An Adaptive Higher-Order Unstructured Finite Volume Solver for Turbulent Compressible Flows

by

Alireza Jalali

B.Sc., Mechanical Engineering, University of Tehran, 2010
M.ASc., Mechanical Engineering, The University of British Columbia, 2012

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate and Postdoctoral Studies

(Mechanical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

January 2017

Abstract

The design of aircraft depends increasingly on the use of Computational Fluid Dynamics (CFD) in which numerical methods are employed to obtain approximate solutions for fluid flows. One route to improve the numerical accuracy of CFD simulations is higher-order discretization methods. Moreover, finite volume discretizations are the method of choice in commercial CFD solvers and also in computational aerodynamics because of intrinsic conservative and shock-capturing properties. Considering that nearly all practical flows with aerodynamic applications are classified as turbulent, we develop a higher-order finite volume solver for the Reynolds Averaged Navier-Stokes (RANS) solution of turbulent compressible flows on unstructured meshes.

Higher-order flow solvers must account for boundary curvature. Since turbulent flow simulations require anisotropic cells in shear layers, we use an elasticity analogy to project the boundary curvature into the interior faces and prevent faces from intersecting near curved boundaries. Furthermore, we improve the accuracy of solution reconstruction and output quantities on highly anisotropic cells with curvature using a local curvilinear coordinate system. A robust turbulence model for higher-order discretizations is fully coupled to the system of RANS equations and an efficient solution strategy is adopted for the convergence to the steady-state solution. We present our higher-order results for simple and complicated configurations in two dimensions. These results are verified by comparison against well-established numerical and experimental values in the literature. Our results show the advantages of higher-order methods in obtaining a more accurate solution with fewer degrees of freedom and also fast and efficient convergence to the steady-state solutions.

Moreover, we propose an hp-adaptation algorithm for the unstructured finite volume solver based on residual-based and adjoint-based error indicators. In this approach, we enhance the local accuracy of the discretization via h-refinement or p-enrichment based on the smoothness of the solution. Mesh refinement is carried out by local cell division and introducing non-conforming interfaces in the mesh while

order enrichment is obtained by local increase of the polynomial order in the reconstruction process. Our results show that this strategy leads to accuracy and efficiency improvements for several types of compressible flow problems.

Preface

The research ideas and methods explored and presented in this thesis are the outcome of the research done by Alireza Jalali during his PhD program in the department of Mechanical Engineering. The implementation of methods, data analysis, and manuscript preparation were done by the author, Alireza Jalali, under the supervision of Professor Carl Ollivier-Gooch throughout the process. The following are the journal articles published or in progress based on the contents of this thesis:

- A. Jalali, M. Sharbatdar, C. Olliovier-Gooch. "Accuracy analysis of unstructured finite volume discretization schemes for diffusive fluxes", *Computers & Fluids*, 101 (2014) 220-232.

  This paper was co-authored with another PhD candidate, M. Sharbatdar, under the supervision of C. Ollivier-Gooch. Some results of this paper are available in Chapter 4 of this thesis. The code development, data analysis and manuscript preparation were done by A. Jalali and an eigen analysis tool developed by M. Sharbatdar for her PhD thesis was used to interpret data.

- A. Jalali, C. Ollivier-Gooch. "Higher-order Unstructured Finite Volume RANS Solution of Turbulent Compressible Flows", *Computers & Fluids,* 143 (2017) 32-47.

  The results of this paper are also available in Chapters 3 and 4 of this thesis.

- A. Jalali, C. Ollivier-Gooch. "An *hp*-adaptive Unstructured Finite Volume Solver for Compressible Flows", Under review, 2016.

  A version of chapter 5 of this thesis has been submitted for publication.

- M. Sharbatdar, A. Jalali, C. Ollivier-Gooch, "Smoothed Truncation Error in

Functional Error Estimation and Correction using Adjoint Methods in an Unstructured Finite Volume Solver", *Computers & Fluids,* 140 (2016) 406-421.

This paper is an excerpt from M. Sharbatdar's PhD thesis about functional correction using adjoint methods. The substantial part of this work including the code development for continuous adjoint and solution interpolation techniques, and also data analysis and manuscript preparation were done by M. Sharbatdar. A. Jalali provided the discrete adjoint solver which was already developed for *hp*-adaptation method described in Chapter 5 of this thesis.

Table of Contents

Appendices

List of Tables

List of Figures

Nomenclature

Roman Symbols

A	Area
a	Wave speed, Sound speed
AR	Aspect ratio
c	Chord length
C_D	Drag coefficient
C_f	friction coefficient
C_L	Lift coefficient
C_p	Pressure coefficient
c_p	Specific heat at constant pressure
D	Destruction, Dissipation matrix
d	Linear elasticity coefficients, distance function
E	Young's modulus
E_t	Total energy
F	Flux vector
h	Mesh size
h_t	Total enthalpy
I	Identity matrix

J	Jacobian of transformation
K	Stiffness matrix
Ma	Mach number
P	Pressure, Production
p	Order of polynomial
Pr	Prandtl number
$Pr_{_T}$	Turbulent Prandtl number
R	Residual operator
Re	Reynolds number
S	Source term, Vorticity strength, Smoothness indicator
s	Control surface length, Stretching factor
T	Temperature
t	Time, Reconstruction weight exponent, Curve parameters
t, n	tangential-normal coordinates
U	Vector of conserved variables, Left singular vector
u	x-velocity
V	Right singular vector
v	y-velocity
W	Vector of primitive variables
w	Reconstruction weight
x, y	Cartesian coordinates
x', y'	Principal coordinates

Z	Adjoint solution
$\mathcal{J}$	Functional
$\mathcal{K}$	Smoothness indicator constant
$\hat{n}$	Unit normal vector
$\vec{r}$	Reference location connecting vector
$\vec{V}$	Velocity vector

Greek Symbols

α	Angle of attack
$\dot{\gamma}$	Shear rate tensor
ϵ	Control volume error contribution
η_l	Linear solver tolerance
γ	Ratio of specific heats
κ	Condition number, von Karman constant
Λ	Diagonalized matrix of eigenvalues
λ	Eigenvalues of flux Jacobian
μ	Dynamic viscosity
μ_T	Turbulent viscosity
ν	Poisson's ratio, Kinematic viscosity
ω	Under-relaxation factor
ϕ	Arbitrary primitive variable, Finite element basis functions
ρ	Density
Σ	Diagonal matrix of singular values reciprocal

σ	Stress tensor components, Spalart-Allmaras diffusion constant
τ	Shear stress tensor
θ	Azimuthal location
$\tilde{\nu}$	Turbulence model working variable
$\vec{\delta}$	Displacement vector
ξ, η	Reference element coordinates

Acknowledgements

First and foremost, I would like to acknowledge my research supervisor, Dr. Carl Ollivier-Gooch, whose constant support, patience and guidance has been invaluable throughout the course of this research. I enjoyed a lot during the years I worked with you as a graduate student and I learned a lot not just in CFD.

I would also like to thank my supervisory committee members, Kendal Bushe, Brian Wetton and Chen Greif. Your comments and criticisms in the annual committee meetings and the time and efforts for reading the manuscript have surely led to a better presentation of this thesis.

Also, I am very grateful for the financial support of the University of British Columbia through the Four Year Doctoral Fellowship and also the Discovery Grant provided by the Canadian Natural Sciences and Engineering Research Council. Moreover, I would like to thank my dear colleagues at the Advanced Numerical Simulation Lab: Reza, Gary, Shayan and Logan. You guys were always available to bounce ideas around and generously offered me your opinion whenever needed.

On a personal side, I would also like to express my sincere appreciation to my wonderful family, especially my dedicated mother, Parvin, and my great father, Mohammadhassan, for their love, encouragement and dedication all through my life.

