at a much slower pace, reducing the algorithm's efficiency. Preconditioners add artificial terms to the pseudo-time derivatives. The goal of preconditioners is to change acoustic speeds in the problem to be scaled on the order of the particle wave speed. Many studies on these steady state preconditioners have been done [29–31].

Another situation that can interfere with time-marching methods efficiency is if the flow becomes incompressible. In this limit, the sound speed becomes infinite and it is impossible for a time-marching algorithm to converge. Chorin [32] introduced the idea of "artificial compressibility" to allow acoustic waves to travel as a function of pressure alone. This allows time-marching methods to converge in the incompressible regime.

More recently, Weiss et al. [33] and Merkle et al. [34] present preconditioning schemes that are able to increase efficiency in low Mach numbers as well as reduce to Chorin's artificial compressibility for truly incompressible flows. These preconditioning schemes are used as the basis for much of the work in this thesis. Chapter 4 goes into further detail on these schemes.

Unlike the steady state solutions described by Eq. 1.1 , unsteady solutions such as rotorcraft simulation, require time accuracy to be maintained. To add real time accuracy

to time-marching algorithms a dual time stepping process is used. Equation 1.1 becomes equation 1.4, where t represents the physical time.

$$\frac{\partial Q}{\partial t} + \frac{\partial Q}{\partial \tau} + \frac{\partial F}{\partial x} = 0 \tag{1.4}$$

To maintain accuracy, only the pseudo-time terms can be preconditioned. The steady-state preconditioners previously mentioned struggle to resolve pressure fields accurately and have reduced efficiency as flow speed becomes low and physical time steps become small. Sankaran [35] presented the introduction of the Strouhal number (Str), a dimensionless quantity that measures frequency and is inversely proportional to time step, to create an unsteady preconditioning scheme. Unsteady preconditioners have increased convergence rates and improved accuracy in pressure fields in low Mach and high Str number limit.

All time-marching schemes, regardless of preconditioning methods used, rely on "artificial dissipation" terms in the fluxes for stability. Details of why these terms are required can be found in section 2.2.1. Hosangadi et al. [1] demonstrated that performance of the unsteady preconditioning scheme, as far as efficiency and accuracy are concerned, is largely dependent on the artificial dissipation terms used.

The different artificial dissipation schemes are outlined in full detail in section 4.3. Only the effects they have on preconditioning are presented here. Hosangadi showed that through the implementation of Roe dissipation, a common dissipation scheme, unsteady preconditioners suffer to correctly resolve velocity fields. Advection Upstream Splitting Method (AUSM) has been combined with preconditioners to solve this issue.

An alternate to using the AUSM dissipation scheme is Convective Upwind and Split Pressure (CUSP) originally proposed by Jameson [36]. Chen and Zha [37, 38] proposed a modifications to CUSP to enable preconditioning. Though their results are promising, their formulation departs from the traditional form of CUSP, resulting in a AUSM like scheme. In this work, a preconditioned CUSP method is presented that maintains the traditional form of CUSP. This allows researchers more familiar with CUSP dissipation to implement preconditioning.

1.4 Outline and scope of Thesis

In this chapter, motivation and previous work have been presented to give background for the research performed in this thesis. In this section, the scope and a general outline for the rest of the thesis are stated.

In order to accomplish the objectives, the scope is limited to one- and two-dimensional finite volume problems for unstructured and strand meshes. The equations are limited to the inviscid Euler equations. The cases run are simplified and designed to highlight the areas of CFD in question. Although these scenarios may seem simple, they have a direct application in fully functional three-dimensional solvers.

In Chapter 2, more background is provided to support the remainder of the thesis. First, a strand grids are explained in further detail. Secondly, finite-volume discretization methods are explained and presented for strand grids. A focus is placed on face reconstructions for second order accuracy. Third, boundaries are discussed in preparation for the new techniques proposed by Chapter 3. Finally, the line-implicit Gauss-Seidel scheme is discussed as well as the dual time stepping unsteady procedure. The background provided here is an important base for understanding the rest of the work.

Chapter 3 focuses on objectives 1 and 2. This chapter outlines the methodologies used to create a unified framework and how interior MMS methods are extended to be effective for boundary conditions. Numerous different boundary conditions are tested and verified. Exact solutions in the cases of quasi-1D and Ringleb flow are used to validate the verification procedures. This is explored for both node- and cell- centered schemes. Traditional airfoils are also tested and validated for derived quantity boundaries.

Next, Chapter 4 discusses the studies on dissipation schemes and preconditioning for objective 3. An overview of the preconditioning schemes tested is given first. Then, a discussion of the different artificial dissipation models is presented. A newly developed preconditioned CUSP is introduced to improve accuracy for convective problems. The new method is compared to the old. A propagating inviscid vortex is used to isolate convective terms and a 1D oscillating back-pressure is used to highlight acoustic problems. The new

scheme is shown to be effective at properly capturing the flow dynamics of these cases in an efficient way.

Finally, general conclusions are found in Chapter 5. A discussion of how the research applies to and accomplishes the objectives is given. General results are discussed. Areas of future work are presented.

Chapter 2

Strand Grid Discretization and Solution Methods

Although discretization methods are not the main focus or research topic in question, they are fundamentally important to the numerical methods in this thesis. Despite the current research in higher-order algorithms, this work focuses on second-order finite volume for both node- and cell-centered schemes. The derivations in this section are applicable to generalized unstructured meshes as well as strand grids. Both mesh types are used extensively throughout this thesis.

This chapter presents the summarized strand grid techniques and how numerical methods are used to solve the governing equations for the physical domain, boundary conditions, and time-dependent terms. Strand grids are first introduced with a basic formulation. Then, a method for cell-centered spacial discretization using finite volume is summarized for the inviscid Euler equations. A focus is placed on stability and maintaining second order accuracy. Boundaries are briefly explore in preparation for the boundary verification research done in Chapter 3. Finally, line-implicit Gauss-Seidel schemes are explored.

2.1 Basics of Strand Grids

Strand grids are created from a surface geometry with n-sided polygon discretization. Tessellated geometries can be formed from any n-sided polygon, including triangles and quadrilaterals in 3D. A strand is created for each vertex of the discretized surface geometry. A unit normal vector is approximated at each surface vertex by averaging the normals of neighboring polygons. Each strand is placed so its origin, or root, coincides with the vertex and initially points in the direction of the corresponding surface normal. Every strand in the mesh share a length and user-defined 1D nodal distribution along the strand. Each strand has a clipping index to "shorten" the strand if the need arises. This is to say that nodes

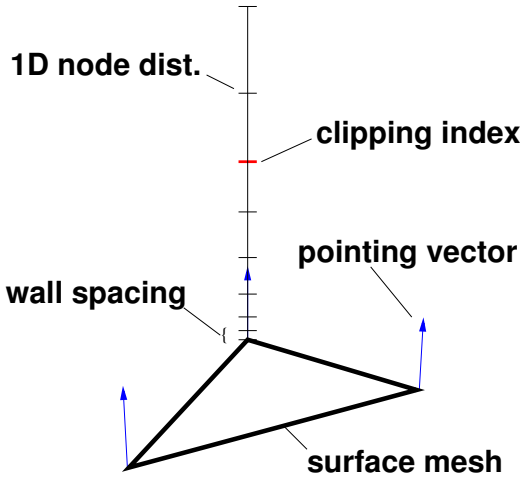


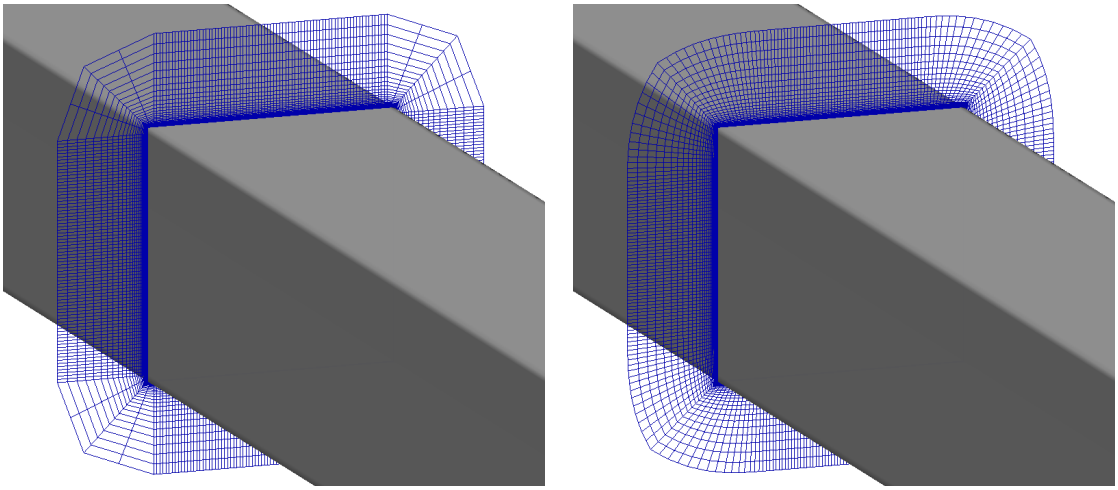
Fig. 2.1: An example of strand grid creation in 3D [2].

that lay beyond the clipping index are not valid physical nodes [2,7]. Fig. 2.1 demonstrates this procedure on a simplified surface triangle.

Defining strand grids in this fashion allows for a structured cells to be produced over the length of a strand. Cell-centered finite-volumes are created between neighboring strands. Inter-strand connectivity does not depend on the surface mesh, but is capable of re-clustering in an unstructured manner. The ability of strand grids to adapt to many different kinds of surface tessellations is a key factor in its use.

Sharp corners, both internal and external, present challenges for strand grids. At internal corners, strands tend to intersect and penetrate the surface geometry. Strands typically fail to have sufficient resolution at external corners. Strand smoothing can help to mitigate the issues presented by corners. Strands smoothing refers to a simple method of adjusting the surface normals to properly cover all regions of the domain. Fig. 2.2(a) shows a completely un-smoothed strand mesh around a square cylinder where Fig. 2.2(b) illustrates this same mesh after being smoothed. Work et al. [4] shows that smoothing is effective for resolving external corners.

Strand smoothing is typically insufficient for sharp internal corners. Even with smoothing, strands often intersect creating a negative volume element. Strand clipping is used to ensure that strand cells do not intersect or penetrate the surface geometry. An example of



(a) No Smoothing (b) Significant Smoothing

Fig. 2.2: Sample strand meshes around a square cylinder to illustrate how smoothing effects strands [2].

a smooth and clipped internal corner is in Fig. 2.3.

Strands grids typically only resolve the near-body and transitional mesh domains, where Cartesian meshes are used off the body. The current work focuses on fundamental numerics and only uses strands as a stand-alone meshing technique.

2.2 Cell-Centered Prismatic Spatial Discretization

This section focuses on the spacial discretization of the Navier-Stokes and Euler equations. The intent is not to fully derive the discrete equations, but to provide sufficient background for the reader to explore the further chapters.

2.2.1 Finite Volume Methods

The Navier-Stokes equations can be written as:

$$\frac{\partial Q}{\partial t} + \frac{\partial F_j}{\partial x_j} = \frac{\partial F_j^v}{\partial x_j} \tag{2.1}$$

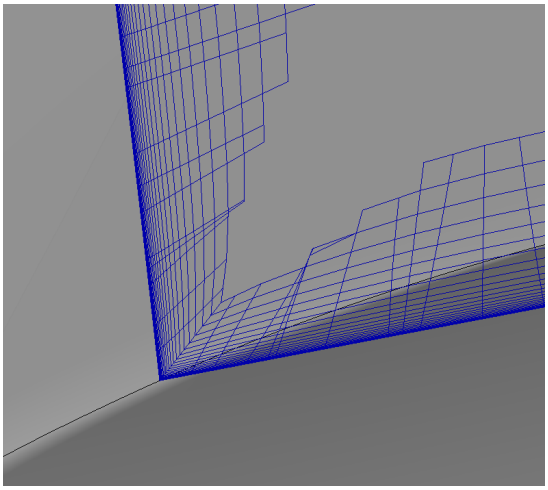


Fig. 2.3: Example of internal corner with smoothing and clipping [2].

where

$$Q = \begin{Bmatrix} \rho \\ \rho u_i \\ \rho E \end{Bmatrix}, \quad F_j = \begin{Bmatrix} \rho u_j \\ \rho u_i u_j + \delta_{ij} p \\ \rho u_j h \end{Bmatrix}, \quad F_j^v = \begin{Bmatrix} 0 \\ \sigma_{ij} \\ \sigma_{ij} u_i - q_j \end{Bmatrix} \tag{2.2}$$

Here, F_j is the inviscid flux vector, F_j^v is the viscous flux vector, and subscripts i and j are used in index notation to capture the dimensionality of the problem (i and j vary from 1 to n, where n is the dimension of the problem).

In order to discretize Eq. 2.1 using the finite volume method, it is necessary to integrate over the volume of the problem.

$$\int_V \left(\frac{\partial Q}{\partial t} + \frac{\partial F_j}{\partial x_j} = \frac{\partial F_j^v}{\partial x_j} \right) dV \tag{2.3}$$

This integral can be simplified using the divergence theorem (often referred to as Gauss's theorem) [39]. This theorem states that the flux through the boundary is equal to the sources or sinks of flux in the closed domain. This yields:

$$\int_V \frac{\partial Q}{\partial t} dV + \int_A F_j n_j dA = \int_A F_j^v n_j dA \tag{2.4}$$

It is important to maintain that integrals with respect to A are taken at the boundaries

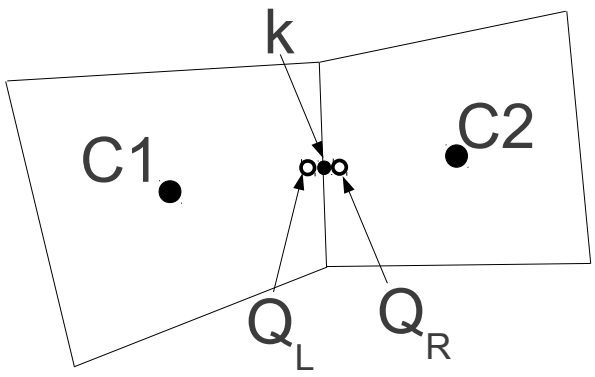


Fig. 2.4: A sample 2D mesh with cells c1 and c2. k represents the face between the cells.

of the control volume. Leibniz's theorem states that for a fixed control volume, the volume is not a function of time and the time integral can be rewritten [40].

$$\frac{d}{dt}\int_V QdV + \int_A F_j n_j dA = \int_A F_j^v n_j dA \tag{2.5}$$

To simplify the derivation, viscous terms are assumed to be negligible and the discussion is limited to the inviscid Euler equations. This yields the following equation:

$$\frac{d}{dt}\int_V QdV + \int_A F_j n_j dA = 0 \tag{2.6}$$

The derivation thus far has made no discrete assumptions and has been entirely continuous. To discretize the above equation, the domain is split into cell-centered control volumes and Q is assumed to be linear in each cell. Q over the entire domain is piecewise linear, which creates a second order accurate scheme.

Discretely defining Q as linear has direct effects on the fluxes on the boundary. The fluxes can be exactly computed on the boundaries using only one quadrature point. A sample of this configuration is found in Fig. 2.4. c1 and c2 are located at the center of the cells and k represents the quadrature point on the boundary. The inviscid fluxes can be written in the form below:

$$\int_A F_j n_j dA = \sum_k \int_{A_k} F_j n_j dA_k \approx \sum_k \hat{\mathcal{F}}_k = R(Q) \tag{2.7}$$

$\hat{\mathcal{F}}_k$ represents the numeric flux at the quadrature points at the cell interface and R represents the "residual." $\hat{\mathcal{F}}_k$ is defined by the following:

$$\hat{\mathcal{F}}_k = \frac{1}{2}\left(\mathcal{F}(Q_L) + \mathcal{F}(Q_R)\right) - D_k(Q_R, Q_L) \tag{2.8}$$

where $D_k(Q_R, Q_L)$ is the artificial dissipation, Q_L and Q_R represent the reconstructed terms presented in the next section, and $\mathcal{F}$ is the area weighted flux given below:

$$\mathcal{F} = F_j n_j A \tag{2.9}$$

The stability of Eq. 4.22 is dependent on the choice of the artificial dissipation term. Without dissipation, the numerical flux becomes immediately unstable. Adding a proper dissipation model makes Eq. 4.22 local extremum diminishing (LED). LED schemes require that local maxima decrease and local minima increase. LED schemes ensure that the local extrema are bounded and prevent them from diverging. The details of how artificial dissipation schemes make the numerical flux LED are left to Chapter 4, where different D_k's are presented and tested with preconditioners.

Finally, the time term is discretized assuming that the conservered variables are linear over the domain. The full discretized model for a cell ($c1$ in this case) is shown below.

$$V\frac{d}{dt}\bar{Q}_{c1} + \sum_k \hat{\mathcal{F}}_k = 0 \tag{2.10}$$

$\bar{Q}$ represents the average Q over the cell domain. The bar will be omitted in further references to this equation for notational simplicity.

2.2.2 Reconstruction for Inviscid Terms

Equation 4.22 is a function of the reconstructed right and left states at the quadrature

points. These reconstructions use the piecewise definition of Q to extrapolate the values from both cell-centers to the quadrature point. Fig. 2.4 shows the locations of Q_R and Q_L. Properly constructing these terms allows the finite-volume method to be second order accurate. The following equation shows how these states are reconstructed:

$$\begin{aligned} Q_L &= Q_{c1} + s\left[(x_k - x_{c1})\left.\frac{dQ}{dx}\right|_{c1} + (y_k - y_{c1})\left.\frac{dQ}{dy}\right|_{c1}\right] \\ Q_R &= Q_{c2} + s\left[(x_k - x_{c2})\left.\frac{dQ}{dx}\right|_{c2} + (y_k - y_{c2})\left.\frac{dQ}{dy}\right|_{c2}\right] \end{aligned} \tag{2.11}$$

The gradients at each cell center are computed using a least squares fit with the surrounding cells. s is a limiter term. When $s = 0$, Eq. 2.11 reduces to a first order reconstruction where the Q's are assumed to be constant over the cell. The second order reconstruction violates LED stability at local extrema. The limiter is designed to give second order reconstruction in smooth regions, but reduce the reconstructions to first order at local maxima and minima. This allows increased accuracy without violating important stability requirements of LED schemes. The mathematical definition of the limiter is given below:

$$s = 1 - \left|\frac{a-b}{|a|-|b|}\right|^q, \quad a = Q_{i+2} - Q_{i+1}, \quad b = Q_i - Q_{i-1} \tag{2.12}$$

where q is a positive integer (chosen to be 3 in this work), $i+1$ and $i+2$ are the cells to the right of the face, and i and $i-1$ are the cells to the left of the face. When a and b have the same sign, the limiter is near unity. At a local extrema, a and b will have opposite signs and the limiter becomes 0.

2.2.3 Boundary Conditions

Boundary condition implementation is a focus of Chapter 3. Only the basic concepts are introduced in this section. Boundaries for strand grids are very similar to the internal discretization, however, unlike internal discretization the flux on the wall is now the actual problem boundary instead of a neighboring cell. Strands use only a cell-centered structure,

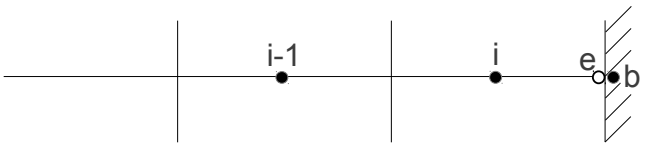


Fig. 2.5: A sample 1D cell centered boundary with extrapolated point and ghost node.

but node-centered boundaries are also introduced.

Cell-Centered

Since known values in cell-centered paradigms are only known at the centroid of the cell, implementing equations at the boundaries is difficult. To apply equations at the boundary, a "ghost-node" is placed at the quadrature point of the boundary cell (b). The values from the interior of the domain are extrapolated in a similar fashion to the interior reconstruction to the boundary (e). Fig. 2.5 shows a simple one dimensional example of this.

$$Q_e = Q_i + (x_e - x_i) \left. \frac{\partial Q}{\partial x} \right|_i \tag{2.13}$$

Like with the interior fluxes, the boundary fluxes can be presented in the form of a boundary residual, $R_b(Q_e, Q_b) = 0$. An example of an outflow that specifies a pressure is:

$$R_b = \begin{Bmatrix} p_b - p_\infty \\ u_b - u_e \\ T_b - T_e \end{Bmatrix} = 0 \tag{2.14}$$

where the subscript ∞ signifies some user-specified value.

Node-Centered

Node-centered schemes have interior nodes that lie on the boundary. The boundary

conditions can be applied directly to these nodes. No extrapolation or ghost nodes are required. Application of the boundary equations at these nodes is much more intuitive. A similar boundary residual is defined $R_b(Q_b) = 0$. Chapter 3 shows how the residual can be defined as a combination of the boundary equations and interior governing equations.

Chapter 3

Verification Techniques

3.1 Introduction

Computational Fluid Dynamics (CFD) is increasingly being applied to applications involving greater complexity than ever before. Common to most of these applications is the requirement for high levels of accuracy at or near domain boundaries. Examples include calculation of aerodynamic lift and drag, computation of boundary layer characteristics, estimation of surface heating, and mass flux computation. For these and other cases, the regions near boundaries are the primary focus of the CFD simulation and require the greatest resolution and numerical accuracy.

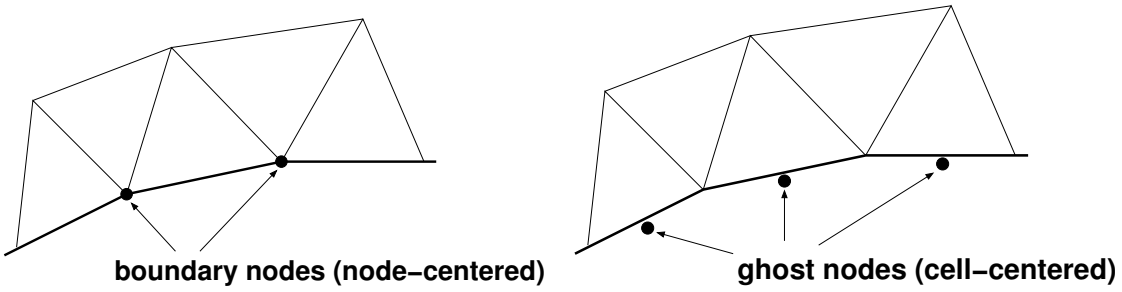
Despite the need for high accuracy near boundaries (or perhaps because of it), numerous boundary procedures have been proposed in the literature. For example, many treatments have been proposed for inviscid walls. The method by Rizzi [23] involves the use of the momentum equation to obtain the pressure. Jameson proposed direct modification of the flux at an inviscid wall to produce zero convective flux contribution [24]. Dadone and Grossman advocate a curvature corrected symmetry condition for an inviscid wall [25]. Balakrishnan and Fernandez recommend a variety of other methods involving additional quantities such as entropy and enthalpy [26]. Numerous other strategies exist for inviscid walls as well as for other boundary conditions, such as inflow, outflow, and no-slip walls. The difficulty is that many of these methods have not been rigorously verified and may or may not be consistent with the interior discretization schemes. Addressing this issue, Choudhary et al. [22] recently proposed a boundary verification procedure using the method of manufactured solutions (MMS). Their approach requires carefully constructed manufactured solutions that already satisfy the boundary conditions, which means that a new manufactured solution

needs to be constructed for each boundary condition and geometry. Nonetheless, their work represents an important step towards comprehensive code verification since most previous verification strategies have neglected boundary effects [17–21].

This work provides an alternate method of boundary verification by focusing on three main goals. First, a general framework for implementing boundary conditions for both node- and cell-centered schemes is provided. The framework applies to arbitrary boundary conditions for any physical system. Second, a rigorous and general approach is provided for the verification of these boundary conditions based on MMS wherein arbitrary manufactured solutions are used in concert with the appropriate formulation of the boundary condition equations. Third, the importance of choosing physically correct boundary conditions for specific physical configurations is demonstrated. It is found that often the choice of well-posed and stable boundary conditions is not unique. However, certain boundary conditions can lead to erroneous results, often in subtle ways.

In order to accomplish these goals, the formulation of a variety of boundary types, including inviscid walls, inflow, and outflow, for both node- and cell-centered finite volume schemes are explored. For each of these boundary types, a variety of conditions that use a combination of Dirichlet, Neumann, and extrapolation conditions are explored. Importantly, the fact that node-centered schemes easily allow the direct use of the governing equations of motion (mass, momentum, and energy) at the boundaries, while cell-centered schemes do not, instead relying on extrapolation or other conditions is highlighted. Other boundary condition types and implementations beyond those discussed in this work are certainly possible. Here, the principal objective is to provide a framework for the implementation and verification of any boundary condition.

The chapter is organized as follows: First, boundary condition formulations are explored for node- and cell-centered finite volume schemes. A common discretization framework is presented to test a variety of governing boundary equations. A demonstration of how to verify these boundary conditions using MMS is provided. Then grid refinement and other qualitative studies are presented. First, a quasi-1D nozzle is tested with both



(a) Boundary degrees of freedom for node-centered methods. (b) Boundary ghost nodes for cell-centered methods.

Fig. 3.1: Location of boundary condition enforcement for node- and cell-centered schemes.

MMS and exact solutions. Next, the use of MMS is extended to two dimensions with wall boundary conditions. Finally, error convergence results for Ringleb flow and a NACA 0012 subsonic airfoil are presented. Then some conclusions derived from these studies are offered.

3.2 Boundary Condition Implementation

In this work a variety of boundary condition implementations and assess the resulting impact on accuracy through rigorous verification studies are tested. In the interior of the domain, the steady Euler equations are solved,

$$\frac{\partial Q}{\partial \tau} + \boldsymbol{\nabla} \cdot \boldsymbol{F} = 0, \tag{3.1}$$

where $Q = (\rho, \rho u, \rho v, \rho e)^T$ is the vector of conserved variables, and $\boldsymbol{F}$ is the inviscid flux vector. Here, ρ is density, u and v are the Cartesian velocity components, and e is the total energy per unit mass. The interest lies in solving the steady equations to which the pseudo-time (τ) derivative for convenience in marching to steady state is added. Both node- and cell-centered finite volume spatial discretizations are tested, which result in a semi-discrete set of non-linear equations of the form

$$\frac{\partial Q}{\partial \tau} + R(Q) = 0, \tag{3.2}$$

where $R(Q)$ is the steady residual at either an interior node or an interior cell location

depending on the discretization procedure. Both cell- and node-centered methods use linear least squares gradient procedures to reconstruct left and right states with CUSP artificial dissipation [36, 41]. Limiters are not employed in this work because all test cases make use of smooth solutions in order to verify order of accuracy. Explicit Runge-Kutta time stepping is used to reach steady state for both node- and cell-centered codes.

In addition to the interior discretization scheme, this work must incorporate boundary conditions to close the system of equations on a finite domain. Here we focus on inviscid wall, inflow, and outflow conditions in order to explore fundamental issues of stability, well-posedness, and numerical accuracy. The methodology developed here is quite general and directly applies to other boundary condition types as well. For node-centered discretizations, boundary conditions are enforced directly at the boundary nodes coincident with the physical boundary, shown in Figure 3.1(a). For cell-centered discretizations, additional unknowns in the form of ghost nodes located at the flux quadrature points of the boundary faces are introduced, shown in Figure 3.1(b). The ghost nodes are then used in an upwind flux formula to determine the numerical flux through the boundary face. In this manner, the cell-centered boundary formulation remains water-tight. The node-centered configuration, however, is not strictly water-tight since the fluxes surrounding the boundary nodes do not always cancel with nearby interior nodes.

For both node- and cell-centered formulations, a "boundary residual," $R_b(Q)$, is defined which is driven to zero at steady state along with the interior residuals, $R(Q)$:

$$R_b(Q) = 0. \tag{3.3}$$

In a node-centered discretization, R_b replaces R at the boundary nodes. In cell-centered discretizations, R_b provides the governing equations for the ghost nodes. In all, this work tests fifteen different boundary implementations, which are listed with a common notational convention in Table 3.1 for clarity. The methods involve a certain number of Dirichlet-specified quantities, with the state specification completed by additional methods, such as Neumann conditions, extrapolation, or in some cases, the equations of motion themselves.

This work will refer to this table as various forms for R_b are developed in the following sections.

Table 3.1: Notation for boundary condition methods tested.

Short Name	Boundary Type		Dirichlet	Other conditions
n-INV1	Node-centered	Inviscid wall	u_n	Lagrange multipliers
n-INV2	Node-centered	Inviscid wall	u_n	u_t Neumann, mass and energy eqs.
n-INV3	Node-centered	Inviscid wall	u_n	$\rho, u_t, \rho e$ Neumann
n-INV4	Node-centered	Inviscid wall	–	zero convective flux
n-INF1	Node-centered	Inflow	s, h_0, u_t	Lagrange multipliers
n-INF2	Node-centered	Inflow	s, h_0, u_t	outgoing characteristic eq.
n-OUT1	Node-centered	Outflow	p	Lagrange multipliers
n-OUT2	Node-centered	Outflow	p	outgoing characteristic eqs.
c-INV1	Cell-centered	Inviscid wall	u_n	$\rho, u_t, \rho e$ extrapolated
c-INV2	Cell-centered	Inviscid wall	–	Pressure extrapolated
c-INV3	Cell-centered	Inviscid wall	u_n	$\rho, u_t, \rho e$ Neumann
c-INF1	Cell-centered	Inflow	s, h_0, u_t	u_n extrapolated
c-INF2	Cell-centered	Inflow	R^-, u_t, s	R^+ extrapolated
c-OUT1	Cell-centered	Outflow	p	ρ, u, v extrapolated
c-OUT2	Cell-centered	Outflow	R^-	R^+, u_t, s extrapolated

3.2.1 Node-Centered Boundaries

All boundary conditions involve the specification of a certain number of Dirichlet (or Neumann) conditions augmented by additional information derived from the interior field. With node-centered schemes it is straightforward to select some combination of the governing equations of motion (mass, momentum, and energy) to enforce at boundary nodes. This is because the boundary nodes lie within control volumes for which flux balances can easily implemented. In contrast, cell-centered boundary conditions require the use of ghost nodes for which there is no natural control volume or flux balance. Thus, it becomes difficult to apply the equations of motion directly at the ghost nodes. Instead, other methods are chosen to define the ghost node state such as solution extrapolation.

Enforcement of boundary conditions in a node-centered scheme involves the specification of the boundary residual, R_b, directly at the boundary nodes shown in Figure 3.1(a). In this section, two procedures to obtain the boundary residual are outlined. The first

involves the use of Lagrange multipliers as discussed by Allmaras [28]. The second involves the multiplication of the governing equations of mass, momentum, and energy by a selection matrix to complete the boundary conditions. In addition, discussion a third method involving a commonly used weak boundary condition for an inviscid wall is presented.

Lagrange Multipliers

A method of boundary condition enforcement that has enjoyed widespread use in the finite element community for several decades is the Lagrange multiplier method introduced by Babuska [42]. The extension of this method to the Navier-Stokes equations was discussed by Allmaras [28]. The method involves a modification of the variational statement to include extra conditions along Dirichlet boundaries, leading to an extended system of the form

$$\begin{Bmatrix} B(Q) - b \\ \frac{\partial Q}{\partial \tau} + R(Q) + \left(\frac{\partial B}{\partial \tilde{Q}}\right)^T \lambda \end{Bmatrix} = 0. \tag{3.4}$$

Here, $B(Q) - b$ represents a vector of m Dirichlet conditions that must be satisfied at boundary nodes. For example, at a fixed inviscid wall, $m = 1$, $B(Q) = \boldsymbol{u} \cdot \boldsymbol{n}$, and $b = 0$. The additional conditions are incorporated via a vector of m Lagrange multipliers, λ. The choice of variables for $\tilde{Q}$ in Equation 3.4 is non-unique and can lead to different sets of boundary equations. While Allmaras chooses entropy variables, $\tilde{Q} = \frac{\rho}{p}(\frac{p}{\rho}\frac{\gamma+1-s}{\gamma-1} - e, u, v, -1)^T$, other variables, such as the primitive variables, $\tilde{Q} = (P, u, v, T)^T$, are possible, leading to different boundary equations. This work employs entropy variables for $\tilde{Q}$, following Allmaras.

While the extended system in Equation 3.4 can be solved directly to include the Lagrange multipliers, these may be eliminated from the problem altogether, resulting in a boundary residual form

$$R_b(Q) = \begin{Bmatrix} B(Q) - b \\ N\left(\frac{\partial Q}{\partial \tau} + R(Q)\right) \end{Bmatrix}. \tag{3.5}$$

Here, N is a $(n-m) \times n$ matrix containing a basis for the nullspace of the boundary condition matrix, such that

$$N\left(\frac{\partial B}{\partial \tilde{Q}}\right)^T = 0. \tag{3.6}$$

In this work, both forms of the Lagrange multiplier method in Equations 3.4 and 3.5 are tested and verified to give identical results in the steady solution and negligible differences in convergence behavior.

A novel aspect of this work is a new method for the verification of boundary conditions through grid refinement studies. This is done for exact solutions which are available for certain simple configurations. However, a more general method that is applicable to a wide variety of boundary conditions, geometries, and governing equations that do not possess exact solutions is desired. To accomplish this, an extension of previous work using the method of manufactured solutions (MMS) [43] is performed to include boundary conditions. This enables the use of a single arbitrary manufactured solution to simultaneously verify the accuracy of both the interior scheme and the boundary conditions. Such a procedure can be accomplished by adding an MMS source term to both the Euler and boundary equations:

$$\frac{\partial Q}{\partial \tau} + \boldsymbol{\nabla} \cdot \boldsymbol{F} = S(\boldsymbol{x})$$

$$B(Q) - b = S_b(\boldsymbol{x}) \tag{3.7}$$

For the Lagrange multiplier method this results in a modified boundary residual of the form

$$R_b(Q) = \begin{Bmatrix} B(Q) - b - S_b(\boldsymbol{x}) \\ N\left(\frac{\partial Q}{\partial \tau} + R(Q) - S(\boldsymbol{x})\right) \end{Bmatrix}. \tag{3.8}$$

Note that the proposed method of boundary verification does not require that the manufactured solution satisfy the boundary conditions as required by other approaches [22]. This greatly simplifies the verification procedure, and allows a single manufactured solution to be used to verify a variety of boundary condition types.

Three boundary conditions are tested that make use of the Lagrange multiplier methodology. These are denoted n-INV1, nINF1, and n-OUT1 in Table 3.1. Method n-INV1 for a node-centered inviscid wall sets the normal velocity to zero and retains a combination of the mass, momentum, and energy equations:

$$B(Q) = u_n, \quad b = 0, \quad N = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & n_y & -n_x & 0 \\ 0 & -u & -v & 1 \end{pmatrix}. \tag{3.9}$$

Here, $\boldsymbol{n} = (n_x, n_y)^T$ is the outward pointing unit normal vector at the surface node, and $u_n = n_x u + n_y v$ is the velocity in the normal direction.

The subsonic inflow condition n-INF1 specifies entropy, s, total enthalpy, h_0, and tangential velocity, $u_t = -n_y u + n_x v$:

$$B(Q) = \begin{pmatrix} h_0 \\ s \\ u_t \end{pmatrix}, \quad b = \begin{pmatrix} h_{0,spec} \\ s_{spec} \\ u_{t,spec} \end{pmatrix}, \quad N = \begin{pmatrix} u_n h_0 & u_t v - n_x(h_0 + \frac{1}{2}q^2) & -u_t u - n_y(h_0 + \frac{1}{2}q^2) & u_n \end{pmatrix} \tag{3.10}$$

Here, the subscript *spec* denotes a specified value and q is the velocity magnitude.

The final method using Lagrange multipliers is n-OUT1 for subsonic outflow. This method fixes the static pressure,

$$B(Q) = p, \quad b = p_{spec}, \quad N = \begin{pmatrix} -u & 1 & 0 & 0 \\ -v & 0 & 1 & 0 \\ -h_0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.11}$$

Again, this work emphasizes that the method of Lagrange multipliers provides a way to determine a complete set of boundary equations. Once $B(Q)$ is determined and a set of variables $\tilde{Q}$, the remaining equations are fixed through the nullspace, N. However, the actual form of N is dependent on the choice of $\tilde{Q}$, which is arbitrary. Further work is needed to determine which set of $\tilde{Q}$ and the resulting N is best suited for various boundary

conditions.

Selection Matrix Method

An alternate method of obtaining a boundary residual is through a selection matrix that picks desired combinations of the equations of motion. Using a selection matrix, L, the boundary residual is defined as

$$R_b(Q) = \Omega(Q) + L\left(\frac{\partial Q}{\partial \tau} + R(Q)\right). \tag{3.12}$$

Here, L selects certain combinations of the equations of motion which augment other boundary conditions, $\Omega(Q)$. These conditions may be Dirichlet (D), Neumann (N), or extrapolated (E):

$$\Omega(Q) = \Omega_D(Q) + \Omega_N(Q) + \Omega_E(Q). \tag{3.13}$$

Whereas the Lagrange multiplier approach only accommodates Dirichlet conditions, this framework allows for general specification of virtually any type of boundary condition. The Lagrange multiplier method may therefore be thought of as a subset of this more general approach, where $\Omega_D(Q) = B(Q) - b$, and $L = N$ augmented with rows of zeros.

This more general form is also easily modified for verification via MMS by introducing source terms for $\Omega(Q)$ and the equations of motion themselves:

$$R_b(Q) = \Omega(Q) - S_b(\boldsymbol{x}) + L\left(\frac{\partial Q}{\partial \tau} + R(Q) - S(\boldsymbol{x})\right). \tag{3.14}$$

Here, $S_b(\boldsymbol{x})$ acts on the $\Omega(Q)$ terms, and $S(\boldsymbol{x})$ acts on the governing equations of motion.

Four node-centered are tested boundary conditions that make use of the selection matrix method. These are denoted n-INV2, n-INV3, n-INF2, and n-OUT2 in Table 3.1. For n-INV2, a Neumann condition is set only for the tangential velocity terms, while the mass

and energy equations are used directly:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ 0 \\ u_n \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = \begin{pmatrix} 0 \\ \partial u_t/\partial n \\ 0 \\ 0 \end{pmatrix}, \quad \Omega_E(Q) = 0, \quad L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.15}$$

In this manner, velocity is treated with a symmetry condition while mass and energy are conserved. These types of symmetry conditions enforced with Neumann conditions have been advocated by many other researchers for slip walls in inviscid flows [44–46].

Method n-INV3 uses Neumann conditions for all equations, completely omitting any contribution from the Euler equations:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ 0 \\ u_n \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = \begin{pmatrix} \partial\rho/\partial n \\ \partial u_t/\partial n \\ 0 \\ \partial(\rho e)/\partial n \end{pmatrix}, \quad \Omega_E(Q) = 0, \quad L = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.16}$$

This condition is strictly a symmetry condition, but is often used at inviscid walls in practice.