Finally, my deepest appreciation goes for my lovely spouse, Mahkame, for her endless love and support in my life. You have truly been my best companion throughout this long journey from the very beginning to this end.

Chapter 1

Introduction

The behavior of fluid flows is governed by a set of partial differential equations (PDEs), which relate flow field variables to their derivatives in time and space. The solution of these equations enables us to predict the dynamics of the fluid by knowing the values of flow quantities, such as pressure, temperature and velocity at desired locations and times. These PDEs, which are typically obtained by some complicated physical models, have an analytic solution only in a limited number of cases derived by non-trivial mathematical techniques [1]. As a result, we need to use alternative options such as experimental or numerical methods in most of the cases with complex geometries and/or physical models.

Experimental fluid dynamics, which dates back to ancient times, has been used over the centuries to give insight into flow patterns and measure field quantities. In recent years, a considerable amount of efforts have been put to improve the accuracy and efficiency of experimental techniques. However, these methods still suffer from many restrictions related to their accuracy and reproducibility. There are some errors that stem from the disturbances induced by inserting probes into the flow field and reduce the fidelity of experimental data. The measurements are often restricted to one quantity at sampled points every time and thus the measurement of the whole flow field requires a large amount of time and resources. Nevertheless, there are some cases where once a model is built, a lot of data can be measured. The experiments provide prediction only for a laboratory-scale model of simple prototypes whereas many real life cases (e.g., such as reentry of space vehicles) cannot be investigated in laboratories since similarity conditions are not achieved.

With the recent progress in computational resources and numerical algorithms, Computational Fluid Dynamics (CFD) has emerged as a reliable tool to examine complex flows and realistic operating conditions. In this method, which has shown a remarkable ability in quantitative prediction of flow phenomena with high resolution in time and space, numerical techniques are employed to provide the solution to the

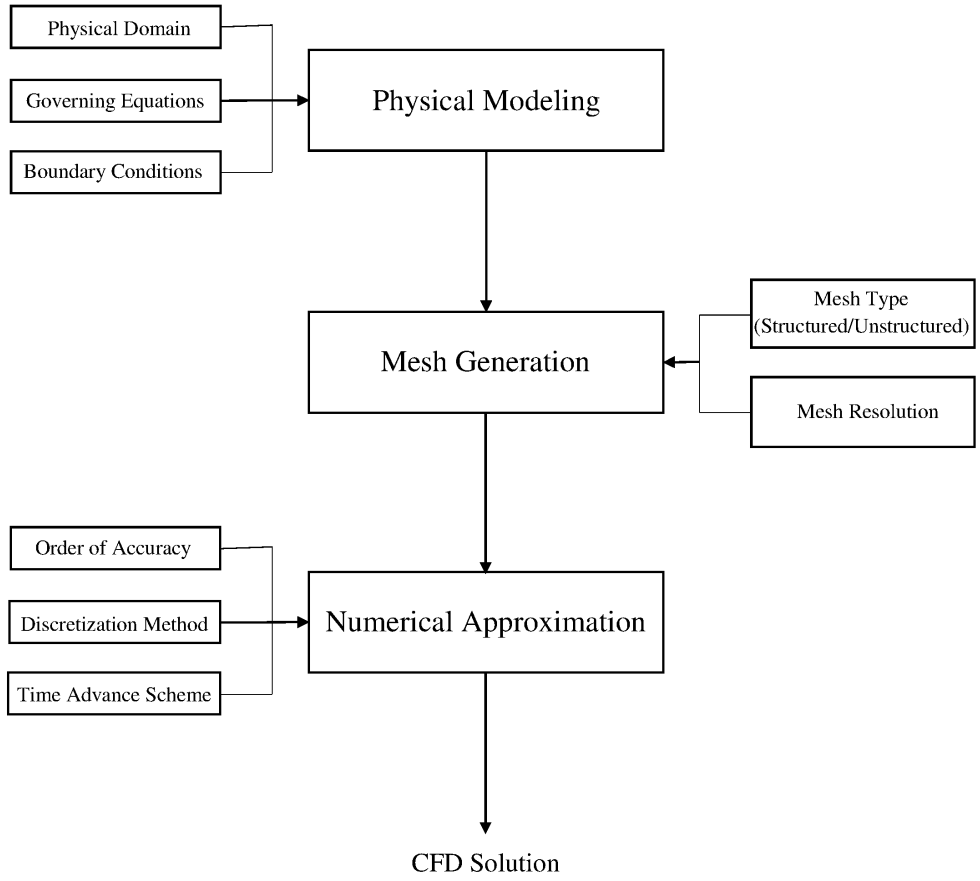


Figure 1.1: General steps in a CFD simulation

PDEs describing flow behavior. CFD is also popular due to its lower cost compared to experiments and its great advantage in computing many flow variables which are not accessible in an experiment. The ultimate goal of CFD is to find an accurate and reliable solution to fluid dynamic problems in a short amount of time and with minimum computational resources.

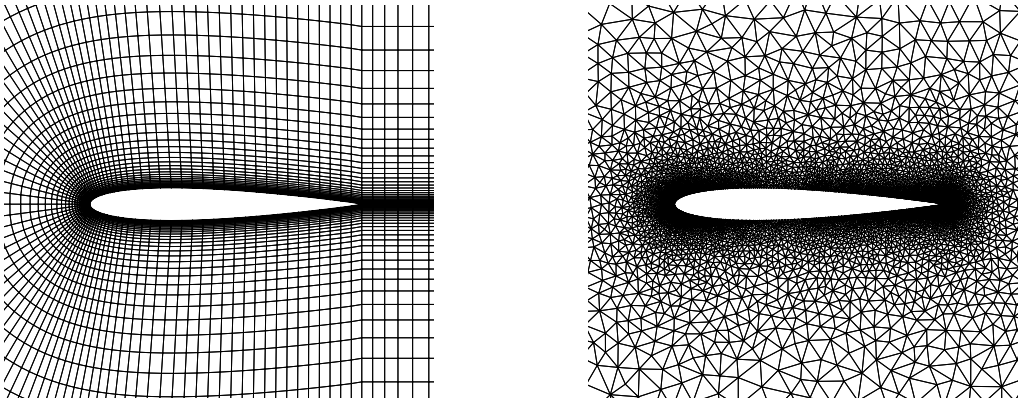
The numerical simulation of flow fields by means of CFD requires three essential elements [2]: physical modeling of the flow; domain decomposition, which is referred to as mesh generation; and the numerical approximation of the governing equations arising from the physical modeling using a robust, efficient and accurate solver. These steps are illustrated in Figure 1.1.

The first element includes the mathematical modeling of the flow physics. In

this phase of the process, the system of governing equations is specified along with the geometry of the domain to be decomposed later and the appropriate physical boundary conditions. This process is fairly straightforward for simple flow scenarios (e.g., subsonic inviscid or viscous laminar flows) with trivial geometries. However, it exhibits complications for more complex situations such as those with multiphase, turbulence and hydromagnetic field effects for which an exact mathematical description of the physical problem is impossible or infeasible, and so physical modeling is required. Therefore, this modeling process leads to a known source of errors in CFD simulations, referred to as physical modeling error. For instance, the turbulence models used to predict the behavior of turbulent flows are tuned based on only a limited number of physical cases and thus fail to provide a general and accurate estimation of turbulence properties in many other applications. Nevertheless, they seem to be the only feasible approach [3, 4] since the Direct Numerical Simulation (DNS) of turbulent flows will not be practical for the next couple of decades due to computing limitations.