For subsonic inflow method n-INF2, the selection matrix extracts the outgoing characteristic equation at the inflow boundary, while three other conditions (total enthalpy, entropy, and tangential velocity) are prescribed. Here, it is desirable to select the equation associated with the outgoing $u_n - c$ wave. To accomplish this, the Euler equations are first rotated to a coordinate system with components normal (n) and tangential (t) to the boundary,

$$R\left(\frac{\partial Q}{\partial \tau} + \frac{\partial F_x}{\partial x} + \frac{\partial F_y}{\partial y}\right) = \frac{\partial Q'}{\partial \tau} + \frac{\partial F_n}{\partial n} + \frac{\partial F_t}{\partial t} = 0. \tag{3.17}$$

Here, the definitions of the rotation matrix, rotated solution, and flux vectors are

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & n_x & n_y & 0 \\ 0 & -n_y & n_x & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad Q' = RQ = \begin{pmatrix} \rho \\ \rho u_n \\ \rho u_t \\ \rho e \end{pmatrix}, \quad F_n = \begin{pmatrix} \rho u_n \\ \rho u_n^2 + p \\ \rho u_n u_t \\ \rho u_n h_0 \end{pmatrix}, \quad F_t = \begin{pmatrix} \rho u_t \\ \rho u_t u_n \\ \rho u_t^2 + p \\ \rho u_t h_0 \end{pmatrix}. \tag{3.18}$$

Equation 3.17 is then multiplied by the modal matrix, M_n^{-1}, containing the right eigenvectors of the normal flux Jacobian, $A_n = \partial F_n / \partial Q'$. Using the fact that $M_n^{-1} A_n M_n = \Lambda_n = \text{diag}(u_n, u_n, u_n + c, u_n - c)$, leads to

$$\frac{\partial \hat{Q}'}{\partial \tau} + \Lambda_n \frac{\partial \hat{Q}'}{\partial n} + M_n^{-1} A_t M_n \frac{\partial \hat{Q}'}{\partial t} = 0, \tag{3.19}$$

where $\hat{Q}' = M_n^{-1} RQ$. It is clear from this form that the fourth equation represents the characteristic wave equation leaving the boundary. This can be accomplished by defining the selection matrix as

$$L = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} M_n^{-1} R, \tag{3.20}$$

where

$$M_n^{-1} R = \begin{pmatrix} 1 - \frac{(\gamma-1)q^2}{2c^2} & (\gamma-1)\frac{u}{c^2} & (\gamma-1)\frac{v}{c^2} & -\frac{\gamma-1}{c^2} \\ u_t & n_y & -n_x & 0 \\ \frac{1}{2c^2}(\frac{\gamma-1}{2}q^2 - cu_n) & \frac{1}{2c^2}(n_x c - (\gamma-1)u) & \frac{1}{2c^2}(n_y c - (\gamma-1)v) & \frac{\gamma-1}{2c^2} \\ \frac{1}{2c^2}(\frac{\gamma-1}{2}q^2 + cu_n) & -\frac{1}{2c^2}(n_x c + (\gamma-1)u) & -\frac{1}{2c^2}(n_y c + (\gamma-1)v) & \frac{\gamma-1}{2c^2} \end{pmatrix} \tag{3.21}$$

The n-INF2 condition is completed by specifying total enthalpy, entropy, and tangential velocity:

$$\Omega_D(Q) = \begin{pmatrix} h_0 - h_{0,spec} \\ s - s_{spec} \\ u_t - u_{t,spec} \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = 0. \tag{3.22}$$

The outflow case n-OUT2 is based on a similar approach as n-INF2, but selects the remaining three characteristics at the outflow, while fixing the static pressure:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ p - p_{spec} \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = 0, \quad L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} M_n^{-1} R \tag{3.23}$$

Weak Form Flux Implementation

A final method of boundary implementation suitable for node-centered schemes is through a modification of the flux. Traditionally, this has been used at an inviscid wall by setting the convective portion to zero and retaining only the pressure terms [24]:

$$F_n = \begin{pmatrix} 0 \\ pn_x \\ pn_y \\ 0 \end{pmatrix}. \tag{3.24}$$

Here, F_n is the directional flux at a boundary face with normal $\boldsymbol{n} = (n_x, n_y)^T$. This method is denoted n-INV4 in Table 3.1.

It is interesting to note that this method does not explicitly involve a boundary residual, and thus cannot be verified using MMS and source terms as the other methods allow. Nonetheless, the results show that this method performs very well as shown through grid refinement studies using exact solutions. It is also interesting to note that this method does

provide a water-tight formulation for node-centered schemes using inviscid walls.

3.2.2 Cell-Centered Boundaries

Unlike node-centered schemes, cell-centered schemes do not contain unknowns that lie directly on the boundary. Instead, ghost nodes are introduced as shown in Figure 3.1(b). It is at these ghost node locations that the boundary residual must be satisfied at steady-state. The ghost nodes do not lie within control volumes and do not contain flux balances from the governing equations. For this reason, it is difficult to apply the discretized equations of motion (e.g. mass, momentum, or energy) at boundary locations. For this reason, the boundary condition choices are limited to a combination of Dirichlet, Neumann, or extrapolation conditions:

$$R_b(Q) = \Omega(Q) = \Omega_D(Q) + \Omega_N(Q) + \Omega_E(Q). \tag{3.25}$$

These conditions may also be verified using MMS by adding a source term to the boundary residual:

$$R_b(Q) = \Omega(Q) - S_b(\boldsymbol{x}). \tag{3.26}$$

An important point is that for extrapolation conditions, the source term should be set to zero. This is because in the limit of grid refinement, the extrapolated value will equal the manufactured solution exactly.

Six cell-centered are tested boundary conditions that make use of the ghost node method. These are denoted c-INV1, c-INV3, c-INF1, c-INF2, c-OUT1, and c-OUT2 in Table 3.1. Method c-INV1 for an inviscid wall sets the normal velocity to zero, while extrapolating density, tangential velocity, and total energy:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ 0 \\ u_n \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = \begin{pmatrix} \rho - \rho_E \\ u_t - u_{tE} \\ 0 \\ \rho e - (\rho e)_E \end{pmatrix}. \tag{3.27}$$

Here, the subscript E refers to the state (linearly) extrapolated from the interior of the domain.

Method c-INV3 for an inviscid wall sets the normal velocity to zero, but uses Neumann conditions for density, tangential velocity, and total energy:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ 0 \\ u_n \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = \begin{pmatrix} \partial\rho/\partial n \\ \partial u_t/\partial n \\ 0 \\ \partial(\rho e)/\partial n \end{pmatrix}, \quad \Omega_E(Q) = 0. \tag{3.28}$$

The Neumann conditions may be more appropriate for a symmetry condition, but this is often used for an inviscid wall in practice. The suitability of symmetry conditions for inviscid walls is analyzed in detail in the results section.

Method c-INF1 for subsonic inflow fixes the thermodynamic stagnation state and incoming tangential velocity, while extrapolating the velocity normal to the boundary:

$$\Omega_D(Q) = \begin{pmatrix} h_0 - h_{0,spec} \\ s - s_{spec} \\ u_t - u_{t,spec} \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ u_n - u_{nE} \end{pmatrix}. \tag{3.29}$$

Method c-INF2 for subsonic inflow uses Riemann invariants normal to the boundary, entropy, and tangential velocity. The Riemann invariants have the usual definition for an ideal gas,

$$R^+ = u_n + \frac{2c}{\gamma - 1}, \quad R^- = u_n - \frac{2c}{\gamma - 1}, \tag{3.30}$$

where c is the speed of sound. The boundary condition then becomes,

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ R^- - R^-_{spec} \\ u_t - u_{t,spec} \\ s - s_{spec} \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = \begin{pmatrix} R^+ - R^+_E \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.31}$$

The subsonic outflow condition c-OUT1 specifies the static pressure, and extrapolates density and velocity:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ p - p_{spec} \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = \begin{pmatrix} \rho - \rho_E \\ u - u_E \\ v - v_E \\ 0 \end{pmatrix}. \tag{3.32}$$

The subsonic outflow condition c-OUT2 is similar to c-INF2, but extrapolates tangential velocity and entropy to be consistent with the number of characteristics leaving the domain:

$$\Omega_D(Q) = \begin{pmatrix} 0 \\ R^- - R^-_{spec} \\ 0 \\ 0 \end{pmatrix}, \quad \Omega_N(Q) = 0, \quad \Omega_E(Q) = \begin{pmatrix} R^+ - R^+_E \\ 0 \\ u_t - u_{tE} \\ s - s_E \end{pmatrix}. \tag{3.33}$$

The final condition listed in Table 3.1 is c-INV2. This condition is similar to n-INV4, which weakly modifies the flux at the boundary to only contain the pressure terms, as in Equation 3.24. Thus only pressure needs to be extrapolated. As in n-INV4, it is not clear what the boundary residual is for this method. Thus, it is difficult to verify the method directly using MMS and source terms as is done in the case of the other methods.

3.3 Results

In this section, results of the various boundary condition implementations in one and two dimensions for node- and cell-centered schemes are presented. First quasi-1D nozzle flow are examined. This section then extends these methods to two dimensions using a manufactured Euler solution in a square domain. Finally, the procedures for Ringleb flow and subsonic flow over a NACA 0012 airfoil are tested. For all cases, grid refinement studies are performed with the boundary condition options listed in Table 3.1.

3.3.1 Quasi-1D Euler Equations

The quasi-1D Euler equations are a 1D formulation with added cross sectional area, $a(x)$, useful for nozzle flows. The quasi-1D Euler equations are defined as

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} = S_p \tag{3.34}$$

$$Q = a \begin{pmatrix} \rho \\ \rho u \\ \rho e \end{pmatrix}, F = a \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u h_0 \end{pmatrix}, S_p = \begin{pmatrix} 0 \\ \frac{\partial a}{\partial x} p \\ 0 \end{pmatrix} \tag{3.35}$$

In addition this work makes use of the ideal gas equation of state and assume constant specific heat. For the area, the following is used,

$$a(x) = a_0 \left(1 + \frac{1}{2} cos(2\pi x)\right), \tag{3.36}$$

where $a_0 = 2/3$ to make the inlet and outlet area unity, with domain extents $x \in [0, 1]$. A view of the nozzle is shown in Figure 3.2.

The governing equations are solved using a node-centered approach to test inflow and outflow conditions n-INF1, n-INF2, n-OUT1, and n-OUT2. In order to verify the accuracy of the boundary methods, the exact solutio, which for fully subsonic flow, may be found simply from preserving constant entropy, enthalpy, and mass flux is considered. For all

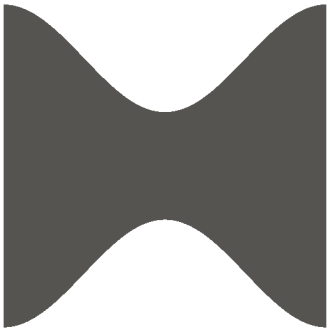


Fig. 3.2: Converging-diverging nozzle used for the Quasi-1D Euler equations.

tests, $M = 0.15$ was selected at the inflow to ensure subsonic flow at the throat, which is suitable for a grid refinement accuracy study.

While the quasi-1D Euler equations possess exact solutions, formulation of a manufactured solution is done to demonstrate the new MMS verification procedure for the boundary conditions. The manufactured solution is chosen as

$$\begin{aligned} \rho^{MMS}(x) &= c_1 + c_{x1} sin(a_{x1} x) \\ u^{MMS}(x) &= c_2 + c_{x2} cos(a_{x2} x) \\ p^{MMS}(x) &= c_3 + c_{x3} sin(a_{x3} x) \end{aligned} \tag{3.37}$$

These constants are selected such that the manufactured solution remains physically meaningful (e.g. positive density). The period of the sinusoidal oscillation is set to approximately 20 times the length scale of the problem to provide a very smooth solution in the asymptotic range of convergence. Values for the constants are shown in table 3.2.

Next, it is necessary to determine the accompanying source terms, $S(x)$ and $S_b(x)$, in Equation 3.14. The interior source terms are defined in the usual manner by setting the source terms equal to the residual quantity remaining after the manufactured solution is substituted into the governing equations. The boundary source terms are computed in a

Table 3.2: MMS constants used for the quasi-1D Euler solution verification.

Constant	Value
c_1	1.0
c_{x1}	0.15
a_{x1}	0.075π
c_2	70.0
c_{x2}	7.0
a_{x2}	0.15π
c_3	1.0×10^{-5}
c_{x3}	2.0×10^{-4}
a_{x3}	0.1π

similar fashion, but considering only the boundary equations, such that

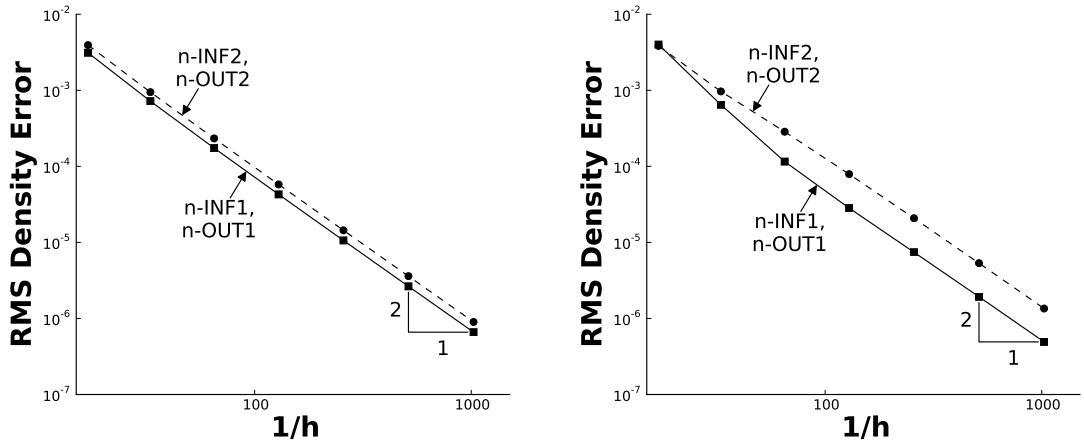
$$S_b = \Omega(Q^{MMS}), \tag{3.38}$$

where Q^{MMS} is the chosen manufactured solution. For example, the n-OUT2 condition, which specifies the exit pressure requires a boundary source term computed with

$$S_b = \Omega(Q^{MMS}) = \begin{pmatrix} 0 \\ 0 \\ p^{MMS} - p_{spec} \end{pmatrix}. \tag{3.39}$$

With these definitions of S and S_b, it is possible to verify the interior scheme and boundary conditions simultaneously using MMS.

The results of the grid refinement study using the exact and manufactured solutions for the quasi-1D nozzle are shown in Figure 3.3. All methods produce second order accurate results. These results highlight two important points. First, the choice of boundary conditions that lead to consistent formulations is not unique. In this case the Lagrange multiplier method and the characteristic matrix selection method produce different equations. However, both lead to second-order accuracy. While the choice of equations is not unique, it is by no means arbitrary. Later this work shows examples of boundary conditions that are wrong and therefore degrade the accuracy.



(a) Error obtained from exact solution

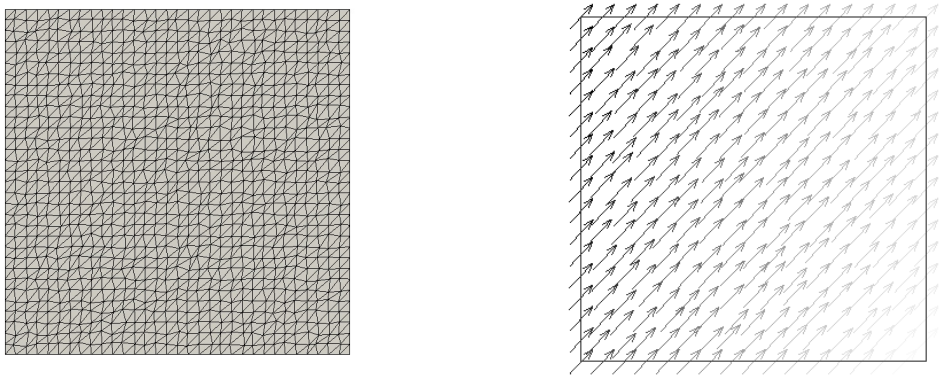
(b) Error obtained from manufactured solution

Fig. 3.3: Grid refinement study for quasi-1D Euler equations using methods n-INF1, n-INF2, n-OUT1, and n-OUT2 with exact and manufactured solutions.

A second important point is that the MMS procedure provides reliable information regarding the accuracy of the boundary formulation. In this case, an exact solution is available. However, in the vast majority of cases that use more complex equations and boundary conditions, exact solutions are not available. In these cases, this section has shown that the MMS procedure can provide a straightforward way to quantify the accuracy of boundary conditions. By determining the proper forms of $S(x)$ and $S_b(x)$, it is possible to use an arbitrary manufactured solution to verify boundary conditions along with the interior scheme.

3.3.2 Manufactured Solution in Square Domain

Next, the one-dimensional formulation above is extended to two-dimensional flows and further explore the verification of boundary conditions via manufactured solutions on a simple square domain. Consider the unit square domain shown in Figure 3.4. The domain is discretized with perturbed triangular cells, shown in Figure 3.4(a), to avoid any fortunate cancellation of solution error due to grid regularity. Velocity vectors for the chosen manufactured solution is shown in Figure 3.4(b). The manufactured solution follows a sinusoidal



(a) 32×32 perturbed triangular mesh (b) Manufactured solution velocity vectors.

Fig. 3.4: Mesh and manufactured solution used for boundary condition verification.

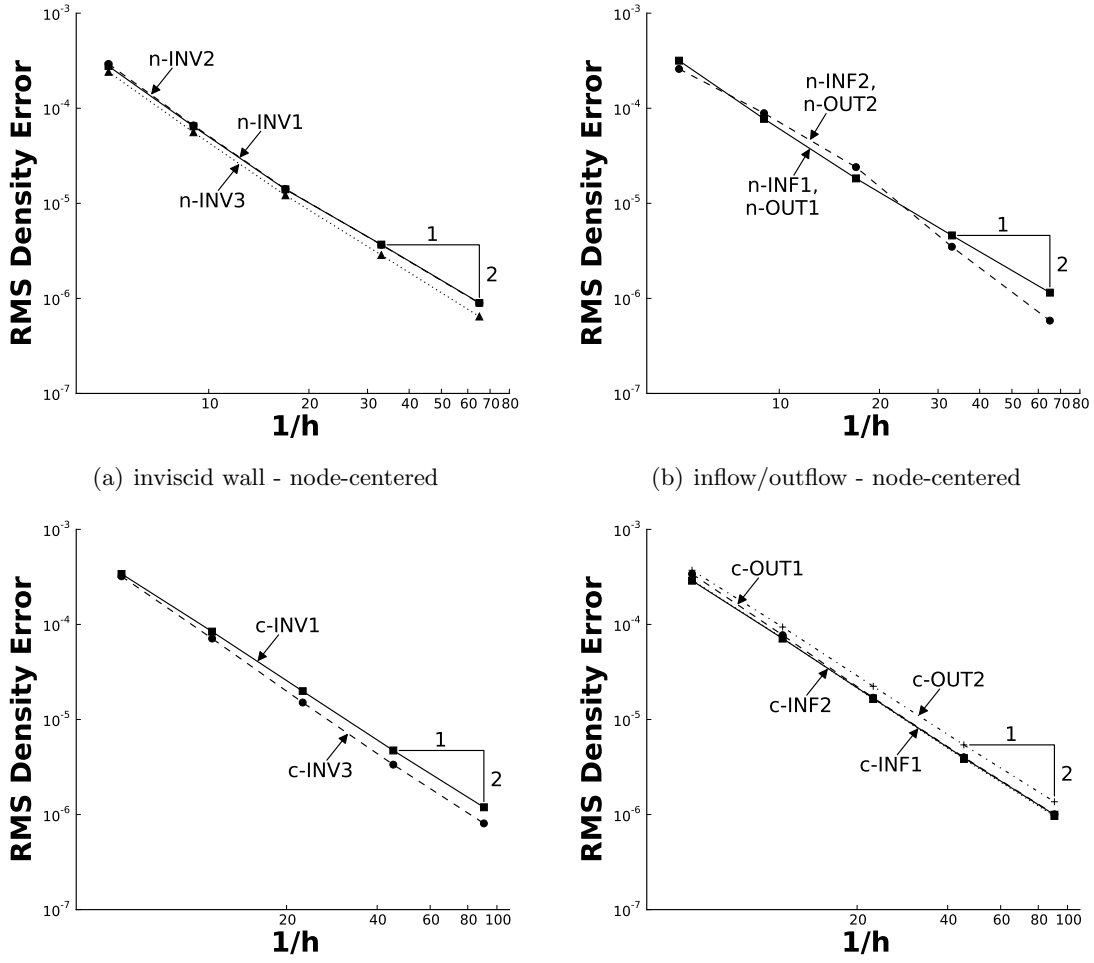
form similar to the solution used for the quasi-1D nozzle flow. An example for density is

$$\rho(x, y) = c_1 + c_{x1} sin(a_{x1} x) + c_{y1} sin(a_{y1} y) + c_{xy1} cos(a_{xy1} xy). \tag{3.40}$$

Here the constants are chosen to be physically meaningful (e.g. positive density). The other flow variables (u, v, p) use similar forms. For the computation of source terms in multiple dimensions it is helpful to use a symbolic math tool such as the one contained in Matlab.

The boundary conditions in Table 3.1 are systematically tested the accuracy of in isolation to ensure that the results are independent of one another. To isolate a given boundary type, pure Dirichlet conditions are enforced on all the other boundaries of the domain except the one in question. The inviscid wall cases (INV) are applied on the top of the domain, inflow (INF) on the left side, and outflow (OUT) on the right side. For some of the methods (n-INV4 and c-INV2), it is not possible to apply the MMS procedure since it is unclear what the exact governing equations are for these weak forms.

Figure 3.5 shows the results of the tests performed for all the available MMS boundary types for node- and cell-centered methods. Figures 3.5(a) and 3.5(c) show the results for



(a) inviscid wall - node-centered

(b) inflow/outflow - node-centered

(c) inviscid wall - cell-centered

(d) inflow/outflow - cell-centered

Fig. 3.5: Grid refinement study for a variety of boundary formulations using MMS.

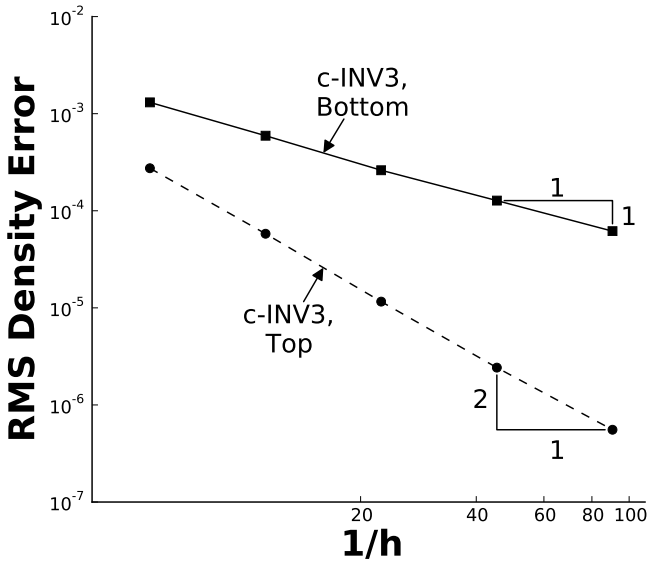


Fig. 3.6: Example of the violation of characteristic directions for MMS solutions.

the isolated inviscid wall for both discretization schemes. The inflow and outflow results are combined into Figures 3.5(b) and 3.5(d) for node- and cell-centered methods respectively. The grid refinement study shows that Dirichlet, Neumann, extrapolation, and the equations of motion themselves are all reliable second-order accurate boundary conditions. The MMS procedure proves successful at verifying the proper implementation of the interior and boundary governing equations. A subtle but important point is that this procedure assumes the boundary normal vectors are exact at the nodes for the node-centered scheme. This assumption will be investigated further.

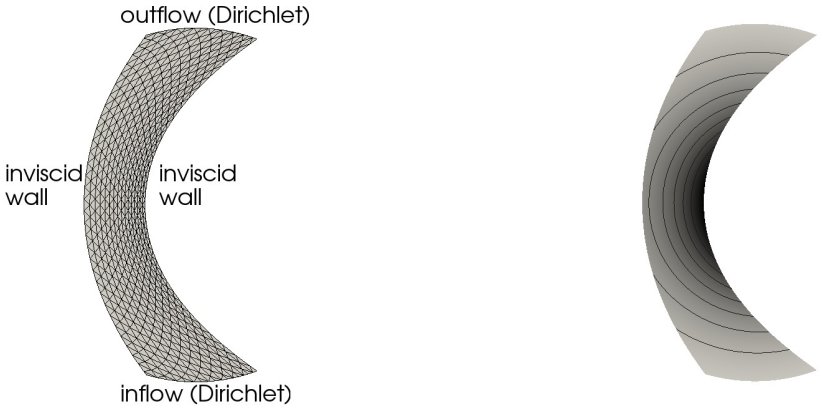
Two important considerations for using MMS verification for boundary conditions are well-posedness and stability. Specifically, it is critical to ensure compatibility of the characteristic directions of the manufactured solution with the boundary conditions being tested. This is illustrated by considering inviscid wall conditions. Figure 3.4(b) illustrates that the flow constants have been chosen such that flow is directed diagonally towards the upper-right of the domain. Because an inviscid wall specifies one piece of information ($u_n = 0$), this limits the location at which MMS verification can be implemented for an inviscid wall to the top of the domain. (Recall that all other boundaries are set to Dirichlet to isolate the

effect of the top wall.) This was the procedure for all the results obtained above. Placing the inviscid wall condition at the bottom of the square domain violates the wave propagation characteristics of the manufactured solution because it would require three specified pieces of information, not one. The effect of this ill-posed condition is shown in Figure 3.6, which shows the comparison of density error for an inviscid wall (c-INV3) at the top and bottom of the domain. It is noteworthy to mention that other inviscid wall boundary formulations (n-INV1-3) would not converge when enforced at the bottom of the domain due to stability issues. It is therefore critical that the manufactured solution and boundaries are chosen such that the characteristic directions are not violated.

3.3.3 Ringleb Flow

In the previous section it was shown that all boundary condition methods tested are second order accurate using the method of manufactured solutions. Methods n-INV4 and c-INV2 could not be tested because the form of the underlying boundary residual, R_b, is unknown. Here, examination of the same methods , including n-INV4 and c-INV2, using Ringleb flow [47], is done, which is an exact solution of the Euler equations. The results of this test case highlight the importance of selecting physically correct conditions for a given problem. Just because a particular boundary condition is implemented in a mathematically consistent way (passes an MMS verification test for example), does not mean that it is an appropriate physical boundary condition. Additionally, this work highlights the difficulties of using derived quantities, such as entropy, as measures of the solution error. Finally, this case also reveals the importance of computing boundary normals correctly for certain node-centered methods.

Ringleb flow describes inviscid compressible flow turning 180^o and involves subsonic, transonic, and supersonic regimes. Here, the focus is on a subsonic portion of the flow. The exact solution is obtained via a hodograph method in the form of (x, y) coordinates parameterized by velocity magnitude, q, and streamline constant, k, which is the inverse of the stream function, $\psi = \frac{1}{q}\sin\theta$. Here, θ is the flow angle. The solution may be expressed



(a) 16 x 32 triangular mesh. (b) Field plot of Ringleb flow density field.

Fig. 3.7: Configuration for Ringleb flow test case.

as

$$x(q,k) = \frac{1}{2\rho}\left(\frac{1}{q^2} - \frac{2}{k^2}\right) + \frac{J}{2} \tag{3.41}$$

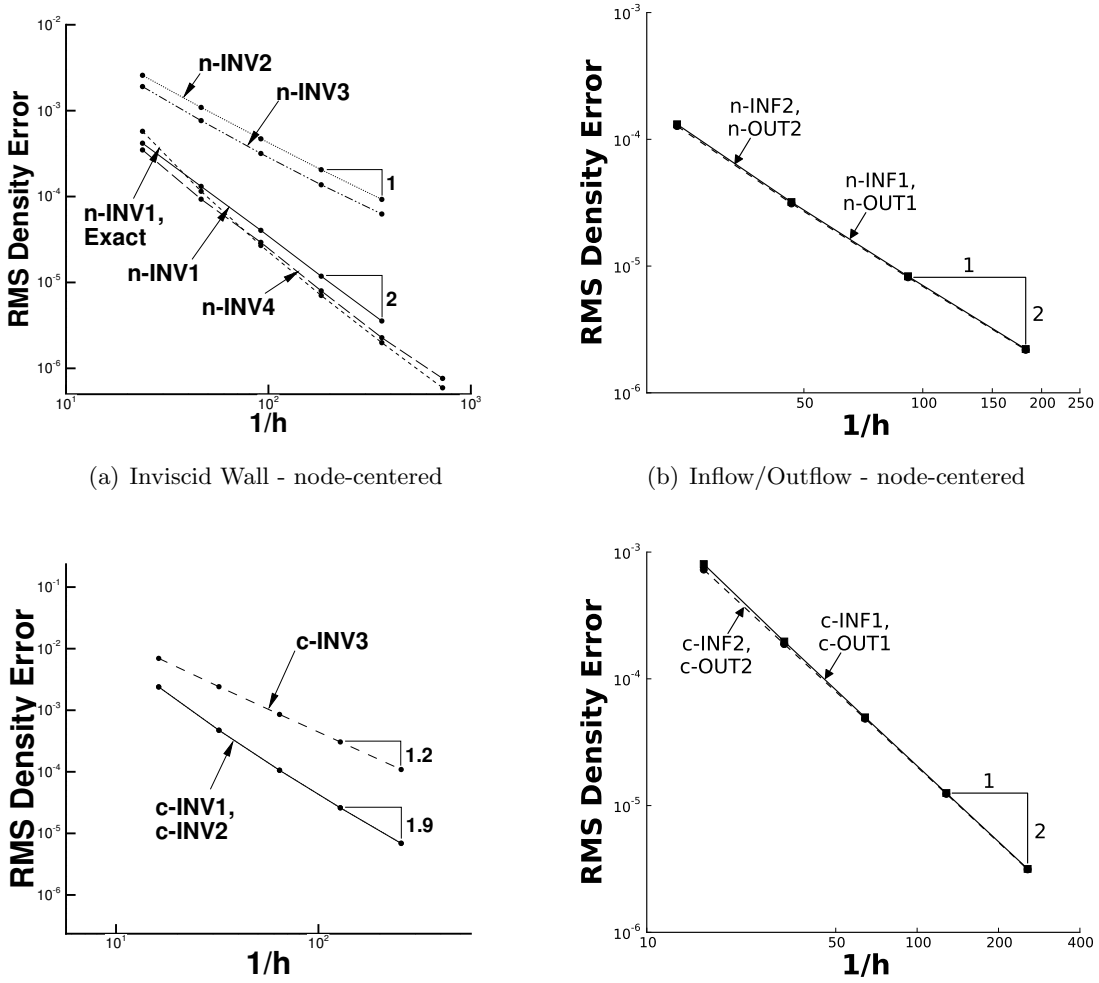
$$y(q,k) = \pm\frac{1}{kq\rho}\sqrt{1-\frac{q^2}{k^2}}, \tag{3.42}$$

where

$$\begin{aligned} J &= \frac{1}{c} + \frac{1}{3c^3} + \frac{1}{5c^5} - \frac{1}{2}\log\left(\frac{1+c}{1-c}\right) \\ c &= \sqrt{1-\frac{\gamma-1}{2}q^2} \\ \rho &= c^{\frac{2}{\gamma-1}}. \end{aligned}$$

Given a location (x, y), it is possible to solve for the corresponding (q, k) pair with a few Newton iterations. The flow solution is dimensionalized with desired stagnation quantities.

A series of increasingly refined meshes is used to verify the accuracy of boundary conditions. The mesh and exact solution for Ringleb flow are shown in Figure 3.7. The domain consists of two streamlines, which are treated as inviscid walls, as well as an inflow and outflow at the bottom and top portions of the domain, as shown in Figure 3.7(a). Each



(a) Inviscid Wall - node-centered

(b) Inflow/Outflow - node-centered

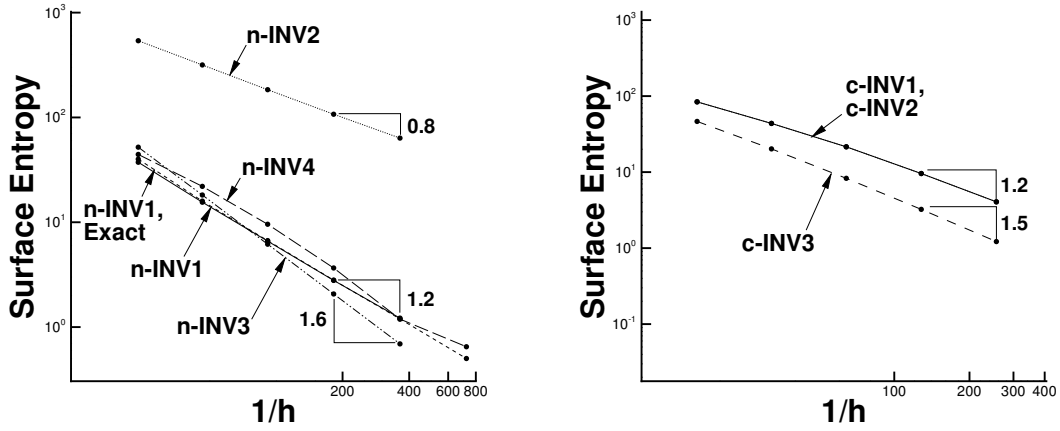
(c) Inviscid Wall - cell-centered

(d) Inflow/Outflow - cell-centered

Fig. 3.8: Grid refinement study for various boundary formulations using Ringleb flow.

boundary type is tested in isolation by enforcing Dirichlet conditions on all boundaries but the one in question. The symmetric density contours of the exact solution are shown in Figure 3.7(b).

Figures 3.8-3.9 show the results for all boundary conditions listed in Table 3.1. While only the density error is shown, momentum and energy errors exhibit the same convergence behavior. Four important observations are made in reference to these figures. First, one can observe in Figures 3.8(a) and 3.8(c) a substantial decrease in accuracy for the n-INV2, n-INV3 and c-INV3 cases. These three conditions, which were all verified as second-order



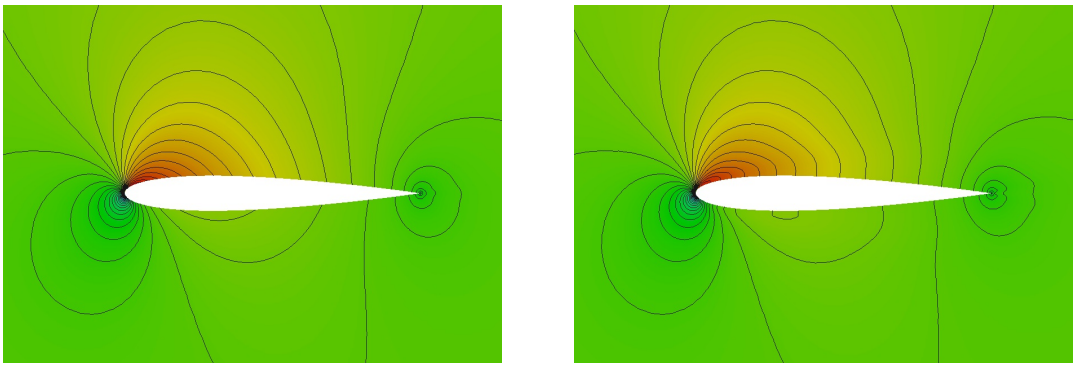
(a) Inviscid Wall - node-dentered

(b) Inviscid Wall - cell-centered

Fig. 3.9: Grid refinement study for inviscid wall boundary formulations using Ringleb flow using entropy as a measure of error.

accurate using MMS, now exhibit first-order accuracy when applied to Ringleb flow. At first this seems surprising, but may be explained by the fact that the MMS verification only addresses the mathematical correctness of the discretization of the chosen conditions, not the appropriateness of the conditions themselves. In all three failed methods, Neumann terms are included in the inviscid boundary conditions. The Neumann conditions are arbitrary and do not correctly represent the physics for this case. In fact, they contradict the Ringleb solution of the Euler equations, which explains the reduced accuracy. This case illustrates the critical importance of selecting appropriate boundary conditions for a given set of physics.

This point is further illustrated by considering subsonic flow over a NACA airfoil at $M = 0.5$ and $\alpha = 3.0^o$. Just as for Ringleb flow, this case also demonstrates that symmetry conditions are not suitable for inviscid walls. Figures 3.10(a)-3.10(b) show Mach contours for method n-INV1 and n-INV2, respectively. Near the surface, method n-INV2 exhibits oddly-shaped Mach contours as a result of the improper symmetry condition. Similar errors have been noted by Wang [27] and Barth [48]. While not shown, extrapolation conditions for cell-centered schemes predict correct Mach contours, while symmetry conditions again lead to erroneously shaped contours.



(a) method n-INV1. (b) method n-INV2 (symmetry).

Fig. 3.10: Mach contours for two node-centered inviscid wall treatments of a NACA 0012 inviscid airfoil at $M = 0.5$, $\alpha = 3.0^o$.

A second observation can be made by examining Figure 3.8(a), which shows the effect of approximating the boundary normal vector at nodal locations for method n-INV1. For Ringleb flow, the exact normal vectors may be deduced from the exact solution, which provides x and y locations along the inviscid wall streamline boundaries. Here, the exact normals are computed. The effect on the accuracy is compared to the approximate normal vectors computed as the average of the surrounding face normals at a node. Using exact normals and approximate normals both result in second-order behavior. However, the level of error using exact normals is nearly half the error using approximate normals. For other cases with more complex geometry, the effects may be amplified. Methods to compute boundary normals consistently for complex geometry are an area of future work.

A third observation is that all inflow and outflow conditions tested show sharp second-order accuracy, as shown in Figures 3.8(b) and 3.8(d). For the inflow and outflow conditions, specified quantities (entropy, total enthalpy, pressure, etc.) are taken directly from the exact solution. All of these methods are physically meaningful and compatible with the current problem. Thus, the MMS results for these conditions are validated. Again, this demonstrates that the MMS verification procedure is sufficient if physically consistent boundary equations are used. This is important because it provides a method for the verification of more complex boundary conditions and interior schemes for which no exact solutions are

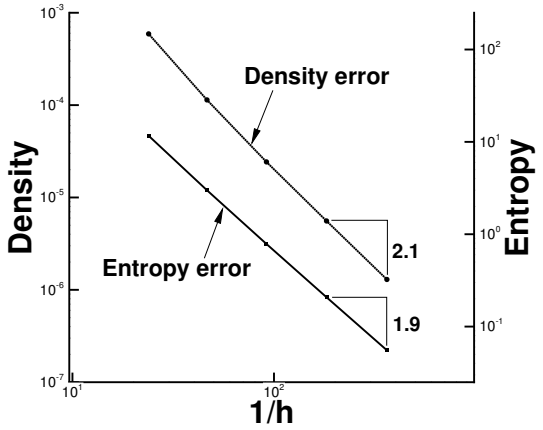


Fig. 3.11: Constant entropy inviscid wall condition, showing second order convergence, both in density and entropy.

known, such as averaged equations with turbulence models, multiphase flows, or combustion.

Finally, it is interesting to consider the impact of the boundary discretizations on other quantities besides the conserved variables, such as entropy. Entropy is sometimes used as a measure of error for inviscid flow cases with uniform inflow [27, 49, 50]. Figure 3.9 shows the convergence of surface entropy for all inviscid wall conditions in Table 3.1. This figure shows that none of the methods are truly second order accurate for entropy. Exact normals were used for the node-centered boundary conditions where possible. Recall that many of the methods were second order accurate considering the error in the conserved variables, as shown in Figures 3.8(a) and 3.8(c). This may be explained by the fact that isentropic flow is only attained if *all* the conservation laws are satisfied at all degrees of freedom. At the boundary locations, the full set of conservations laws are not solved. Thus, entropy may not demonstrate the same convergence behavior as the mesh is refined.

It is possible to formulate boundary conditions which preserve specific flow features, such as isentropic flow. Entropy-based boundary conditions have been explored by Balakrishnan and Fernandez [26], for example. As a check, a simple method to preserve entropy for inviscid flow with isentropic inflow conditions by specifying the entropy instead of the energy combination in Equation 3.9 is used. The third row in the null space matrix is set

to zero, and a condition $s - s_{spec} = 0$ is used instead, where $s = p/\rho^{\gamma}$ is a measure of the entropy. Using this condition, isentropic flow is preserved to the order of accuracy of the scheme, as shown in Figure 3.11. The figure shows that entropy converges as second order, just as density does. Extending this method to cases in which incoming entropy is not constant is an area of future research.

3.4 Conclusions

In this work, a general formulation for arbitrary boundary conditions for node- and cell-centered finite volume schemes is presented. These conditions may include any combination of Dirichlet, Neumann, extrapolation, and, in the case of node-centered schemes, the equations of motion themselves. The ease with which one may apply the conservation equations of motion is one advantage of node-centered schemes over cell-centered schemes. This avoids the need for extrapolation or extra Neumann conditions which may conflict with the interior scheme or physical configuration of the problem.

Importantly, the traditional use of the method of manufactured solutions (MMS) is extended to include boundary conditions. By defining a boundary residual equation to accompany the interior residual equation, MMS can be used in both node- and cell-centered contexts to verify the combined interior-boundary scheme. In this manner, a single manufactured solution can be used to verify any number of boundary conditions conveniently. Computation of boundary source terms is straightforward, generally requiring much less algebraic manipulation than the interior source terms.