The next step is to create a mesh on which the solution is approximated. In this process, the physical domain is tessellated with shapes that are recognizable by the solver. Considering that the governing equations are discretized on the mesh, an appropriate domain decomposition is essential to compute an accurate solution. As a result, high quality mesh generation is considered as one of the most crucial elements in CFD.

There are basically two different types of meshes: structured and unstructured. In a structured mesh, all cells and vertices have the same topology and their connectivity is specified by their indices. For example, cell (i, j) is always topologically to the right of cell $(i-1, j)$. Such implicit information leads to faster processing by the solver but restricts the topology of the mesh which in turn makes the automation of grid generation challenging for complex geometries. On the other hand, the vertices in unstructured meshes are topologically different and thus their connectivity must be declared. These meshes are typically formed as a collection of polygons in 2D and polyhedrons in 3D. Despite the fact that the processing time of the solver is longer for unstructured meshes, they exhibit a higher flexibility and thus are better candidates for arbitrary geometries. The use of unstructured meshes is becoming more popular in modern CFD applications as they promise to be more capable and successful for complex aerodynamic problems [5]. Figure 1.2 illustrates the two types of the meshes



(a) Structured (b) Unstructured

Figure 1.2: Example of mesh types over a 2D airfoil

around a 2D airfoil.

After physical modeling and mesh generation are complete, the infinitely accurate governing PDEs must be turned into a finite approximation of flow quantities over the mesh. This process is known as discretization whose error can be expressed as $\mathcal{O}\left(h^{p+1}\right)$ for problems with smooth solution, where h is a characteristic length for the mesh and $p+1$ is the asymptotic order of accuracy. Clearly, the accuracy of a numerical solution can be increased via h-refinement (finer mesh) and/or p-enrichment (higher-order accurate discretization). In the CFD community, those discretization schemes where $p > 1$ (higher than second-order accurate), are typically known as higher-order schemes. The discretization is typically performed by one of the following methods: finite difference, finite element and finite volume.

The finite difference method is the most traditional method and has historical importance. In this method, the point wise quantities are approximated using a difference relation obtained by the Taylor series expansion of the solution on a structured mesh. Finite difference methods can be easily programmed and their extension to higher-order is straightforward; however, the corresponding solution does not necessarily conserve mass, momentum and energy. Also, the method is impractical for unstructured meshes. Due to recent interest in simulating complex geometries, which is most easily accomplished by the use of unstructured meshes, finite difference is less

frequently used in CFD codes nowadays.

The finite element method originated from the structural analysis of solids but is also applicable to fluid flows. In this method, the solution is represented by local basis functions over the elements of the mesh. The degree of the polynomial basis function is chosen based upon the desired order of solution accuracy. While the solution represented by the polynomial does not completely satisfy the governing equations, the polynomial coefficients are chosen so as to minimize the residuals weighted by a test function. The extension of the finite element method to higher-order solutions can be easily carried out using a basis function with higher degree. Also, it can be used for elements with arbitrary shape and thus is well suited for unstructured meshes. Although the conservation of mass, momentum and energy may be achieved in the finite element formulation, it is not trivial, particularly in flows with discontinuities.

The finite volume method has been designed based on the control volume analysis used for thermodynamics or fluid dynamics systems. The finite volume method discretizes the governing equations in conservative form over cells in the mesh to yield the conservation of mass, momentum and energy across the boundaries of control volumes. The conservation property enhances the capability of the finite volume method in capturing discontinuous phenomena such as shock waves. In this technique, control volume averages are used to find the reconstructed piecewise polynomial over each finite volume. The volume boundary fluxes are computed based on the reconstructed polynomial, and the flux integrals are used to update the control volume averages. The finite volume technique, which shows remarkable flexibility for complex geometries and unstructured meshes, can be extended to higher-order accuracy [6, 7], although with more difficulty compared to the finite difference or finite element methods.

In this thesis, we focus on finite volume methods for compressible flows encountered in computational aerodynamics. In particular, we are interested in higher-order methods on unstructured meshes for turbulent flows in 2D. In what follows, a brief review of turbulence modeling approaches employed in CFD simulations is given. Moreover, the state of the art in higher-order CFD methods used in computational aerodynamics is reviewed. This chapter continues with a clear description of objectives set for this thesis and ends with the outline of the following chapters.

1.1 Turbulence Modeling in CFD

Turbulence is frequently observed in nearly all areas of fluid mechanics ranging from turbomachinery and combustion engines to the design of cars and aircraft. A great deal of effort has been put into experimental and theoretical studies to improve the understanding of the turbulent motions. However, a universal turbulence theory which is able to predict the features of turbulent flows in different applications is not yet available.

Numerical simulation of turbulent flows has practical importance in science and engineering. In the context of CFD, there exist three approaches to simulate turbulent flows. These approaches are the Direct Numerical Simulation (DNS), the Large Eddy Simulation (LES), and the Reynolds-Averaged Navier-Stokes equations (RANS).

In the DNS approach, the deterministic Navier-Stokes equations are solved without the incorporation of any turbulence model. As a result, the whole range of spatial and temporal scales of the turbulent motions must be resolved to obtain an accurate prediction of the flow field. The smallest turbulence scales, which are referred to as the Kolmogorov micro-scales, must be captured by the computational length scale (mesh size) and time step. The calculations show that the smallest spatial and temporal scales scale with the Reynolds number of turbulent flows as $Re^{-3/4}$ and $Re^{-1/2}$, respectively [8]. This induces huge memory and computational power requirements for realistic flow problems at relatively high Reynolds numbers ($Re > 10^6$). As an example, the computation of the turbulent flow around an aircraft in one second of the flight time would require several thousand years and 10^{16} grid points using a supercomputer with 10^{12} Flops [9]. Therefore, the application of the DNS approach has been limited to simple geometries and flows with low Reynolds number [10, 11]. In the DNS simulations, higher-order approximations are typically employed to overcome the numerical dissipation of the lower-order methods [12, 13]. In addition, explicit time integration is mostly used due to the large memory requirements and the need to resolve small time scales.

In LES, the computational cost of DNS is reduced by ignoring the smallest turbulence scales. The main idea behind LES is to extract the large scale energy containing components by the convolution of the dependent variables of the Navier-Stokes equations with a predefined low-pass filter, which can be effectively viewed as a spatial and temporal averaging. Therefore, the information about the small scales, which is

removed from the numerical solution, must be modeled using a subgrid scale model [8]. In other words, the numerical solution accounts for the geometry dependent large scale eddies of the flow, while the universal smaller scale eddies are captured implicitly. This separation of scales is possible away from walls where the turbulence is not in equilibrium but cannot be applied close to solid boundaries where turbulence manifests in the form of coherent structures [14]. Therefore, an accurate modeling of wall effects is a dominant challenge in LES. Even though LES is computationally less expensive compared to DNS, it still needs a considerable amount of resources for large scale engineering problems with high Reynolds number. In recent years and with significant progress in computational power, LES is reaching a level of maturity and is becoming a reliable tool for engineering and industrial computations [15]. The overall success of LES is dependent on the accuracy of discretization method and the performance of the simulation in a time- and cost-effective manner [8].

The most common approach for the modeling of turbulent flows with engineering applications is RANS. In this approach, which is extensively used for steady flow problems, the physical quantities are decomposed via the Reynolds decomposition into time-averaged and fluctuating components. The substitution of the decomposed quantities into the Navier-Stokes equations yields the governing equations for the mean flow variables. This process leads to the appearance of non-linear Reynolds stresses which are dependent on the fluctuating velocity components. Additional modeling is required for such terms to provide the closure for the system of RANS equations.