A number of test cases, including quasi-1D flow and 2D flow demonstrate the use of the MMS procedure. It is found that the manufactured solution must be chosen such that the characteristic directions are respected during the boundary condition verification procedure to maintain well-posedness and stability. While the manufactured solution is arbitrary, boundary locations should be chosen to roughly coincide with the physics of the manufactured solution. If this practice is violated, erroneous results will be obtained or non-convergence will result.

While the correct choice of boundary conditions for a given problem is often not unique,

a key finding from studies involving Ringleb flow and inviscid airfoils is that certain boundary conditions may conflict with the interior scheme or the problem configuration. An example is the use of Neumann symmetry conditions for inviscid walls. Even though the symmetry condition is verifiably second-order accurate using the MMS procedure, it fails to produce the proper results when applied in an inappropriate physical context. Thus, the MMS procedure is only valid when used with physical insight to apply boundary conditions in the correct way. If used correctly, the MMS procedure provides a method for the verification of complex boundary conditions and interior schemes for which no exact solutions are known. In addition, it is found that it is best to use the conserved variables directly to measure the solution error, as opposed to derived quantities, such as entropy. Entropy appears to converge as second-order only when all conservation laws are satisfied, or when specialized conditions are used to preserve constant entropy.

Future work will focus on modifications of the node-centered approach to allow for a water-tight flux formulation. One possible approach is to introduce ghost nodes into the node-centered formulation in a manner similar to the cell-centered formulation. Another important area of investigation includes consistent methods for boundary normal computation. A possible strategy includes locally reconstructing the surface description from surrounding geometry information using a least squares procedure. Derivatives of the reconstructed surface may then be used to estimate normal vectors.

Chapter 4

Preconditioning

4.1 Introduction

Compressible time-marching CFD algorithms rely on iterations to remove numeric errors in two different ways: convection and dampening. Convection transports iterative error out of the domain through boundaries at speeds on the order of the particle and acoustic velocities. Large disparities between these two wave speeds at low Mach numbers can cause dramatic convergence deceleration. This is because the time scale a time-marching method must be able to resolve the fastest waves (typically acoustic) in the domain. When the particle waves speeds are much lower in magnitude, the higher acoustic resolution becomes unsuitable for the relatively slow moving particle waves, requiring many iterations to remove errors at these speeds. Preconditioners alter the magnitude of the acoustic waves such that they are the same order of magnitude as the particle wave speeds in a pseudo-time approach. This results in faster and more efficient convergence [29–31, 33, 34].

Along with convection, dampening also removes iterative errors from the domain through dissipation of high-frequency error. After decades of research, it appears that some form of artificial dissipation is essential for stability where physical viscosity effects are small or non-existent, such as for the inviscid Euler equations. Many different models have been proposed to effectively introduce artificial dissipation into the problem, including scalar, matrix, and hybrid schemes. Since artificial dissipation is generally constructed based on the eigenvalues (wave speeds) inherent in the problem, preconditioning modifies the dissipation operator. This has an effect on convergence and steady-state accuracy at low Mach numbers.

A steady state solution is found once numeric errors have been completely removed.

This solution is still capable of differing from the real world solution. Assumptions such as discretization linearity and simplified physics can cause these real errors. The preconditioning techniques explored in this work are capable of increasing the accuracy of steady state solutions.

Unsteady (time independent) problems present additional challenges for preconditioning. Unsteady problems not only rely on convection and dissipation for convergence, but have physically relevant acoustic and particle waves as part of the time dependent solution itself. Capturing the physical time scales inherent in the problem is the goal of an unsteady simulation. Preconditioners must be capable of optimizing convergence of the error propagating waves, as well as capturing the unsteady dynamics of the problem. Devising such "unsteady preconditioners" can be difficult to formulate and is the topic of this chapter.

Much work has been done using Roe diffusion with unsteady preconditioners [1,29,51]. Hosengadi et al. [1] show that Roe dissipation fails in low speed flows with small time steps. To fix this issue, Liou [52,53] developed a new AUSM$^+$-up scheme that is capable of accurately resolving both the pressure and velocity fields in this regime. Potsdam et al. [51] propose a blended Roe scheme that is capable of improved accuracy.

This work focuses on a new method of dissipation used in conjunction with preconditioning to accurately solve highly unsteady low speed flows. A modified CUSP artificial dissipation scheme is developed that provides improvements to traditional Roe dissipation methods in certain flow regimes. Similar to Roe, CUSP's formulation relies on the wave speeds in it's definition. The modification to these speeds through preconditioning causes instabilities in traditional CUSP and AUSM schemes. The addition of pressure dissipation in the AUSM$^+$-up scheme was shown to be effective. Here, this pressure dissipation is adapted to take the form of CUSP. CUSP's basic advantages lie in the fact that CUSP splits the pressure dissipation from the flow field. This allows the modified CUSP scheme to increase accuracy of results over preconditioned Roe dissipation.

This chapter is outlined as follows: First, a general preconditioning theory and framework is discussed for steady and unsteady flows. Second, existing and new preconditioning

schemes are formulated, with a focus on artificial dissipation operators. Third, results are presented for steady and unsteady flows. The unsteady flows presented include a convection dominated case (propagating inviscid vortex) and a pressure dominated case (1D pipe flow with oscillating back-pressure). Finally, conclusions based on the studies are given.

4.2 Preconditioning Scheme

Preconditioning schemes are used to adjust steady-state waves to help increase convergence, and in some cases accuracy. The purpose of this section is to give the basics of preconditioning implementation. General preconditioners are presented following the framework that Merkle [34] presented. Details of different preconditioned sound speeds are also presented. For simplicity, the following sections all address two-dimensional problems. It is simple to extend these details to either 1D or 3D codes.

4.2.1 Conversion to Primitive Variables

In two-dimensions, the steady-state inviscid Euler equations can be represented by the following equation.

$$\frac{\partial Q}{\partial \tau} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 \tag{4.1}$$

where

$$Q = \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{Bmatrix}, \quad F = \begin{Bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho u h^0 \end{Bmatrix}, \quad G = \begin{Bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho v h^0 \end{Bmatrix} \tag{4.2}$$

Here, Q represents the conserved variable vector and F and G represent the inviscid fluxes in the x- and y-directions respectively. The preconditioning method outlined by Merkle requires that the conservered variable vector, Q, be converted to primitive variables, Q_v.

$$Q_v = \begin{Bmatrix} p \\ u \\ v \\ T \end{Bmatrix} \tag{4.3}$$

The use of primitive variables are primarily for convenience in derivation, but do have some advantages over conserved variables. Primitive variables make it easier to implement arbitrary equations of state (ie ideal gas, super-critical, incompressible, etc.) to close the model. For example, enthalpy is often tabulated as a function of temperature. With conservative variables it is necessary to iteratively find temperature from the enthalpy. Since temperature is a known quantity of the primitive variables, the enthalpy can be extracted with little effort.

Conversion from conserved variables to primitive variables is possible through a Jacobian matrix, Γ. The following equation is equivalent to Eq. 4.1 by use of the chain rule.

$$\Gamma \frac{\partial Q_v}{\partial \tau} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0 \tag{4.4}$$

where

$$\Gamma = \frac{\partial Q}{\partial Q_v} = \begin{bmatrix} \rho_p & 0 & 0 & \rho_T \\ u\rho_p & \rho & 0 & u\rho_T \\ v\rho_p & 0 & \rho & v\rho_T \\ h^0\rho_p + \rho h_p - 1 & \rho u & \rho v & h^0\rho_T + \rho h_T \end{bmatrix} \tag{4.5}$$

Subscripts p and T refer to partial derivatives with respect to that variable. For example,

$$\rho_p = \frac{\partial \rho}{\partial p} \tag{4.6}$$

Here, it is assumed that density (ρ) and enthalpy (h) are functions of only pressure (p) and temperature (T). For example, density and enthalpy are defined for an ideal gas in the following equation.

$$\rho = \rho(p,T) = \frac{p}{RT}, \quad h = h(p,T) = \frac{R\gamma}{\gamma - 1}T \tag{4.7}$$

This conversion of variables allows for a simple yet robust preconditioner to be applied.

4.2.2 Preconditioning Terms

In order to understand how the preconditioner will be added to the new primitive variable equation, it is imperative to understand the convective wave dynamics of Eq. 4.4. Eq. 4.4 can be rewritten using new Jacobian matrices $\mathcal{A}_x$ and $\mathcal{A}_y$ as shown below.

$$\Gamma\frac{\partial Q_v}{\partial \tau} + \mathcal{A}_x\frac{\partial Q_v}{\partial x} + \mathcal{A}_y\frac{\partial Q_v}{\partial y} = 0 \tag{4.8}$$

The convective waves travel at the speed of the Eigenvalues (λ) of this problem. λ can be found by solving the simple Eigenvalue problem presented below.

$$(\Gamma^{-1}\mathcal{A} - \lambda I) = 0 \tag{4.9}$$

where I is the identity matrix. With the discretization scheme presented in Chapter 2 these Eigenvalues can be found to be,

$$\lambda = \left\{ q_a, q_a, q_a \pm Ac \right\}^T \tag{4.10}$$

where

$$q_a = A_x u + A_y v, \quad A = \sqrt{A_x^2 + A_y^2} \tag{4.11}$$

Here, c represents the physical acoustic sound speed, A represents the face area with x and y components A_x and A_y, and q_a is the face area weighted particle velocity. The area weighted terms are used here for 2D finite volume flows.

The scheme, thus far, has yet to assume an equation of state to close the model. Consistent with this, it is necessary to define the physical sound speed in an arbitrary

manner. Eq. 4.12 shows this arbitrary defintion

$$c^2 = \frac{\rho h_T}{\rho h_T \rho_p + \rho_T(1 - \rho h_p)} \tag{4.12}$$

As previously discussed, with a large disparity in the magnitude of these waves, convergence and efficiency suffers. To attain fast convergence at low particle velocities, the Jacobian Γ is replaced by a new Γ_p. Γ_p is selected to ensure that the wave speeds are of the same order of magnitude. Γ_p can only apply to the pseudo-time terms to maintain time accuracy of the simulation, since the real-time waves need to remain un-preconditioned. Γ_p is defined below.

$$\Gamma_p = \begin{bmatrix} \rho'_p & 0 & 0 & \rho_T \\ u\rho'_p & \rho & 0 & u\rho_T \\ v\rho'_p & 0 & \rho & v\rho_T \\ h^0\rho'_p + \rho h_p - 1 & \rho u & \rho v & h^0\rho_T + \rho h_T \end{bmatrix} \tag{4.13}$$

Γ_p is nearly identical to the actual Jacobian where ρ'_p replaces ρ_p. ρ'_p is a new term that is not related to the physical derivatives. Selection of the best definition is part of the study done here. To leave this derivation as general as possible, ρ'_p is purposefully not defined at this moment, but is addressed in the next section. The general Eigenvalues for any selection of ρ'_p are shown below.

$$\lambda = \begin{Bmatrix} q_a \\ q_a \\ \frac{1}{2}\left[q_a\left(1 + \frac{d}{d'}\right) \pm \sqrt{q_a^2\left(1 - \frac{d}{d'}\right)^2 + 4\frac{\rho h_t}{d'}A^2}\right] \end{Bmatrix} \tag{4.14}$$

where

$$\begin{aligned} d = &\quad \rho h_T \rho_p + \rho_T(1 - \rho h_p) \\ d' = &\quad \rho h_T \rho'_p + \rho_T(1 - \rho h_p) \end{aligned} \tag{4.15}$$

For preconditioning to be effective it is necessary for the acoustic wave speeds to be

the same order of magnitude as the particle speeds. $\lambda_{3,4}$ in Eq. 4.14 contain both particle waves and acoustic waves. Sankaran [29] and Merkle [34] showed that the acoustic waves for preconditioned problems now travel at a new speed V_p. From Eq. 4.14 the "preconditioned sound speed" can be written as

$$V_p^2 = \frac{\rho h_t}{d'} \tag{4.16}$$

By selecting V_p to be the same order of magnitude as the particle wave speeds, the goal of preconditioning is achieved. Once the preconditioned sound speed is selected it is possible to back out the definition of ρ'_p as

$$\rho'_p = \frac{1}{V_p^2} - \frac{\rho_T(1 - \rho h_p)}{\rho h_T} \tag{4.17}$$

The selection of V_p drives the definition of ρ'_p in the preconditioning Jacobian matrix Γ_p. The next sections show two popular methods for defining the preconditioned sound speed.

Steady Preconditioning

Assuming a problem is steady state simplifies the model by letting physical time terms go to zero. Steady state problems only have convective waves in pseudo-time. Modification of these wave speeds can increase convergence, but not change the order of the scheme. For this case, the artificial steady state sound speed (V_p^s) is set to the particle wave speed,

$$V_p^s = \sqrt{u^2 + v^2} \tag{4.18}$$

At low speeds, defining V_p^s this way increases convergence and can reduce the error [29]. When Mach numbers are greater or equal to 1, there is no need to adjust the acoustic waves. In fact, setting the artificial sound speed higher than the physical sound speed reduces accuracy [29]. To limit the steady state artificial sound speed a simple minimum function is used.

$$V_p^s = min(\sqrt{u^2 + v^2}, c) \tag{4.19}$$

In supersonic regimes, Eq. 4.19 will set the artificial sound speed to the actual sound speed. This reduces the Eigenvalues to the their non-preconditioned form.

Unsteady Preconditioning

When physical time terms can no longer be neglected, the problem becomes unsteady. The numerics to solve unsteady problems relies on stepping through time in some small increment, Δt. As Δt becomes small, the physical acoustics dominate the problem. Capturing these physical acoustics accurately requires that the sound speed not be altered. If it is altered, convergence slow downs and accuracy problems may arise [1].

The Strouhal number (Str) is a dimensionless number that relates the time step to a reference length and particle velocity. In effect, the larger the Strouhal number, the more acoustically driven the problem is.

$$Str = \frac{L}{\pi \Delta t \sqrt{u^2 + v^2}} \tag{4.20}$$

To create the unsteady preconditioner V_p^u, the Strouhal number is introduced to Eq. 4.19 to create,

$$V_p^u = min\left(max\left(\sqrt{u^2 + v^2}, Str\sqrt{u^2 + v^2}\right), c\right) \tag{4.21}$$

There are several important characteristics of the unsteady preconditioner in Eq. 4.21. First, as Δt becomes small and Str becomes large, the scheme reduces to a non-preconditioned state. This occurs because the preconditioned sound speed is "capped" at c. This behavior is desirable to resolve the physically acoustic waves. The other main characteristic is that as time steps become large (and the Str decreases) V_p^u behaves as the steady state preconditioner. This intuitively makes sense, as the particle velocities are now more important and the time scale is less significant. It is also consistent with the idea that a steady state

problem has an infinite time step. The addition of the Strouhal number bridges the gap between steady and acoustically driven unsteady problems.

Stagnation Points

Stagnation points are areas of the flow where particle velocities are zero. This is typical in external aerodynamics where a leading wall interacts with a flow, such as the tip of an airfoil. Both of the previous definitions break down at stagnation points. When the cell reference velocity becomes zero, Eq. 4.17 becomes invalid due to a division by zero. In the application of external flows, a free stream velocity is easily found. Using the free-stream velocities near stagnation points is a prominent way of dealing with this problem. For internal combustion-like flows, a free-stream value is not readily available and is not easily definable. Using the particle velocity from near by cells as a reference at the stagnation point solves this issue. The cases with stagnation points in this chapter were solved using the free-stream velocities near stagnation regions.

4.3 Challenges with Dissipation Terms

Artificial dissipation is an essential term in the fluxes for stability. The numeric flux is defined as,

$$\hat{\mathcal{F}}_k = \frac{1}{2}\left(\mathcal{F}(Q_L) + \mathcal{F}(Q_R)\right) - D_k(Q_R, Q_L) \tag{4.22}$$

where D_k is the artificial dissipation at face k.

Many different models exist to define D_k. Roe dissipation is widely used in preconditioning application and is explored here. Much work has already been done with preconditioners, the goal of presenting it here is to validate that the models used in thesis corroborate with previously found work. There are known issues with preconditioners and Roe diffusion. For this reason a new modified CUSP is introduced. The basics of the original CUSP scheme are presented in this section.

4.3.1 Roe

Roe discovered that the fluxes can be written in terms of the flux Jacobian and the conserved variables, if the flux Jacobian is computed at the "Roe averaged state." He was successful in showing that

$$F_R - F_L = \mathcal{A}_k(Q_R - Q_L) \tag{4.23}$$

where $\mathcal{A} = \frac{\partial F}{\partial Q}$ is the flux Jacobian. For the previous equation to hold true, $\mathcal{A}$ must be computed using quantities that are averaged in a special way from the right and left states.

$$u_k = \frac{\sqrt{\rho_L}u_l + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad v_k = \frac{\sqrt{\rho_L}v_l + \sqrt{\rho_R}v_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad h_k^0 = \frac{\sqrt{\rho_L}h_l^0 + \sqrt{\rho_R}h_R^0}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{4.24}$$

By using this relationship, Roe was able to create a widely popular dissipation scheme that is capable of stabilizing the the numeric fluxes.

$$D_k = \frac{|\mathcal{A}_k|}{2}(Q_R - Q_L) = \frac{X_k |\Lambda_k| X_k^{-1}}{2}(Q_R - Q_L) \tag{4.25}$$

where X_k and X_k^{-1} represent the left and right Eigenvectors, Λ_k is a diagonal matrix of the Eigenvalues, and subscript k indicates the values are found at the face between cells. The absolute value of the Eigenvalues helps to ensure that the proper wave directions are maintained in the numeric fluxes. This method is often called Matrix dissipation.

It is straightforward to apply Roe's matrix dissipation to primitive variables with preconditioning.

$$D = \Gamma_v \left|\Gamma_v^{-1}\mathcal{A}_v\right|(Q_{vR} - Q_{vL}) = \Gamma_v X_v |\Lambda_v| X_v^{-1}(Q_{vR} - Q_{vL}) \tag{4.26}$$

The preconditioned Roe scheme provides stability for the new preconditioned wave speeds. X_v and X_v^{-1} are defined below:

$$X_v = \begin{bmatrix} 0 & 0 & \frac{\tilde{\lambda}_4 - q_n \frac{d}{d'}}{\tilde{\lambda}_4 - \tilde{\lambda}_3} & \frac{\tilde{\lambda}_3 - q_n \frac{d}{d'}}{\tilde{\lambda}_3 - \tilde{\lambda}_4} \\ 0 & -n_y & \frac{n_x}{\rho(\tilde{\lambda}_3 - \tilde{\lambda}_4)} & \frac{n_x}{\rho(\tilde{\lambda}_4 - \tilde{\lambda}_3)} \\ 0 & n_x & \frac{n_y}{\rho(\tilde{\lambda}_3 - \tilde{\lambda}_4)} & \frac{n_y}{\rho(\tilde{\lambda}_4 - \tilde{\lambda}_3)} \\ 1 & 0 & \frac{1-\rho h_p}{\rho h_T} \frac{\tilde{\lambda}_4 - q_n \frac{d}{d'}}{\tilde{\lambda}_4 - \tilde{\lambda}_3} & \frac{1-\rho h_p}{\rho h_T} \frac{\tilde{\lambda}_3 - q_n \frac{d}{d'}}{\tilde{\lambda}_3 - \tilde{\lambda}_4} \end{bmatrix} \tag{4.27}$$

$$X_v^{-1} = \begin{bmatrix} -\frac{1-\rho h_p}{\rho h_T} & 0 & 0 & 1 \\ 0 & -n_y & n_x & 0 \\ 0 & \rho \left[\tilde{\lambda}_3 - q_n \frac{d}{d'} \right] n_x & \rho \left[\tilde{\lambda}_3 - q_n \frac{d}{d'} \right] n_y & 0 \\ 0 & \rho \left[\tilde{\lambda}_4 - q_n \frac{d}{d'} \right] n_x & \rho \left[\tilde{\lambda}_4 - q_n \frac{d}{d'} \right] n_y & 0 \end{bmatrix} \tag{4.28}$$

where

$$n_x = \frac{A_x}{A}, \quad n_y = \frac{A_y}{A}, \quad q_n = n_x u + n_y v, \quad \tilde{\lambda}_{3,4} = \frac{\lambda_{3,4}}{A} \tag{4.29}$$

Extensive research has been done on Roe dissipation in conjunction with both steady and unsteady preconditioners. Hosengadi [1] showed through asymptotic analysis the general trend of behavior for these schemes. A table showing the effects of Strouhal number and Mach number have on the accuracy can be found in Table 4.1.

From Table 4.1, it is clear that Roe is sufficient for steady or low Strouhal number problems. Roe dissipation breaks down when the Strouhal number is high and Mach number is low. In these cases, the different preconditioning schemes struggle in different areas. Steady preconditioning is capable of resolving velocity fields, but pressure (acoustic) suffers. The opposite is true of the unsteady preconditioner. These results are validated in section 4.4.

4.3.2 CUSP Background

Roe Diffusion is computationally expensive. Inversion and creation of large matrices are costly. CUSP was originally introduced to be more computationally efficient than Roe as well as have better upwinding and shock capturing characteristics. Jameson [36] presented a similar framework to Roe, but with constants (α^* and β) that are less computationally

Table 4.1: Description of how preconditioned effects convergence (O(1) is preferred) using a Roe diffusion scheme [1].

Steady Low Mach Number Limit

	Pressure Field	Velocity Field
No Preconditioning	O(M)	O(1/M)
Steady Preconditioning	O(1)	O(1)

Unsteady Low Mach/ Low-Str Number Limit

	Pressure Field	Velocity Field
No Preconditioning	O(M)	O(1/M)
Steady Preconditioning	O(1)	O(1)
Unsteady Preconditioning	O(1)	O(1)

Unsteady Low Mach/ High-Str Number Limit

	Pressure Field	Velocity Field
No Preconditioning	O(1)	O(1/M)
Steady Preconditioning	O(Str)	O(1)
Unsteady Preconditioning	O(1)	O(1/M)

expensive to compute. In two dimensions with area weighted terms this can be expressed by,

$$D = \frac{1}{2}\alpha^* cA\Delta Q + \frac{1}{2}\beta\left(\mathcal{F}(Q_R) - \mathcal{F}(Q_L)\right) \tag{4.30}$$

where Δ refers to the difference between right and left states. For example,

$$\Delta Q = Q_R - Q_L \tag{4.31}$$

Using simple algebra and several substitutions the above equations can be rewritten into a more convenient form:

$$D = \frac{1}{2}\alpha Ac\Delta Q + \frac{1}{2}\beta\bar{Q}\Delta q_a + \frac{1}{2}\beta\Delta f_p \tag{4.32}$$

where

$$\alpha Ac = \alpha^* Ac + \beta \bar{q_a}, \quad f_p = \left\{ \begin{array}{c} 0 \\ A_x p \\ A_y p \\ q_a p \end{array} \right\} \tag{4.33}$$

and an over-bar refers to the arithmetic average of the right and left states.

α and β are chosen to give optimal characteristics in subsonic, transonic and supersonic regions. In supersonic regions, true upwinding is desired. Upwinding allows information at on faces to be only a function of the cells that lie "up wind" from the face. One point on the interior of the transonic shock also provides crisp and efficient shock waves. In subsonic regions the directions of the waves is persevered in the fluxes. The details of derivation of what α and β should be are omitted in this work [36, 54]. They are defined below:

$$\alpha = |M_a| \tag{4.34}$$

$$\beta = \left\{ \begin{array}{cc} max(0, 2M_a - 1) & \text{if } 0 \leq M_a \leq 1 \\ -max(0, 2M_a + 1) & \text{if } -1 \leq M_a \leq 0 \\ sign(M_a) & \text{if } |M_a| \geq 1 \end{array} \right. \tag{4.35}$$

M_a in the above equations is the local area weighted Mach number, $M_a = \frac{q_a}{Ac}$. This distinction in 2D is important. Using the area weighted Mach number ensures that diffusion is only introduced on faces where fluxes cross the boundaries of the cell.

CUSP has clear advantages in numerical efficiency, drastically reducing the number of computations required. CUSP's ability to discretely capture shocks and upwind for supersonic flows demonstrate some the accuracy advantages of CUSP. A primary disadvantage of CUSP is that when linearized, it is not diagonally dominant. Many implicit architectures, such as line-implicit Gauss-Siedel used in this work, require the left hand side (LHS) of the equations be diagonally dominant. Roe diffusion naturally can be linearized to meet this requirement. For the tests in this chapter, Roe diffusion is used for the LHS regardless of

which method is implemented on the right hand side of the equation.

4.3.3 Modified CUSP Scheme

The limitations on preconditioned Roe diffusion highlighted by Table 4.1 has been studied by a number of researchers. Potsdam [51] created a blended Roe scheme to remedy the issue. Potsdam et al. used selection matrices to combine the pressure fields of the unsteady preconditioner and the velocity fields of steady preconditioners. Although effective, this work focuses on the benefits of CUSP's split pressure formulation to generate accurate solutions in the low Mach number, high Strouhal number limit.

The CUSP scheme previously discussed breaks down when the preconditioning schemes are introduced at low Mach numbers. Liou et al. [53] showed a similar trend with AUSM, a CUSP-like dissipation scheme. To solve this issue in AUSM, Liou introduced two new forms of dissipation into the AUSM scheme; added pressure dissipation for the interfacial Mach number and additional velocity diffusion to the pressure terms. In order to determine to correct dissipation additions required, Liou relied on asymptotic analysis. This analysis is not repeated here.

While the addition of pressure dissipation is readily accepted in research [1, 37, 38], the velocity dissipation is much more controversial. Liou's velocity dissipation terms are designed to only have an effect for transonic and supersonic flows and to have no impact on low speed flows. Hosengadi [1] showed that it was necessary for shock like applications to remove pressure oscillations, but that the amount of dissipation added was dramatic and had an adverse impact on accuracy.

Following the work of Liou, the additional pressure diffusion is added to CUSP in this work. Since the focus is on low speed flows, velocity dissipation is omitted. Liou's pressure dissipation term is written in a form that is consistent with CUSP below.

$$\gamma A \bar{c} \bar{Q} \Delta p \tag{4.36}$$

where

$$\gamma = \frac{K_p}{f_a} \frac{max(1 - \alpha \bar{M_e}^2, 0)}{\bar{\rho}\bar{c}^2} \tag{4.37}$$

and

$$f_a = M_o(2 - M_o), \quad M_o^2 = min(max(\bar{M_e}^2, \bar{M_e}^2 Str^2), 1) \tag{4.38}$$

$$M_e = \frac{\sqrt{u^2 + v^2}}{c} \tag{4.39}$$

K_p is a constant between 0 and 1. Liou proposes that a value of 0.25 is effective. This value is used for the results of this chapter.

It is interesting to note that the Mach number (M_e) for the added pressure dissipation cannot be the area weighted Mach number as used for the traditional CUSP scheme. It is possible for the area weighted Mach number to become 0 on certain faces, which causes divide by 0 errors in the γ term. For this reason and to be consistent with the preconditioner definitions, the actual local Mach number is used.

The newly modified CUSP scheme can be written entirely by

$$D = \frac{1}{2}\alpha A c \Delta Q + \frac{1}{2}\beta \bar{Q} \Delta q_a + \frac{1}{2}\beta \Delta f_p + \gamma A \bar{c} \bar{Q} \Delta p \tag{4.40}$$

4.4 Results

In this section, many different test scenarios are presented to highlight the new CUSP scheme. First, it is necessary to validate and verify that both Roe and CUSP were implemented properly. Verification of the interior scheme is done using MMS solutions. A steady state airfoil is analyzed for validation. Second, in order to isolate the velocity field from the pressure field, two problems are introduced: an inviscid traveling vortex and 1D pipe with a time oscillating pressure on the outflow. The vortex is a velocity driven problem, whereas the pipe is an acoustically driven problem. The results from Table 4.1 are validated using Roe dissipation and compared to the new CUSP algorithm described by this chapter.

Table 4.2: List of test scenarios for Fig .4.1

Symbol	Test Scenario
R^N	Roe dissipation with No preconditioning
R^S	Roe dissipation with Steady preconditioning
C^N	CUSP dissipation with No preconditioning
C^S	CUSP dissipation with Steady preconditioning

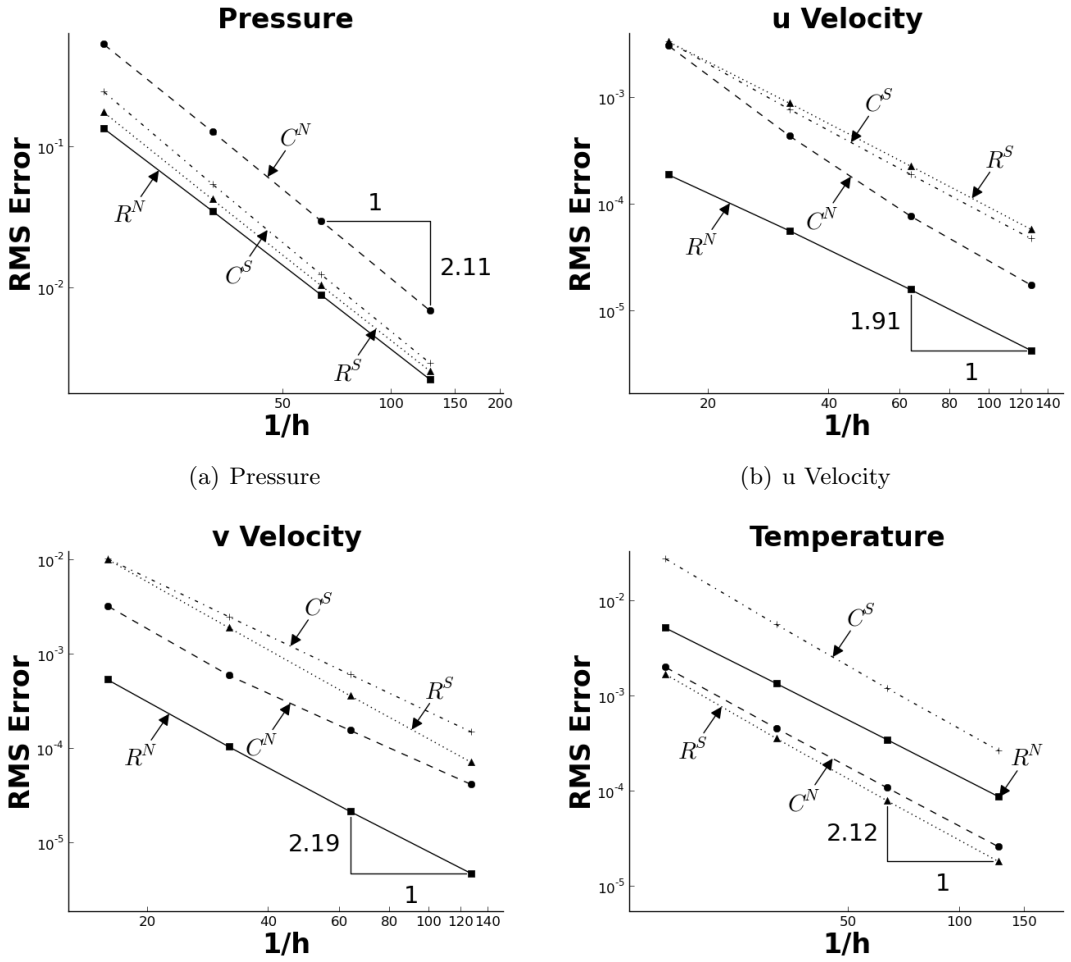
Convergence of these cases is demonstrated to show the effect of low-speed preconditioning. Although many tests were performed to understand the physics and numerics of these particular problems, only the fundamentally important cases are shown in this section. All tests in this section are performed on 2D strand grids.

4.4.1 Verification and Validation

Verification provides a robust way of confirming numeric accuracy and code implementation. It is beneficial to use MMS verification any time a new scheme is being introduced. This simple check was performed in Fig. 4.1 using steady state MMS. The different lines and symbols found throughout Fig. 4.1 are described by Table 4.2.

The MMS verification is able to show that for all quantities and methods, the output is second order accurate and behaves properly. It is interesting to observe that although all lines are second order accurate, the accuracy is clearly changing. This could be due to a number of factors. Regardless of whether the RHS is using CUSP or Roe, the LHS requires the use of the diagonally dominant Roe scheme. For the parameters of source terms chosen to define the MMS solution, the optimum preconditioned pressure additions of CUSP could be providing an excessive amount of dissipation.

To get general validation of the dissipation schemes, a NACA 0012 airfoil was tested at a relatively low Mach number. Fig. 4.2 shows the pressure contours for a variety of cases. A free-stream cut-off velocity was specified for preconditioning at the stagnation point and is necessary for convergence. Fig. 4.2(c) shows the solution without preconditioning. The stagnation point is not captured in the detail as the two preconditioned cases found



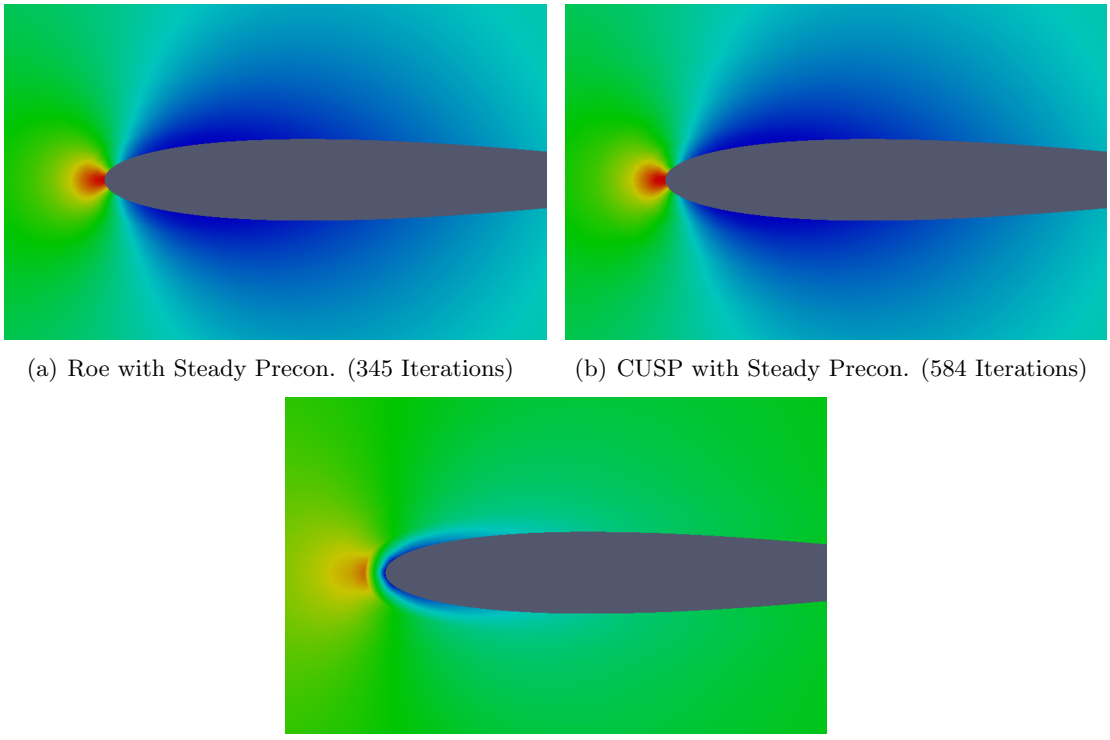
(a) Pressure

(b) u Velocity

(c) v Velocity

(d) Temperature

Fig. 4.1: Verification plots using steady state MMS for primitive variables p, u, v, and T with a Mach number of 0.05. Symbols are defined by Table 4.2



(a) Roe with Steady Precon. (345 Iterations)

(b) CUSP with Steady Precon. (584 Iterations)

(c) Roe without Precon. (2724 Iterations)

Fig. 4.2: Pressure contours of a 2D NACA 0012 airfoil with free stream Mach number of 0.05 with no angle of attack.

in Figs. 4.2(a) and 4.2(b). Near the stagnation point, the particle velocity approaches zero. This creates a very large disparity in the wave speeds. Although the solution in the non-preconditioned case converges, it is possible for the overall convergence criteria to be reached where large errors are still propagating locally near the stagnation point. Local preconditioning aids in overall convergence, as well as the convergence at the stagnation point.

The iterations for each airfoil is also reported in Fig. 4.2. Without preconditioning, dramatic efficiency problems are seen. Implementation of CUSP slows down convergence slightly. This is generally attributed to the difference between the LHS and RHS in the implementation. This simple case highlights how preconditioning is able to increase the accuracy of the solution as well as aids in convergence.

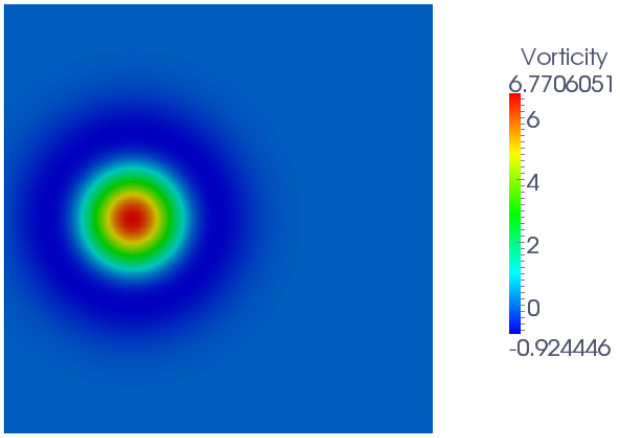


Fig. 4.3: Initial contour of vorticity on a 64 by 64 strand mesh.

4.4.2 Inviscid Propagating Vortex

Inviscid vortices are primarily velocity driven problems. Although a pressure field exists, pressures only respond to the velocity field. An inviscid propagating vortex is an unsteady problem, where some vortex travels at a free stream (U_∞) velocity in time. Ideally, the vortex will maintain it's shape with time and simply translate through the domain. CFD discretization creates diffusion that will introduce errors into the solution. To test this phenomena, a simple 2D vortex is used to initialize the domain. Unsteady time steps are used to allow the vortex to propagate across the domain. In this simplified case, it is assumed the free stream velocities only exist in the x direction.

The vortex is initialized using several user defined parameters. R is the radius of the vortex, β is the strength of the vortex, and x_c and y_c are used to specify its starting point. Fig. 4.3 shows a sample 2D initialization of the vorticity over a square domain. The following equations define how the vortex is initialized and assume an ideal gas equation of state.

$$\delta u = -U_\infty \beta \frac{y-y_c}{R} e^{-r^2/2}$$

$$\delta v = U_\infty \beta \frac{x-x_c}{R} e^{-r^2/2} \tag{4.41}$$

$$\delta T = 0.5(U_\infty \beta)^2 e^{-r^2}/c_p$$

where

$$u_0 = U_\infty + \delta u, \quad v_0 = \delta v, \quad T_0 = T_\infty - \delta T$$

$$\rho_0 = \rho_\infty \left(\frac{T_0}{T_\infty}\right)^{\frac{1}{\gamma-1}}, \quad \rho_\infty = \frac{P_\infty}{R_{gas} T_\infty}, \quad P_0 = \rho_0 * R_{gas} T_0 \tag{4.42}$$

$$r = \frac{\sqrt{(x-x_c)^2+(y-y_c)^2}}{R}, \quad U_\infty = M_\infty \sqrt{\gamma R_{gas} T_\infty}$$

Due to the unsteady nature of the problem, the boundaries of the domain become difficult to enforce. The exact solution as a function of time was imposed on the boundaries. This may cause issues as the vortex enters or leaves the domain. To avoid this issue, boundaries were placed "far" from the edges of the vortex.