Most of the turbulence models used in engineering practice employ the Boussinesq assumption in which the Reynolds stresses in the Reynolds-averaged momentum and energy equations are assumed to be proportional to the mean strain tensor. In this approximation, the constant of proportionality is isotropic and is called the turbulent viscosity. The turbulent viscosity can be related to the mean flow quantities, as in algebraic (zero-equation) models [16, 17] or be obtained by one or more auxiliary field equations for the turbulent kinetic energy and time scales [18, 19, 20].

On the other hand, the Reynolds stress transport models, which do not use the Boussinesq assumption, are considered as the highest level of available closure for the system of RANS equations [21]. These models are superior to the turbulent viscosity models because they remove the assumption that the Reynolds stresses respond immediately to changes in the mean strain rate. Moreover, they take into

account the anisotropy of turbulence. However, this comes with the price of solving additional PDEs for each component of the Reynolds stress tensor and thus they are more expensive compared to the turbulent viscosity models in terms of CPU time and memory usage. In addition, numerical stability problems arise due to the absence of the turbulent viscosity and strong coupling [22].

Most of the numerical investigations in the field of computational aerodynamics have been based on steady low-order RANS simulations during the last few decades. This is due to the fact that the second-order RANS simulations became robust and affordable on small CPU clusters. Most commercial CFD solvers use RANS turbulence models for different applications. However, it is well understood that the RANS models are unable to provide good predictions for flows with separation and/or vortex dominated flows [23]. Considering that such problems are encountered in aerodynamic problems (e.g., landing and take-off with high lift configurations), more powerful modeling tools are required. With the recent advances in hardware technology, higher level CFD approaches such as higher-order methods [24], hybrid RANS/LES [25, 26] and LES [27] are being examined in the field of computational aerodynamics.

1.2 Higher-order Methods

Higher-order (higher than second-order) discretizations of the fluid dynamic equations have recently received substantial attention due to their potential advantages in obtaining more accurate solutions with lower cost. With higher-order accurate methods, the computational cost increases on a fixed mesh; however, a coarser mesh can be used to save time and memory and increase accuracy as well. Several families of higher-order methods have been developed over the years in the CFD community to solve different types of problems. In this section, a short review of previous work using higher-order methods for the solution of compressible flows is given.

Many higher-order methods used in computational aerodynamics have been successfully developed for computations on structured meshes [28, 29]. Although higher-order unstructured methods are advantageous for complex geometries, it has been shown that the finite difference method on structured meshes are superior in terms of computational cost and efficiency for boundary layer simulations [30]. Considering that structured single-block generation of high quality meshes is challenging for realistic problems, higher-order multi-block finite difference solvers have become attractive

in recent years [31]. In these methods, higher-order finite difference approximations are employed on each block and some type of interface conditions is applied to transfer information between adjacent blocks [32, 33, 34]. Some other higher-order schemes such as the Essentially Non-Oscillatory finite volume methods (ENO/WENO) have been designed for the solution of hyperbolic equations on multi-dimensional structured meshes. In these methods, which rely on the conservation of flow quantities in each control volume, the one-dimensional ENO/WENO reconstruction operators are applied along the coordinate lines to provide a higher-order approximation of numerical fluxes across the interfaces. The reconstruction operators use adaptive stencils so that higher-order accuracy and non-oscillatory properties near discontinuities are obtained [35, 36]. For structured meshes, it has been shown that, for practical levels of accuracy, using a higher-order accurate method can be more efficient both in terms of solution time and memory usage [37, 6].

Research in high-order unstructured solvers is motivated by the accuracy and efficiency advantages demonstrated in the application of these schemes on structured meshes with the flexibility of unstructured meshes in adaptivity and for complex geometries. As described, the finite element approach can be easily implemented on the unstructured meshes and offers a simple path to higher-order discretizations. However, the standard, continuous Galerkin method is inappropriate for use on convection-dominated problems due to its stability issues and shortcomings in shock capturing. In recent years, a class of the finite element methods known as the Discontinuous Galerkin (DG) schemes have become very popular particularly in the context of compressible flow simulations. In these methods, the basis functions are chosen so that jumps in the solution are allowed across element interfaces. The discontinuous approximation space provides stability for convection problems. The work by Cockburn *et al.* [38, 39], where DG methods were employed to solve non-linear time dependent hyperbolic equations, led to the rapid development of higher-order DG solvers. The compactness and flexibility of DG methods for higher-order discretizations on unstructured meshes make them attractive even though the number of degrees of freedom grows rapidly with polynomial degree, making these schemes very expensive per element. These methods have been extensively used for the higher-order solution of compressible Euler [40, 41] and Navier-Stokes [42, 43] equations. Also, recent studies have demonstrated the ability of DG in computing RANS turbulent flows with shock waves [44, 45].

Considering that the finite volume methods are traditionally the most popular method in computational aerodynamics because of intrinsic conservative and shock-capturing properties, the use of higher-order finite volume on unstructured meshes is quite attractive. The first step in the extension of finite volume methods to higher-order discretizations on unstructured meshes dates back to the quadratic reconstruction operator developed by Barth and Frederickson [46] for the Euler Equations. Third- and fourth-order discretizations of these methods have been successfully applied to the Euler and Navier-Stokes equations [47, 7, 48, 49]. These methods have shown their ability for efficient computations of flows with/without discontinuities with promising accuracy on irregular meshes. From the mathematical point of view, finite volume schemes can be written as a Petrov-Galerkin variant of the discontinuous Galerkin method regardless of the type of reconstruction operator [50]. The next major step towards the general adoption of higher-order finite volume methods for use in CFD solvers is turbulence modeling which is the subject of this thesis.

1.3 Objectives

The higher-order unstructured finite volume solver previously developed at the Advanced Numerical Simulation Lab at UBC is capable of solving inviscid and viscous laminar compressible flow problems on unstructured meshes. Considering that common types of flow in aerodynamic applications are turbulent, the accuracy that can be achieved in a flow simulation strongly depends on the prediction of the characteristics of the turbulent flow field. Important physical phenomena, such as boundary layer separation and shock-boundary layer interaction, can be predicted with highly accurate simulation of turbulent flows. The first objective of this thesis is to extend the capabilities of the current flow solver so that it can compute higher-order RANS solutions on unstructured meshes over two-dimensional aerodynamic configurations.

Having accomplished the first objective, a higher-order unstructured finite volume solver for a wide range of aerodynamic problems from inviscid subsonic to high-Reynolds viscous turbulent will be in hand. However, higher-order accuracy is advantageous only in smooth regions of the solution where there is no discontinuity in the solution. In aerodynamic applications, several sources of discontinuities such as shocks and contact discontinuities exist which negate the benefit of using higher-order discretizations. Therefore, we can enhance the efficiency (and also robustness)

of the solver using a so-called *hp*-adaptation technique. *hp*-adaptation combines grid refinement (i.e., *h*-refinement) and order enrichment (i.e., *p*-enrichment) into a single adaptive method where the level and location of refinement are determined based on an *a posteriori* error estimation. The second objective of this thesis is to develop an *hp*-adaptive algorithm for our compressible flow solver and apply that on different classes of problems which are of particular interest in computational aerodynamics.

1.4 Outline

This thesis is organized as follows.

Chapter 2 provides an overview of the finite volume solver that has been used as the existing infrastructure for the fulfillment of our objectives. Several aspects of the solver including solution reconstruction, flux functions, integration rules and time advancement schemes are discussed in detail. Also, the elements required for extension to RANS simulation of compressible aerodynamic flows, which is our first objective, are introduced briefly. The details of each of these elements is the subject of the next two chapters.

Chapter 3 explains the treatment of anisotropic meshes considering that cells with very high aspect ratio are necessary for turbulent flow simulations. These treatments include curving the interior faces of a linear mesh for higher-order computations, cubic mesh pre-processing algorithms and accurate and well-conditioned solution reconstruction on highly anisotropic meshes.