In the absence of viscous terms, the exact solution should allow the vortex to travel with no diffusion at the free stream velocity. The exact solution can be represented by the initialization equations from Eqs. 4.41 and 4.42 where the center is moving. The new center is found as

$$x_c(t) = x_{c0} + U_\infty t, \quad y_c(t) = y_{c0} \tag{4.43}$$

At a given time, the exact vorticity can be found to be

$$\omega = \frac{U_\infty \beta}{R} e^{-r^2/2}(2 - r^2) \tag{4.44}$$

For unsteady problems, selection of the physical time step is critical. It is important to choose a time step to capture the flow dynamics efficiently and correctly. The "CFL"

number relates a velocity, time step, and cell width to help specify the time step.

$$CFL = \frac{\mathcal{U}dt}{dx} \tag{4.45}$$

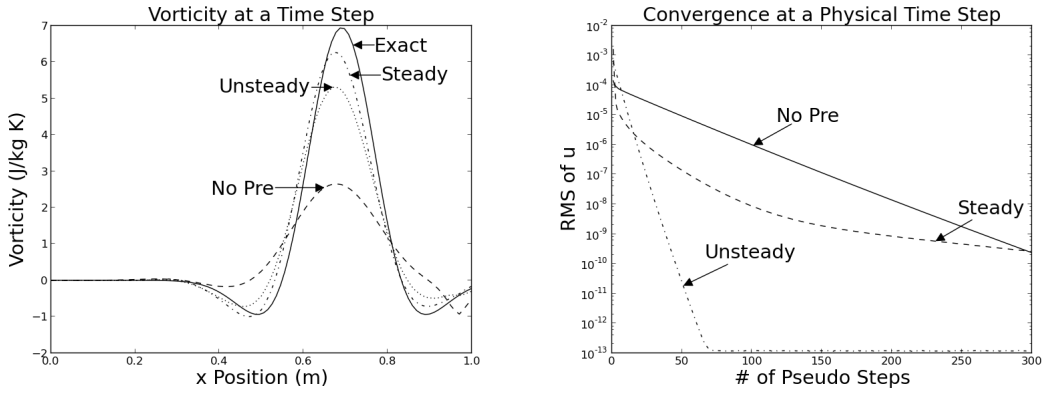
where $\mathcal{U}$ is some velocity scale, dx is typically taken as the average cell width and dt is the physical time step. $\mathcal{U}$ can be chosen to be either the particle speed (CFL_u) or the acoustic speed (CFL_c). $CFL_u = 1$ allows the particles to move about 1 cell per time step, where $CFL_c = 1$ allows the physical acoustics do the same. These two situations are of current interest specifically how the new CUSP modification and preconditioners behave.

In order to validate the accuracy of Table 4.1, as well as the implementation of the preconditioners, Roe diffusion was tested. For this velocity driven case, Table 4.1 indicates that using no preconditioner and the unsteady preconditioner in the high Strouhal limit will yield reduced accuracy.

Fig. 4.4(a) shows the data for a "snapshot" of vorticity at some time after the vortex is allowed to translate down stream for various preconditioners. The vorticity is taken at the centerline. The steady preconditioner captures the vorticity (or velocity field) much better than the non-preconditioned case. The introduction of the Strouhal number in the unsteady case, works to "blend" the steady preconditioner to the un-preconditioned case. Here, the time step is large enough that the high Strouhal limit is not reached.

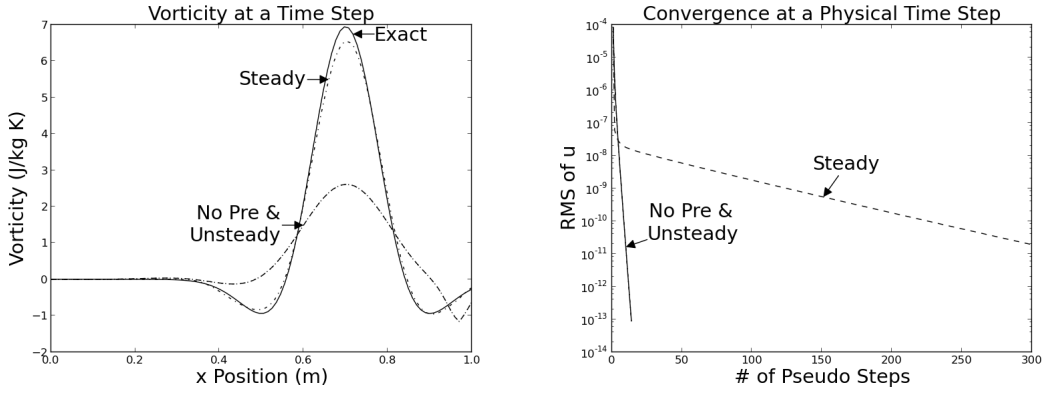
Fig. 4.4(b) shows the convergence of the psuedo-time terms at a given time step. The unsteady preconditioner outperforms the other schemes. By converging with fewer iterations, the unsteady preconditioner decreases the computation time required. Fig. 4.4(b) highlights just one of many time steps. As the number of time steps in a problem grows, this speed up becomes increasingly important. The results from Fig. 4.4 are consistent with those found in Hosengadi et al. [1] and with Table 4.1.

To explore the effects of higher Strouhal numbers, the CFL number was lowered. Fig. 4.5 shows a similar case to Fig. 4.4, but with a $CFL_c = 1$ rather than $CFL_u = 1$. The reduction in the time step causes a dramatic increase to the Strouhal number. In this case, the unsteady preconditioner reduces to the un-preconditioned state, consistent with



(a) Vorticity (b) Convergence

Fig. 4.4: Plots of propagating vortex with Roe diffusion on a square strand domain with a $CFL_u = 1$, $Str = 20.4$, $Mach = 0.005$, and 20 points across the vortex.

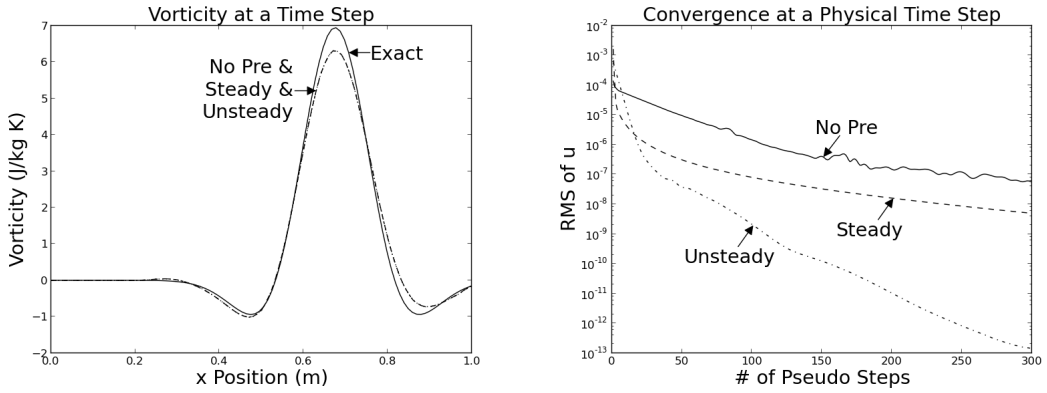


(a) Vorticity (b) Convergence

Fig. 4.5: Plots of propagating vortex with Roe diffusion on a square strand domain with a $CFL_c = 1$, $Str = 4076$, $Mach = 0.005$, and 20 points across the vortex.

its definition. The vorticity plot found in Fig. 4.5(a) shows this behavior and highlights the unsteady preconditioners inability to accurately describe the velocity field at high Strouhal numbers. Again, Fig. 4.5(b) shows that the steady preconditioner suffers to efficiently converge in this highly unsteady case.

The unsteady preconditioning method is extremely effective at converging efficiently. When the time step is very small, it outperforms the steady preconditioner. As time steps grow, it is capable of reducing to the steady preconditioner. This process is shown by Fig. 4.4(b). The convergence advantages of the unsteady preconditioner are offset in this



(a) Vorticity (b) Convergence

Fig. 4.6: Plots of propagating vortex with CUSP diffusion on a square strand domain with a $CFL_u = 1$, $Str = 20.4$, $Mach = 0.005$, and 20 points across the vortex.

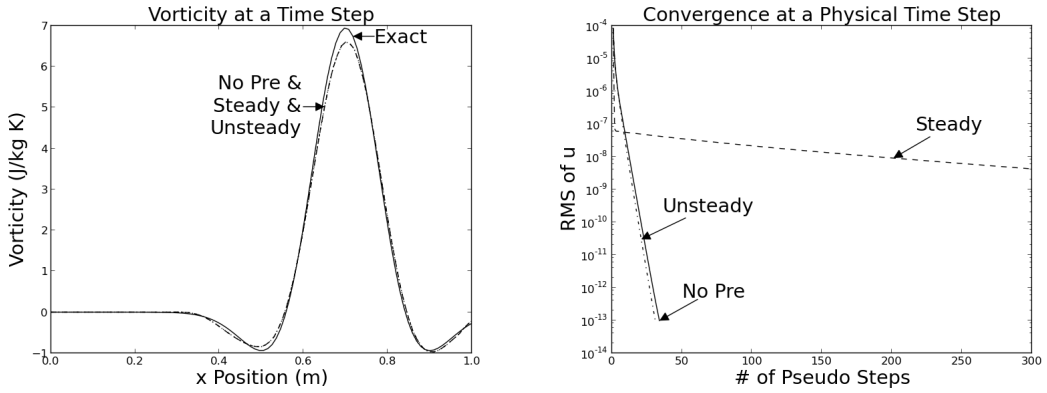
scenario by the drop in velocity field accuracy. The desire to capitalize on the unsteady preconditioners superior convergence while not loosing accuracy is the main motivation behind the implementation of a preconditioned CUSP.

In order to showcase the effectiveness of CUSP, the cases in Figs. 4.4 and 4.5 were re-run using the newly modified CUSP diffusion algorithm. These results are found in Figs. 4.6 and 4.7.

Both Figs. 4.6(a) and 4.7(a) show accuracy improvements in the unsteady preconditioned case when compared to their Roe counter parts. In the low Mach number high Strouhal number limit, the modified CUSP is able to resolve the unsteady issue that Roe portraits. Similar to the airfoil in the previous section, CUSP shows a slight decrease in convergence when compared with Roe. This is most likely due to the LHS and RHS discrepancies.

4.4.3 Oscillating Back Pressure

The previous case highlighted the velocity field. In order to isolate the pressure field, a case was devised to purposefully introduce pressure waves as the driving factor. A simple 1D pipe was modeled using a 2D strand code with uniform cells in the y-direction. Pressure on the outflow was oscillated sinusoidally in time to create pressure waves. This is represented



(a) Vorticity (b) Convergence

Fig. 4.7: Plots of propagating vortex with CUSP diffusion on a square strand domain with a $CFL_c = 1$, $Str = 4076$, $Mach = 0.005$, and 20 points across the vortex.

by

$$p_{x=L} = P_\infty(1 + \epsilon sin(wt)) \tag{4.46}$$

where P_∞ is the free-stream pressure, $p_{x=L}$ is the outlet pressure, w is the frequency that the pressure is driven, t is the physical time, and ϵ is some scale factor.

The selection of ϵ was set so the overall pressure varied less than 1/2 of the dynamic pressure. This prevents the pressure oscillation from reversing the flow direction. It is noteworthy that 1/2 was chosen relatively arbitrarily. ϵ is defined below by

$$\epsilon = 0.5\frac{0.5\rho_\infty u_\infty^2}{P_\infty} \tag{4.47}$$

Another dimensionless parameter is introduced that relates the frequency of oscillation, the length scale of the domain, and the initial flow velocity. The length scale (length of the pipe L) for this problem was set to one. This parameter (Ω) is used to set the frequency of oscillation.

$$\Omega = \frac{wL}{u_0} \tag{4.48}$$

The boundary conditions for the inlet of the pipe warrant discussion. It is necessary

to fix enough of the state to close the problem, while still allowing the outflow pressure to propagate. One method is to use non-reflecting Reimann invariants. Here, stagnation pressure and enthalpy were fixed at the inlet. These allow the pressure to propagate out the inlet, but do exhibit some reflective behavior. At low frequencies, this was found to be negligible. At high frequencies, it was possible to observe flow behavior before the information arrived at the inlet, avoiding the issue altogether.

This case has the advantage of having an exact solution for truly incompressible flows. The solutions show that at a given time step, the pressure is linear in space while the velocity is constant. The exact solution is presented without derivation below.

$$u(t) = -\frac{\epsilon}{\rho_\infty u_\infty (1+\Omega^2)} \left[sin(wt) - \Omega cos(wt) + \Omega e^{-u_\infty t/L} \right] \tag{4.49}$$

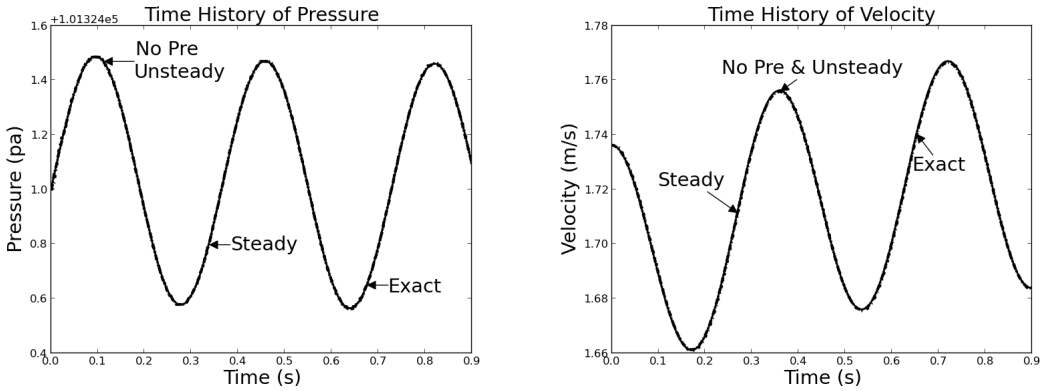
$$p(x,t) = \left[\epsilon sin(wt) + \rho_\infty u_\infty u(t) \right] \frac{x}{L} - \rho_\infty u_\infty u(t) \tag{4.50}$$

When frequency and Mach number are low, the incompressible solution should be a good approximation of the compressible solution. Figs. 4.8 and 4.9 show this situation for both CUSP and Roe. Pressure (at the center of the domain) and velocity are plotted over time. Good agreement is shown between the exact incompressible solution for all methods, again validating the algorithms. Figs. 4.8(c) and 4.9(c) reiterate that the unsteady preconditioner is clearly preferred for convergence characteristics.

To understand the flow behavior as frequency increases, it is necessary to explore how the acoustic wave travel in time and space. The period of the outflow pressure oscillation (T_w) in seconds can be calculated by

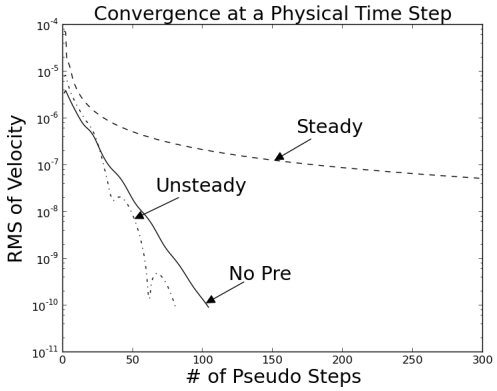
$$T_w = \frac{2\pi}{w} = \frac{2\pi L}{\Omega M_\infty c_\infty} \tag{4.51}$$

These pressure waves travel in space at the acoustic speed (c). Multiplying the period by the speed at which the wave travels, yields the physical period (L_w) of the sinusoidal pressure wave.



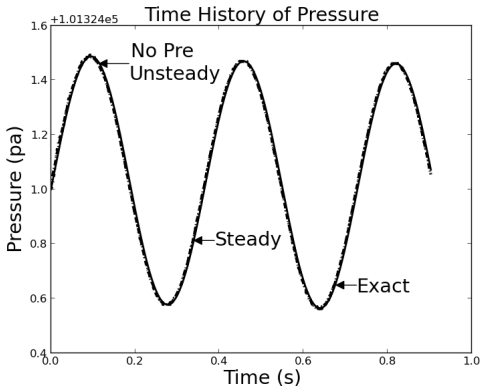
(a) Pressure over time (at center).

(b) Velocity over time.

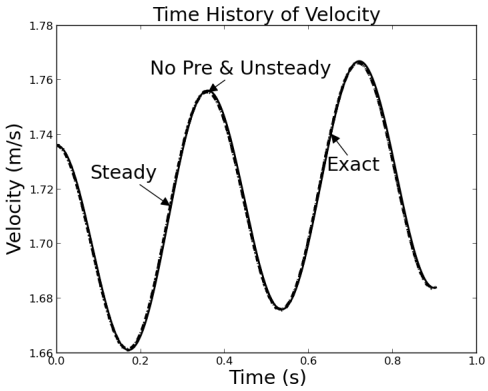


(c) Convergence

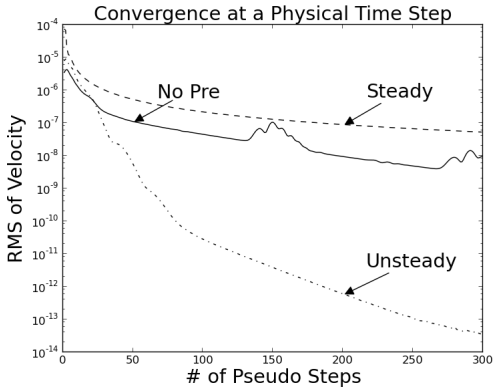
Fig. 4.8: Plots of oscillating back pressure with Roe diffusion using a 2D strand mesh of 128 by 4 with a $CFL_c = 100$, $Str = 81.5$, $Mach = 0.005$, $\Omega = 10$.



(a) Pressure over time (at center).

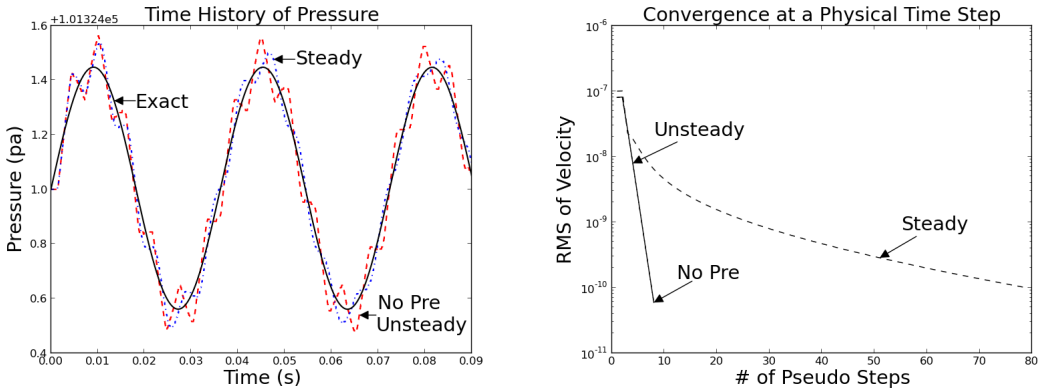


(b) Velocity over time.



(c) Convergence

Fig. 4.9: Plots of oscillating back pressure with CUSP diffusion using a 2D strand mesh of 128 by 4 with a $CFL_c = 100$, $Str = 81.5$, $Mach = 0.005$, $\Omega = 10$.



(a) Pressure over time (at center).

(b) Convergence

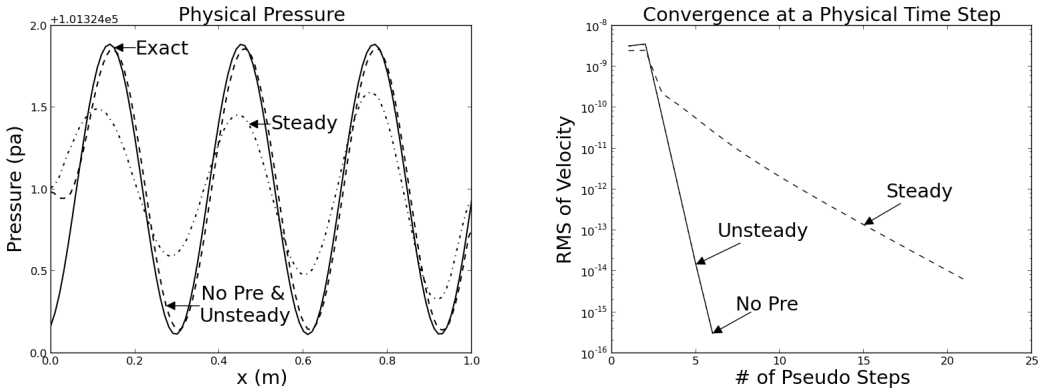
Fig. 4.10: Plots of oscillating back pressure with Roe diffusion using a 2D strand mesh of 128 by 4 with a $CFL_c = 1$, $Str = 8150$, $Mach = 0.005$, $\Omega = 100$.

$$L_w = c_\infty T_w = \frac{2\pi L}{\Omega M_\infty} \tag{4.52}$$

When frequencies are low (i.e. $\Omega = 10$) the physical period of the wave is much larger than the domain of the problem. In these cases, the pressure will appear linear and the incompressible exact solution becomes a good approximation. L_w in Figs. 4.8 and 4.9 was approximately 125 times larger than the domain length. As Ω increases the solution becomes increasingly invalid. Fig. 4.10 shows a case where $\Omega = 100$. Here, the physical period is about 12.5 times larger than the domain scale. The curvature of the physically acoustic waves is now no longer negligible. Fig. 4.10(a) shows the pressures at the center of the domain. It is clear that the the compressible effect of the pressure waves is beginning to manifest in the solution.