In addition, we require a robust and accurate RANS turbulence model for higher-order computations and an effective solution strategy for obtaining the steady-state solution of the flow field. These elements are described in Chapter 4 along with our numerical results for a variety of 2D turbulent aerodynamic problems.

Chapter 5 is devoted to the *hp*-adaptive algorithm proposed to accomplish our second objective. This chapter describes in detail the error estimation methods used to drive adaptation, and the mesh refinement and order enrichment techniques employed. Also, the results of our adaptive solver, including its efficiency and accuracy advantages, are presented for different flow conditions ranging from inviscid to turbulent flows.

Finally, Chapter 6 summarizes the research in this thesis, provides conclusions based on the results, and proposes some future work.

Chapter 2

Background

This chapter provides background material relevant for the current research. This work is based on the higher-order unstructured finite volume solver previously developed at the Advanced Numerical Simulation Lab (ANSLib) for inviscid and viscous laminar flows. The background includes an overview of relevant aspects of this CFD solver and those elements that are necessary for its extension to turbulent flows.

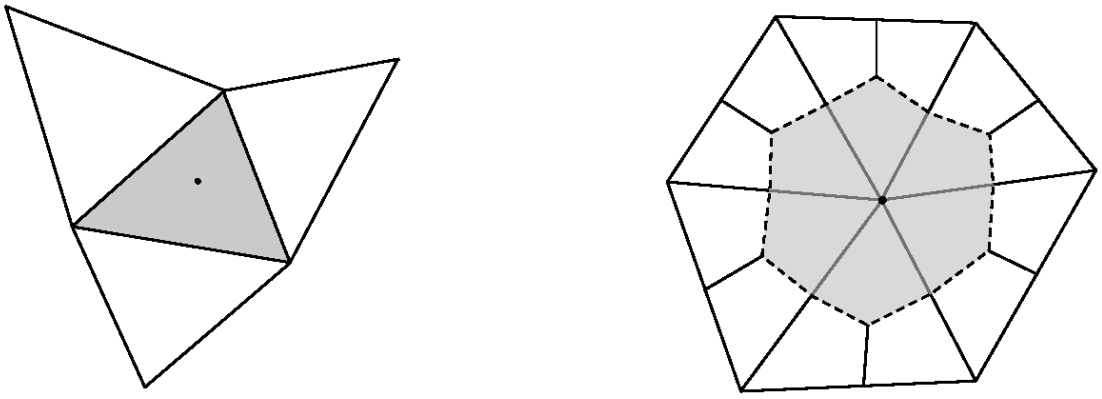
Finite volume methods use the fully conservative control volume form of the governing equations that describe fluid flows. These methods compute control volume averages of the unknowns over a set of finite volumes that cover the domain. ANSLib is capable of supporting two types of control volumes that are widely used to tessellate a two-dimensional domain: (1) cell-centered type where the cells of the mesh form the control volumes with their centroid as reference locations and (2) vertex-centered type in which the mesh vertices are chosen as the control volumes reference points and the perimeter of each control volume (median-dual) is defined as the lines surrounding a vertex connecting the midpoint of edges to the centroid of the cells. Figure 2.1 illustrates the definition of both types of control volumes on a triangular mesh. In three-dimensions, ANSLib only supports cell-centered data structure as the formation of vertex-centered control volumes requires a large number of quadrature points per cell.

To discretize the flow equations using the finite volume method, the governing equations should be recast in fully conservative form as:

$$\frac{\partial U}{\partial t} + \nabla \cdot \vec{F} = 0 \tag{2.1}$$

in which U denotes the conserved solution vector and $\vec{F}$ is the flux vector.

Integrating Equation 2.1 over an arbitrary control volume in 2D and using the divergence theorem gives the finite volume formulation of the governing equations in the form of an area and a surface integral, Equation 2.2, where dA is the infinitesimal area and $\hat{n}$ and ds represent the outward unit normal vector and infinitesimal length



(a) Cell-center control volume

(b) Vertex-center control volume

Figure 2.1: Control volume illustration in 2D

along the surface, respectively.

$$\int \int_{CV} \frac{\partial \vec{U}}{\partial t} dA + \oint_{CS} \vec{F} \cdot \hat{n} \, ds = 0 \tag{2.2}$$

Assuming the discretized physical domain does not change in time, U can be brought out from the integral in Equation 2.2, as the average solution vector of the control volume:

$$A_{CV_i} \frac{d\overline{U}_i}{dt} = - \oint_{CS_i} \vec{F} \cdot \hat{n} \, ds \tag{2.3}$$

The left hand side of Equation 2.3 is the time derivative of the average solution vector in the i^{th} control volume. The right hand side is called the flux integral or residual of the control volume, which is dependent on the solution vector in general and represents the spatial discretization of the same control volume:

$$A_{CV_i} \frac{d\overline{U}_i}{dt} = -R\left(\overline{U}_i\right) \tag{2.4}$$

The residual is evaluated by numerical integration along the interfaces of a control volume using sufficient quadrature points.

2.1 Governing Equations

Considering that the focus of this research is on two-dimensional compressible flow problems, we turn our attention to 2D descriptions. The conservative form of the compressible Navier-Stokes equations describing the conservation of mass, momentum and total energy in two dimensions is given as:

$$\frac{\partial U}{\partial t} + \nabla \cdot \left(\vec{F}_c \left(U \right) - \vec{F}_v \left(U, \nabla U \right) \right) = 0 \tag{2.5}$$

where U is the conserved solution vector, $\vec{F}_c$ is the convective flux, which depends on solution only and $\vec{F}_v$ is the viscous flux, which is dependent on both the solution and gradient. For a compressible viscous laminar flow, the solution and flux vectors are:

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E_t \end{pmatrix}, \quad F_c^x = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u \left(E_t + P \right) \end{pmatrix}, \quad F_c^y = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ v \left(E_t + P \right) \end{pmatrix}$$

$$F_v^x = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} + c_p \left(\frac{\mu}{Pr} \right) \frac{\partial T}{\partial x} \end{pmatrix}, \quad F_v^y = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} + c_p \left(\frac{\mu}{Pr} \right) \frac{\partial T}{\partial y} \end{pmatrix} \tag{2.6}$$

where ρ is the fluid density, $\vec{V} = (u, v)$ are the Cartesian velocity components, P is the fluid pressure, E_t is the total energy, c_p is the specific heat at constant pressure, T is the fluid temperature, Pr is the Prandtl number and τ_{ij} is the viscous stress tensor. Assuming a Newtonian fluid, the viscous stress term becomes:

$$\tau_{ij} = 2\mu\dot{\gamma}_{ij} \tag{2.7}$$

$$\dot{\gamma}_{ij} = \frac{1}{2} \left(\frac{\partial V_i}{\partial x_j} + \frac{\partial V_j}{\partial x_i} \right) - \frac{1}{3} \frac{\partial V_k}{\partial x_k} \delta_{ij}$$

in which μ is the fluid dynamic viscosity. The fluid pressure can be related to the total energy by introducing the ideal gas equation of state given as:

$$P = (\gamma - 1)\left[E_t - \frac{1}{2}\rho\left(u^2 + v^2\right)\right] \tag{2.8}$$

Note that the conserved solution vector is different from primitive variables, $W = (\rho\, u\, v\, P)^T$. We assume that the working fluid is air, $\gamma = 1.4$, $Pr = 0.72$. The viscous flux is zero in the case of an inviscid flow governed by Euler's equation.