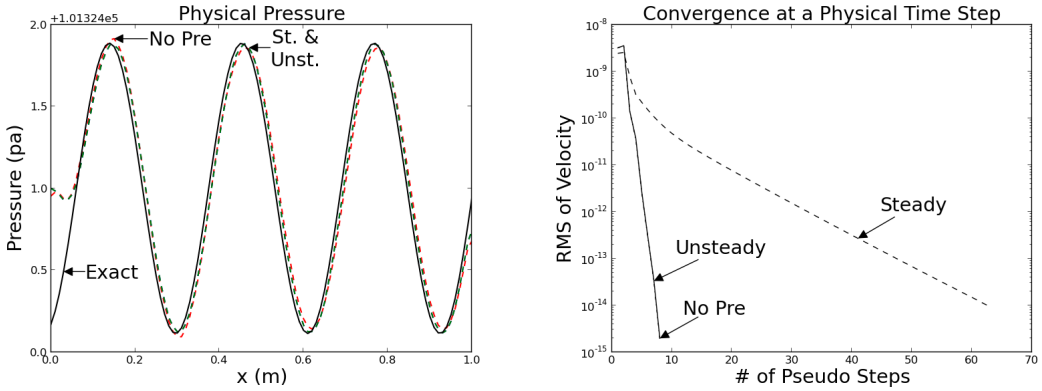
To explore the effects of the acoustic waves over the domain of the pipe, Ω was increased to 4000. This creates a physical period about 1/3 the length of the domain. The expected solution over the domain is a perfectly traced sinusoidal pressure wave that correlates to the outflow oscillations. Figs. 4.11(a) and 4.12(a) show the pressure waves, as they are allowed to travel leftwards through the domain. In order to accurately track the high frequency waves, CFL_C was reduced dramatically to 0.025.

Table 4.1 states that for a pressure driven problem, the steady state preconditioner



(a) Pressure over the domain. (b) Convergence

Fig. 4.11: Plots of oscillating back pressure with Roe diffusion using a 2D strand mesh of 128 by 4 with a $CFL_c = 0.025$, $Str = 326000$, $Mach = 0.005$, $\Omega = 4000$.



(a) Pressure over the domain. (b) Convergence

Fig. 4.12: Plots of oscillating back pressure with CUSP diffusion using a 2D strand mesh of 128 by 4 with a $CFL_c = 0.025$, $Str = 326000$, $Mach = 0.005$, $\Omega = 4000$.

should struggle when Roe dissipation is used. This is consistent with the wave portrayed in Fig. 4.11(a). The addition of the modified CUSP scheme improves the steady state preconditioner's accuracy in this regime.

Even though CUSP is able to improve the accuracy of the physical pressure waves using the steady preconditioner, the steady preconditioner convergence still suffers. This is highlighted in Figs. 4.11(b) and 4.12(b). Unsteady preconditioning in combination with CUSP is capable of accurately capturing both velocity and pressure fields, and has better convergence characteristics.

4.5 Conclusions

Shortcomings of Roe artificial dissipation models with preconditioning were shown in the low Mach number, high Strouhal number limit. A new CUSP scheme was introduced to improve accuracy in this regime for both acoustic and convective unsteady problems. The behavior of the preconditioned CUSP scheme was studied for acoustic dominated problems by setting up flow in a pipe with oscillating back-pressure. For convection-dominated flows, the scheme was evaluated for a propagating inviscid vortex. The new CUSP scheme significantly reduced discretization errors in both regimes. It was also shown that the introduction of the Strouhal number into the preconditioning definition improves efficiency greatly as time steps become small. Combining the new CUSP with unsteady preconditioning is robust and improves accuracy as well as efficiency of compressible CFD algorithms for low speed flows.

Future work will focus on the linearization of the CUSP scheme. This will reduce the stability and convergence issues when having different left and right hand side dissipation implementation. More work will focus on defining the Strouhal number locally, rather than with global scales. The current Strouhal number definition requires some insight to the physics of the problem, where local scales could better "tune" the preconditioning and require less user input. Exploration into the addition of velocity dissipation terms is also planned. Another important are for future work includes investigation of the fluxes on moving meshes. This is a particular interest of strand grids.

Chapter 5

Conclusions

The objectives of this thesis were to:

1. Determine a robust and general method for boundary condition verification.
2. Devise a general method to unify boundary implementation for arbitrary formulations with arbitrary spatial discretization methods.
3. Formulate a preconditioning scheme with improved accuracy for acoustic and convective unsteady problems.

Strategies for accomplishing these objectives were discussed in detail in the main body of this thesis. Here, a summary of the findings for each objective is given.

5.1 Conclusions for Objective 1 - New Boundary Verification Technique

The method of manufactured solutions (MMS), which has been heavily researched as an effective tool for verification of interior discretizations, was extended to apply to boundary conditions. Introducing a new boundary residual that mimics the form of the interior scheme allows a single manufactured solution to verify both the boundary and interior simultaneously. This method applies to both node- and cell-centered paradigms.

The verification method was tested using quasi-1D flow and 2D flow. The proper order of accuracy was obtained using MMS when the boundary conditions were implemented properly. To ensure that the boundaries were implemented properly, cases with exact solutions were tested.

Situations were explored where verification failed, but the exact solution proved that the boundaries in question were implemented correctly. This occurred when the MMS solutions chosen violated the characteristic direction of the boundary. It is necessary to

ccarefully hoose either the location of the boundary or the MMS solution to ensure the characteristic directions are respected.

Verification is a mathematical exercise and shows only the correctness of the boundary algorithms. Verification is not sufficient or intended to ensure that physical problems have the proper boundary conditions imposed. An example of this was explored with Neumann symmetry at an inviscid wall boundary for Ringleb flow. The Neumann symmetry verifies correctly, but is not a proper boundary condition for physics of the problem. This was also evident with inviscid airfoil cases, in which qualitative errors at surface boundaries were apparent.

5.2 Conclusions for Objective 2 - New Generalized Boundary Implementation Framework

A generalized framework was created for boundary implementation that includes Dirichlet, Neumann, extrapolation, and conservation law conditions at boundaries. Unlike node-centered boundary conditions, cell-centered boundary conditions are not easily amenable to using the equations of motion. Consequently, care must be taken with cell-centered schemes to choose conditions that do not conflict with the governing equations of motion. Extrapolation was found to be useful for cell-centered schemes for this reason.

5.3 Conclusions for Objective 3 - New Preconditioning Scheme

Shortcomings of Roe artificial dissipation models with preconditioning were shown in the low Mach number, high Strouhal number limit. A new convective upwind split pressure (CUSP) scheme was introduced to improve accuracy in this regime for both acoustic and convective unsteady problems.The behavior of the preconditioned CUSP scheme was studied for acoustic dominated problems by setting up flow in a pipe with oscillating back-pressure. For convection-dominated flows, the scheme was evaluated for a propagating inviscid vortex. The new CUSP scheme significantly reduced discretization errors in both regimes.

5.4 Suggestions for Future Work

Future work for boundary verification will focus on modifications of the node-centered approach to allow for a water-tight flux formulation. One possible approach is to introduce ghost nodes into the node-centered formulation in a manner similar to the cell-centered formulation. Another important area of investigation includes consistent methods for boundary normal computation. A possible strategy includes locally reconstructing the surface description from surrounding geometry information using a least squares procedure. Derivatives of the reconstructed surface may then be used to estimate normal vectors.

Future work for preconditioning schemes needs extension to viscous flows. Viscous flows add additional scales to the problem, making preconditioning less intutive. Viscous flows are prevelant in many practical problems of interest. Further exploration of needs to occur in three-dimensions and for more complex flow fields. Rotorcraft simulation provide a good scenario where viscous boundaries, complex flows and three-dimensional effects all exist.

References

[1] Hosangadi, A., Sachdev, J., and Sankaran, V., "Improved Flux Formulations for Unsteady Low Mach Number Flows," ICCFD7 Paper ICCFD7-2202, 7th International Conference on Computational Fluid Dynamics, Big Island, Hawaii, July 2012.

[2] Katz, A., Wissink, A., Sankaran, V., Meakin, R., and Chan, W., "Application of strand meshes to complex aerodynamic flow fields," *Journal of Computational Physics*, Vol. 230, 2011, pp. 6512–6530.

[3] Chan, W., Rogers, S., Pandya, S., Kao, D., Buning, P., Meakin, R., Boger, D., and Nash, S., "Chimera Grid Tools User's Manual," [http://people.nas.nasa.gov/ wchan/cgt/doc/man.html], 2010.

[4] Work, D., Tong, O., Workman, R., Katz, A., and Wissink, A., "Strand Grid Solution Procedures for Sharp Corners," AIAA paper 2013-0800, AIAA 51st Aerospace Sciences Meeting, Dallas, TX, January 2013.

[5] Wissink, A., Sitaraman, J., Sankaran, V., Mavriplis, D., and Pullmiam, T., "A multi-code python-based infrastructure for overset CFD with adaptive cartesian grids," AIAA paper 2008-927, AIAA 46st Aerospace Sciences Meeting, Reno, NV, January 2008.

[6] Sitaraman, J., Katz, A., Jayraman, B., Wissink, A., and Sankaran, V., "Evaluation of a multi-solver paradigm for CFD using overset unstructured and structured adaptive Cartesian grids," AIAA paper 2008-660, AIAA 46st Aerospace Sciences Meeting, Reno, NV, January 2008.

[7] Meakin, R., Wissink, A., Chan, W., Pandya, S., and Sitaraman, J., "On Strand Grids for Complex Flows," AIAA paper 2007-3834, AIAA 18th Computational Fluid Dynamics Conference, Miami, FL, June 2007.

[8] Wissink, A., Katz, A., Chan, W., and Meakin, R., "Validation of the Strand Grid Approach," AIAA paper 2009-3792, AIAA 19th Computational Fluid Dynamics Conference, San Antonia, TX, June 2009.

[9] Wissink, A., Katz, A., and Sitaraman, J., "PICASSO: A Meshing Infrastructure for Strand-Cartesian CFD Solvers," Tech. Rep. 2012-2916, 2012.

[10] Roache, P., "Code Verification by the Method of Manufactured Solutions," *ASME Journal*, Vol. 124, March 2002.

[11] Roy, C., Nelson, C., Smith, T., and Ober, C., "Verification of Euler/Navier-Stokes codes using the method of manufactured solutions," *International Journal for Numerical Methods in Fluids*, Vol. 44, 2004, pp. 599–620.

[12] Veluri, S., Roy, C., and Luke, E., "Comprehensive Code Verification for an Unstructured Finite Volume CFD Code," AIAA paper 2010-127, AIAA 48th Aerospace Sciences Meeting, Orlando, FL, January 2010.

[13] Diskin, B., and Thomas, J., "Comparison of Node-Centered and Cell-Centered Unstructured Finite-Volume Discretizations: Inviscid Fluxes," AIAA paper 2010-1079, AIAA 48th Aerospace Sciences Meeting, Orlando, FL, January 2010.

[14] Diskin, B., Thomas, J., Nielsen, E., Nishikawa, H., and White, J., "Comparison of node-centered and cell-centered unstructured finite-volume discretizations. Part I: viscous fluxes," AIAA Paper 2009-0597, 2009.

[15] Diskin, B., and Thomas, J., "Effects of mesh regularity on accuracy of finite-volume schemes," AIAA paper 2012-0609, AIAA 50th Aerospace Sciences Meeting, Nashville, TN, January 2012.

[16] Eriksson, S., and Nordström, J., "Analysis of mesh and boundary effects on the accuracy of node-centered finite volume schemes," AIAA paper 2009-3651, AIAA 19th Computational Fluid Dynamics Conference, San Antonia, TX, June 2009.

[17] Katz, A., and Sankaran, V., "Mesh Quality Effects on the Accuracy of Euler and Navier-Stokes Solutions on Unstructured Meshes," ICCFD6 Paper, 6th International Conference on Computational Fluid Dynamics, St. Petersburg, Russia, July 2010.

[18] Diskin, B., and Thomas, J., "Accuracy Analysis for Mixed-Element Finite-Volume Discretization Schemes," NIA Report 2007-08, National Institute of Aerospace, 2007.

[19] Roy, C., "Review of Code and Solution Verification Procedures for Computational Simulation," *Journal of Computational Physics*, Vol. 205, 2005, pp. 131–156.

[20] Luke, E., Hebert, S., and Thompson, D., "Theoretical and Practical Evaluation of Solver-Specific Mesh Quality," AIAA paper 2008-0934, AIAA 46th Aerospace Sciences Meeting, Reno, NV, January 2008.

[21] Giles, M., "Accuracy of Node-based Solutions on Irregular Meshes," *Lecture Notes in Physics*, Vol. 323, 1989, pp. 273–277.

[22] Choudhary, A., Roy, C., Luke, E., and Veluri, S., "Issues in Verifying Boundary Conditions for 3D Unstructured CFD Codes," AIAA paper 2011-3868, AIAA 20th Computatonal Fluid Dynamics Conference, Honolulu, June 2011.

[23] Rizzi, A., "Numerical Implementation of Solid Body Boundary Conditions for the Euler Equations," *ZAMM*, Vol. S8, 1978, pp. 301–304.

[24] Jameson, A., Baker, T., and Weatherill, N., "Calculation of Inviscid Transonic Flow Over a Complete Aircraft," AIAA paper 83-0103, AIAA 24th Aerospace Sciences Meeting, Reno, NV, January 1986.

[25] Dadone, A., and Grossman, B., "Surface Boundary Conditions for the Numerical Solution of the Euler Equations," *AIAA Journal*, Vol. 32, 1994, pp. 285–293.

[26] Balakrishnan, N., and Fernandez, G., "Wall Boundary Conditions for Inviscid Compressible Flows on Unstructured Meshes," *Int. J. Numer. Meth. Fluids*, Vol. 28, 1998, pp. 1481–1501.

[27] Wang, Z. J., and Sun, Y., "Curvature-Based Wall Boundary Condition for the Euler Equations on Unstructred Grids," AIAA paper 2002-0966, AIAA 40th Aerospace Sciences Meeting, Reno, NV, January 2002.

[28] Allmaras, S. R., "Lagrange Multiplier Implementation of Dirichlet Boundary Conditions in Compressible Navier-Stokes Finite Element Methods," *AIAA Computation Fluid Dynamics Conference*, AIAA 2005-4714.

[29] Sankaran, V., and Merkle, C. L., "Analysis of Preconditioning Methods for the Euler and Navier-Stokes Equations," Tech. Rep., von Karman Institute for Fluid Dynamics, Tullahoma, TN, 1999.

[30] Turkel, E., "Preconditioning Techniques in Computational Fluid Dynamics," Annual Review 1999.31, Annu. Rev. Fluid Mech., 1999.

[31] Caughey, D., and Jameson, A., "Fast preconditioned multigrid solution of the Euler and NavierStokes equations for steady, compressible ows," *International Journal for Numerical Methods in Fluids*, Vol. 43, 2003, pp. 537–553.

[32] Chorin, A., "A Numerical Method for Solving Incompressible Viscous Flow Prolbems," *Journal of Computational Physics*, Vol. 2, August 1967, pp. 12–26.

[33] Weiss, J., and Smith, W., "Preconditioning Applied to Variable and Constant Density Flows," *AIAA Journal*, Vol. 33, No. 11, November 1995, pp. 2050–2057.

[34] Merkle, C., Sullivan, J., Beulow, P., and Sankaran, V., "Computation of Flows with Arbitrary Equations of State," *AIAA Journal*, Vol. 36, No. 4, 1998, pp. 512–521.

[35] Sankaran, V., and Merkle, C., "Efficiency and Accuracy Issues in Contemporary CFD Algorithms," AIAA Paper 2000-2251, AIAA, 2000.

[36] Jameson, A., "Analysis and Design of Numerical Schemes for Gas Dynamics 1 Artificial Diffusion, Upwind Biasing, Limiters and Their Effect on Accuracy and Multigrid Convergence," *International Journal of Computational Fluid Dynamics*, Vol. 4, 1995, pp. 171–218.

[37] Shen, Y., and Zha, G., "Low diffusion E-CUSP scheme with implicit high order WENO scheme for preconditioned NavierStokes equations," *Computers and Fluids*, Vol. 55, 2012, pp. 13–23.

[38] Zha, G., Shen, Y., and Wang, B., "An improved low diffusion E-CUSP upwind scheme," *Computers and Fluids*, Vol. 48, 2011, pp. 214–220.

[39] Gustafson, K., *Introduction to Partial Diferential Equations and Hilbert Space Methods*, Dover, Mineola, 3rd ed., 1999.

[40] Kundu, P. K., and Cohen, I. M., *Fluid Mechanics*, Elsevier, Oxford, 4th ed., 2008.

[41] Jameson, A., "Analysis and Design of Numerical Schemes for Gas Dynamics 2 Artificial Diffusion and Discrete Shock Structure," *International Journal of Computational Fluid Dynamics*, Vol. 5, 1995, pp. 1–38.

[42] Babuska, I., "Error-Bounds for the Finite Element Method," *Numer. Math.*, Vol. 16, 1971, pp. 322–333.

[43] Katz, A., and Sankaran, V., "Mesh Quality Effects on the Accuracy of Euler and Navier-Stokes Solutions on Unstructured Meshes," *Journal of Computational Physics*, Vol. 230, No. 20, 2011, pp. 7670–7686.

[44] Tota, P., and Wang, Z. J., "Meshfree Euler Solver Using Local Radial Basis Functions for Inviscid Compressible Flows," AIAA paper 2007-4581, AIAA 18th Computational Fluid Dynamics Conference, 2007.

[45] Kirshman, D., and Liu, F., "Gridless Boundary Condition Treatment for a Non-Body-Conforming Mesh," AIAA paper 2002-3285, AIAA 32nd Fluid Dynamics Conference, St. Louis, MO, June 2002.

[46] Koh, E., and Tsai, H., "Euler Solution Using Cartesian Grid with Least Squares Technique," AIAA paper 2003-1120, AIAA 41st Aerospace Sciences Meeting, Reno, NV, January 2003.

[47] Shapiro, A. H., *The Dynamics and Thermodynamics of Compressible Fluid Flow*, Vol. 2, The Ronald Press Company, New York, 1954.

[48] Barth, T. J., "Aspects of Unstructured Grids and Finite-Volume Solvers for the Euler and Navier-Stokes Equations," Tech. Rep., von Karman Institute for Fluid Dynamics, Moffett Field, CA, March 1994.

[49] Luo, H., Xiao, H., Nourgaliev, R., and Cai, C., "A Comparative Study of Different Reconstruction Schemes for a Reconstructed Discontinuous Galerkin Method on Arbitrary Grids," AIAA paper 2011-3839, AIAA 20th Computational Fluid Dynamics Conference, Honolulu, HI, June 2011.

[50] Andren, J., Gao, H., Yano, M., Darmofal, D., Ollivier-Gooch, C., and Wang, Z., "A Comparison of Higher-Order Methods on a Set of Canonical Aerodynamics Applications," AIAA paper 2011-3230, AIAA 20th Computational Fluid Dynamics Conference, Honolulu, June 2011.

[51] Potsdam, M., Sankaran, V., and Pandya, S., "Unsteady Low Mach Preconditioning with Application to Rotorcraft Flows," AIAA paper 2007-4473, AIAA Computational Fluid Dynamics Conference, Orlando, FL, July 2007.

[52] Liou, M., "A Sequel to AUSM: AUSM$^+$," *Journal of Computational Physics*, Vol. 129, 1996, pp. 364–382.

[53] Liou, M., "A sequel to AUSM, Part II: AUSM$^+$-up for all speeds," *Journal of Computational Physics*, Vol. 214, 2006, pp. 137–170.

[54] Katz, A., "Course Notes for MAE 6440 - Advanced Computational Fluid Dynamics," Tech. Rep., Utah State University, Logan, UT, 2012.