2.2 Solution Reconstruction

The spatial accuracy of the finite volume solution depends on the accuracy of the flux integral. A high-order accurate flux integral in a control volume requires an accurate numerical flux and accurate integration. An accurate numerical flux is obtained by an accurate approximation of the unknown variable in the control volume. Assume that the unknown variable represented by ϕ is one of the primitive variables of the flow field such as density, pressure, or velocity components. The flow solver approximates the unknown variables of the flow field in the control volume by reconstructing a piecewise polynomial about the control volume's reference point (x_i, y_i).

$$\begin{aligned}\phi_i^R(x,y) &= \phi|_i + \left.\frac{\partial\phi}{\partial x}\right|_i (x - x_i) + \left.\frac{\partial\phi}{\partial y}\right|_i (y - y_i) + \left.\frac{\partial^2\phi}{\partial x^2}\right|_i \frac{(x - x_i)^2}{2} \\ &\quad + \left.\frac{\partial^2\phi}{\partial x\partial y}\right|_i (x - x_i)(y - y_i) + \left.\frac{\partial^2\phi}{\partial y^2}\right|_i \frac{(y - y_i)^2}{2} + \ldots \end{aligned} \tag{2.9}$$

In Equation 2.9, ϕ_i is the value of the reconstructed solution and $\frac{\partial^{n+m}\phi_i}{\partial x^n \partial y^m}$ are its derivatives at the reference point of control volume i. These values are the coefficients of the Taylor polynomial. The degree of the reconstructed polynomial determines the order of accuracy of the solution. For example, a second-order accurate solution approximation can be achieved by knowing the gradient of the solution at the control volume reference point and reconstructing a linear polynomial in the control volume.

$$\phi_i^R(x,y) = \phi|_i + \left.\frac{\partial\phi}{\partial x}\right|_i (x - x_i) + \left.\frac{\partial\phi}{\partial y}\right|_i (y - y_i) + \mathcal{O}\left(\Delta x^2, \Delta y^2\right) \tag{2.10}$$

Higher order solutions require the values of higher derivatives at the reference point and a higher order polynomial.

For compressible flow simulations, the hyperbolic system of equations may lead to discontinuities in the solution. Therefore, the reconstruction operator must ensure the desired order of accuracy while capturing discontinuities such as shocks. Essentially non-oscillatory (ENO) finite volume schemes were first developed for structured meshes [36] and then extended to irregular unstructured meshes by Abgrall [51] and Sonar [52]. In this family of schemes, including weighted ENO (WENO) schemes, the reconstruction stencil is selected based on the solution smoothness and thus stencils containing discontinuities are avoided. In this way, the reconstructed solution falls within the expected order of accuracy and sharp changes in the solution are captured automatically. Nevertheless, these higher-order finite volume schemes suffer from difficulties in selecting the appropriate stencil for multi-dimensional problems with a large number of variables. For these problems, the selection of a different stencil for each flow variable results in complexities and computational costs that limit their widespread use in general applications.

The construction of a k-exact reconstruction operator is another candidate to reconstruct a solution polynomial based on the control volume data such that the truncation error of the solution in each control volume remains at the desired order of accuracy. This method dates back to Barth and Frederickson's work designing a quadratic reconstruction operator to estimate the advective flux of the Euler equations [46]. Also, Barth incorporated the reconstruction operator into a upwind finite volume scheme to solve a range of advection-diffusion equations [53]. Later, Ollivier-Gooch and Van Altena [54] described a new k-exact reconstruction procedure for the higher-order approximation of gradients within a cell which are essential for diffusive fluxes. They demonstrated the high-order accuracy of the reconstruction operator for an advection-diffusion problem. In this approach, a fixed stencil is used and a least-squares system is formed that gives the reconstruction coefficients at the reference point of each control volume. For hyperbolic system of equations, the k-exact reconstruction procedure can be combined with a higher-order limiting strategy to capture flow discontinuities [55].

Our flow solver uses k-exact reconstruction where the coefficients are computed by minimizing the error in predicting the mean value of nearby control volumes [54].

The conservation of the mean within a control volume requires that

$$\frac{1}{A_i}\int_{V_i}\phi_i^R dA = \bar{\phi}_i \tag{2.11}$$

By expanding the left-hand side of Equation 2.11 term by term, one can easily show that

$$\frac{1}{A_i}\int_{V_i}\phi_i^R dA = \phi|_i + \left.\frac{\partial\phi}{\partial x}\right|_i \overline{x}_i + \left.\frac{\partial\phi}{\partial y}\right|_i \overline{y}_i + \left.\frac{\partial^2\phi}{\partial x^2}\right|_i \frac{\overline{x^2}_i}{2} + \left.\frac{\partial^2\phi}{\partial x\partial y}\right|_i \overline{xy}_i + \left.\frac{\partial^2\phi}{\partial y^2}\right|_i \frac{\overline{y^2}_i}{2} + ... \tag{2.12}$$

where

$$\overline{x^n y^m}_i = \frac{1}{A_i}\int_{V_i}(x-x_i)^n (y-y_i)^m dA. \tag{2.13}$$

In addition, the error of the mean value of the reconstructed solution for control volumes in the stencil $\{V_j\}_i$ should be minimized. In other words, the difference between the actual control volume average $\bar{\phi}_j$ and the average of ϕ_i^R over control volume j is minimized. The mean value for a single control volume V_j of the reconstructed solution ϕ_i^R is

$$\begin{aligned}\frac{1}{A_j}\int_{V_j}\phi_i^R dA &= \phi|_i + \left.\frac{\partial\phi}{\partial x}\right|_i \left\{\frac{1}{A_j}\int_{V_j}(x-x_i)\,dA\right\} + \left.\frac{\partial\phi}{\partial y}\right|_i \left\{\frac{1}{A_j}\int_{V_j}(y-y_i)\,dA\right\} \\ &+ \left.\frac{\partial^2\phi}{\partial x^2}\right|_i \left\{\frac{1}{2A_j}\int_{V_j}(x-x_i)^2\,dA\right\} \\ &+ \left.\frac{\partial^2\phi}{\partial x\partial y}\right|_i \left\{\frac{1}{A_j}\int_{V_j}(x-x_i)(y-y_i)\,dA\right\} \\ &+ \left.\frac{\partial^2\phi}{\partial y^2}\right|_i \left\{\frac{1}{2A_j}\int_{V_j}(y-y_i)^2\,dA\right\} + ...\end{aligned} \tag{2.14}$$

To avoid computing moments of each control volume in $\{V_j\}_i$ about the reference point of control volume i, replace $x - x_i$ and $y - y_i$ with $(x - x_j) + (x_j - x_i)$ and

$(y - y_j) + (y_j - y_i)$, respectively. Using Equation 2.13, we obtain

$$\begin{aligned}
\frac{1}{A_j}\int_{V_j} \phi_i^R dA &= \phi_i + \left.\frac{\partial \phi}{\partial x}\right|_i \left(\overline{x}_j + (x_j - x_i)\right) + \left.\frac{\partial \phi}{\partial y}\right|_i \left(\overline{y}_j + (y_j - y_i)\right) \\
&+ \left.\frac{\partial^2 \phi}{\partial x^2}\right|_i \left(\frac{\overline{x^2}_j + 2\overline{x}_j (x_j - x_i) + (x_j - x_i)^2}{2}\right) \\
&+ \left.\frac{\partial^2 \phi}{\partial x \partial y}\right|_i \left(\overline{xy}_j + \overline{x}_j (y_j - y_i) + (x_j - x_i)\,\overline{y}_j + (x_j - x_i)(y_j - y_i)\right) \\
&+ \left.\frac{\partial^2 \phi}{\partial y^2}\right|_i \left(\frac{\overline{y^2}_j + 2\overline{y}_j (y_j - y_i) + (y_j - y_i)^2}{2}\right) + \ldots \qquad (2.15)
\end{aligned}$$

The geometric terms in this equation are just dependent on the mesh and can be computed once and stored. Using the parallel axis theorem, it is possible to show that these terms have the general form of

$$\begin{aligned}
\widehat{x^n y^m}_{ij} &\equiv \frac{1}{A_j}\int_{V_j} \left((x - x_j) + (x_j - x_i)\right)^n \cdot \left((y - y_j) + (y_j - y_i)\right)^m dA \qquad (2.16) \\
&= \sum_{l=0}^{n}\sum_{k=0}^{m} \frac{n!}{l!\,(n-l)!}\frac{m!}{k!\,(m-k)!}(x_j - x_i)^k \cdot (y_j - y_i)^l \cdot \overline{x^{n-k} y^{m-l}}_j
\end{aligned}$$

Hence, Equation. 2.15 can be re-written as

$$\begin{aligned}
\bar{\phi}_j &= \phi|_i + \left.\frac{\partial \phi}{\partial x}\right|_i \widehat{x}_{ij} + \left.\frac{\partial \phi}{\partial y}\right|_i \widehat{y}_{ij} \qquad (2.17) \\
&+ \left.\frac{\partial^2 \phi}{\partial x^2}\right|_i \frac{\widehat{x^2}_{ij}}{2} + \left.\frac{\partial^2 \phi}{\partial x\,\partial y}\right|_i \widehat{xy}_{ij} + \left.\frac{\partial^2 \phi}{\partial y^2}\right|_i \frac{\widehat{y^2}_{ij}}{2} + \cdots
\end{aligned}$$

This equation is written for every control volume within the stencil of control volume i. The minimum number of neighboring control volumes in the stencil is equal to the number of reconstruction coefficients. The control volumes are chosen [54] based on their topological proximity to the reconstruction control volume as all neighbors at a given level are added one by one until the number of control volumes in the stencil reaches the number requested for each order of accuracy. The requested number is typically 50% more than the minimum to result in more robust solution reconstruction in the presence of non-smooth and/or vigorously oscillatory data. Figure 2.2 gives this choice of stencil for an interior control volume labeled R for a cell-centered case; the numeric labels show the order of accuracy at which a control volume is added to

the stencil. The resulting least-squares problem is

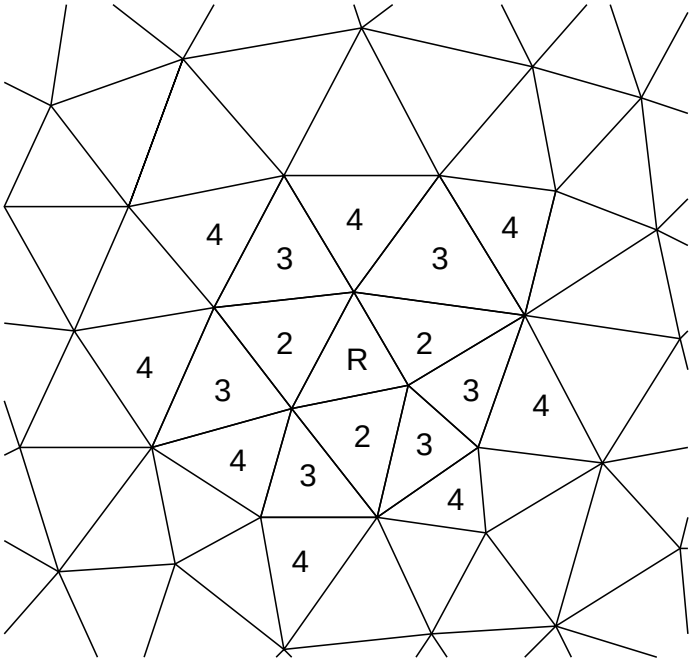


Figure 2.2: Cell-centered stencil

$$\left[\begin{array}{cccccc|c} 1 & \overline{x}_i & \overline{y}_i & \overline{x^2}_i & \overline{xy}_i & \overline{y^2}_i & \cdots \\ \hline w_{i1} & w_{i1}\widehat{x}_{i1} & w_{i1}\widehat{y}_{i1} & w_{i1}\widehat{x^2}_{i1} & w_{i1}\widehat{xy}_{i1} & w_{i1}\widehat{y^2}_{i1} & \cdots \\ w_{i2} & w_{i2}\widehat{x}_{i2} & w_{i2}\widehat{y}_{i2} & w_{i2}\widehat{x^2}_{i2} & w_{i2}\widehat{xy}_{i2} & w_{i2}\widehat{y^2}_{i2} & \cdots \\ w_{i3} & w_{i3}\widehat{x}_{i3} & w_{i3}\widehat{y}_{i3} & w_{i3}\widehat{x^2}_{i3} & w_{i3}\widehat{xy}_{i3} & w_{i3}\widehat{y^2}_{i3} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \\ w_{iN} & w_{iN}\widehat{x}_{iN} & w_{iN}\widehat{y}_{iN} & w_{iN}\widehat{x^2}_{iN} & w_{iN}\widehat{xy}_{iN} & w_{iN}\widehat{y^2}_{iN} & \cdots \end{array}\right] \left(\begin{array}{c} \phi \\ \hline \frac{\partial \phi}{\partial x} \\ \frac{\partial \phi}{\partial y} \\ \frac{1}{2}\frac{\partial^2 \phi}{\partial x^2} \\ \frac{\partial^2 \phi}{\partial x \partial y} \\ \frac{1}{2}\frac{\partial^2 \phi}{\partial y^2} \\ \vdots \end{array}\right)_i = \left(\begin{array}{c} \bar{\phi}_i \\ \hline w_{i1}\bar{\phi}_1 \\ w_{i2}\bar{\phi}_2 \\ w_{i3}\bar{\phi}_3 \\ \vdots \\ w_{iN}\bar{\phi}_N \end{array}\right) \tag{2.18}$$

where N is the number of nearby control volumes in the stencil and the line separates the conservation of mean constraint from equations to be solved by least-squares. In this problem, geometric weights w_{ij} can be set to emphasize the importance of geometrically nearby data:

$$w_{ij} = \frac{1}{|\vec{r}_j - \vec{r}_i|^t} \tag{2.19}$$

Throughout this thesis, the unweighted LS refers to the case where $t = 0$ while the weighted LS implies $t = 1$. Note that increasing the value of t in Equation 2.19,

highlights the importance of geometrically closer cells in the reconstructed polynomial regardless of the order of accuracy. The least-squares system of Equation 2.18 is a standard least-squares problem with the mean constraint. Gaussian elimination is applied for the constraint, replacing the sub-diagonal entries in the first column with zeros, and the remaining unconstrained (reduced) least-squares problem is solved for every control volume by the singular value decomposition (SVD) method [56]. In the case of viscous flow simulations, the Dirichlet boundary conditions pertaining to the no-slip condition at the walls are also added to the least-squares system. Although this can be done by enforcing those conditions as extra hard constraints [54], we just leave them for least-squares minimization with a much larger weight.

Having computed the coefficients of the piecewise polynomial, the flux vectors must be evaluated at the quadrature points along the interfaces of a control volume to obtain the numerical flux integral. Considering that the flux vectors depend on the solution and/or gradient of the solution, the Taylor series expansions must be used to yield such values in an arbitrary quadrature point within the control volume, (x_g, y_g):

$$\begin{aligned}\phi_g &= \phi|_i + \left.\frac{\partial \phi}{\partial x}\right|_i (x_g - x_i) + \left.\frac{\partial \phi}{\partial y}\right|_i (y_g - y_i) + \left.\frac{\partial^2 \phi}{\partial x^2}\right|_i \frac{(x_g - x_i)^2}{2} \\ &\quad + \left.\frac{\partial^2 \phi}{\partial x \partial y}\right|_i (x_g - x_i)(y_g - y_i) + \left.\frac{\partial^2 \phi}{\partial y^2}\right|_i \frac{(y_g - y_i)^2}{2} + \ldots \end{aligned} \tag{2.20}$$

$$\left.\frac{\partial \phi}{\partial x}\right|_g = \left.\frac{\partial \phi}{\partial x}\right|_i + \left.\frac{\partial^2 \phi}{\partial x^2}\right|_i (x_g - x_i) + \left.\frac{\partial^2 \phi}{\partial x \partial y}\right|_i (y_g - y_i) + \ldots \tag{2.21}$$

$$\left.\frac{\partial \phi}{\partial y}\right|_g = \left.\frac{\partial \phi}{\partial y}\right|_i + \left.\frac{\partial^2 \phi}{\partial x \partial y}\right|_i (x_g - x_i) + \left.\frac{\partial^2 \phi}{\partial y^2}\right|_i (y_g - y_i) + \ldots \tag{2.22}$$

2.3 Flux Functions

The solutions and gradients reconstructed at the two sides of an arbitrary quadrature point using Equations 2.20 to 2.22 are not necessarily equal (Figure 2.3). However, the conservation property of finite volume method requires that the flux leaving a control volume must enter its neighbor. A numerical flux function is required to

construct a unique flux vector from the two different ones obtained from each side. Flux functions used for the computation of numerical fluxes are divided into two major categories: convective (inviscid) and diffusive (viscous) flux functions. These two groups are characterized based on the physical properties of convection and diffusion. The transport of properties with the velocity of flow field is known as convection whereas the mixing or mass transport without requiring bulk motion is referred to as diffusion. Convective fluxes only depend on the solution while viscous fluxes are also dependent on the gradient of the solution as shown in Equation 2.6. The flux functions used for each category should respect the physical characteristics of the fluid flow to provide a robust and accurate discretization of the governing equations. Here, we explain the flux functions used in ANSLib for each type for the Euler/Navier-Stokes equations.

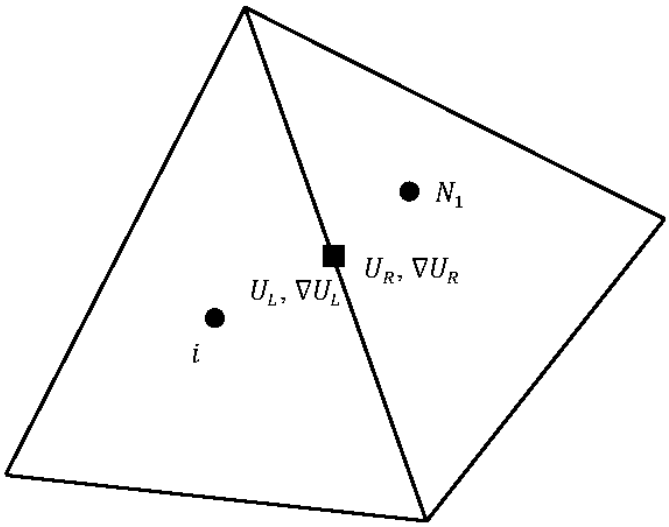


Figure 2.3: Solution and gradient reconstruction from two sides for flux integration

2.3.1 Convective Fluxes

The convective (inviscid) part of the governing equations for compressible flow problems is a classic example of a system of non-linear hyperbolic partial differential equations (PDEs). To have an idea about the appropriate treatment of convective fluxes, it is helpful to start from the first-order linear wave equation known as the

simplest hyperbolic PDE:

$$\frac{\partial \phi}{\partial t} + a\frac{\partial \phi}{\partial x} = 0 \tag{2.23}$$

This equation governs the propagation of waves traveling at a wave speed a. For positive values of a, the wave propagates strictly from left to right along the x axis as shown in Figure 2.4. According to the physical characteristics of the model equation, it is evident that the information in the field is propagating in the wave direction (left to right). Therefore, the solution at point i is influenced by the solution at point $i-1$ and the solution at point $i+1$ will not physically affect point i. As a result, in finite difference methods, it is reasonable to replace $\frac{\partial \phi}{\partial x}$ by $\frac{\phi_i - \phi_{i-1}}{\Delta x}$ rather than forward or central differencing. Such a treatment is called upwind discretization where the information is borrowed from upstream in the direction of "wind". It is well understood that using central or forward differencing can cause non-physical and/or oscillatory behavior and often leads to instability in the solution as these schemes do not follow the physical characteristics of the governing equation. Upwind discretization in the direction where the information travels is ideal for other hyperbolic equations.

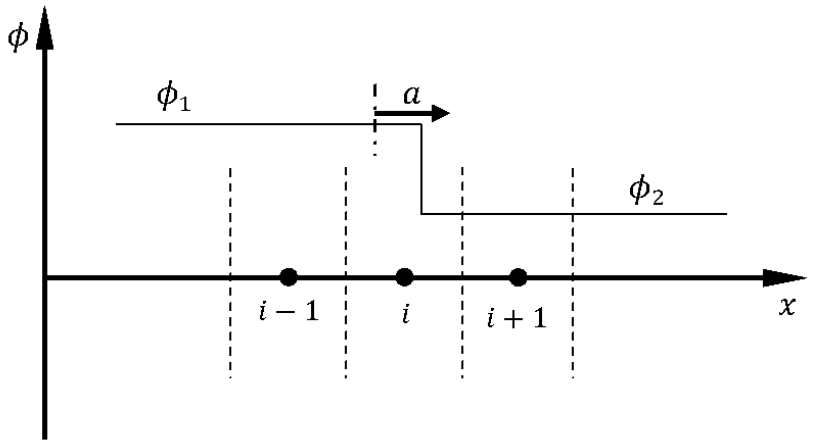


Figure 2.4: Propagation of a linear wave in positive direction

The Euler equations, which form the convective part of the governing equations, are a more complex example of hyperbolic equations due to the non-linearity and vector form. In this case, the information travels along the characteristic lines whose

slopes are the eigenvalues of the normal flux Jacobian, $\frac{\partial F_c^n}{\partial U}$:

$$F_c^n = F_c^x \hat{n}_x + F_c^y \hat{n}_y = \begin{pmatrix} \rho u \hat{n}_x + \rho v \hat{n}_y \\ (\rho u^2 + P)\,\hat{n}_x + \rho u v \hat{n}_y \\ \rho u v \hat{n}_x + (\rho v^2 + P)\,\hat{n}_y \\ u\,(E_t + P)\,\hat{n}_x + v\,(E_t + P)\,\hat{n}_y \end{pmatrix}$$

$$\frac{\partial F_c^n}{\partial U} = \begin{bmatrix} 0 & \hat{n}_x & \hat{n}_y & 0 \\ (\gamma-1)\,q\hat{n}_x - uV_n & V_n - (\gamma-2)\,u\hat{n}_x & u\hat{n}_y - (\gamma-1)\,v\hat{n}_x & (\gamma-1)\,\hat{n}_x \\ (\gamma-1)\,q\hat{n}_y - vV_n & v\hat{n}_x - (\gamma-1)\,u\hat{n}_y & V_n - (\gamma-2)\,v\hat{n}_y & (\gamma-1)\,\hat{n}_y \\ ((\gamma-1)\,q - h_t)\,V_n & h_t\hat{n}_x - (\gamma-1)\,uV_n & h_t\hat{n}_y - (\gamma-1)\,vV_n & \gamma V_n \end{bmatrix}$$

$$q = \frac{1}{2}\left(u^2 + v^2\right),\ V_n = u\,\hat{n}_x + v\,\hat{n}_y,\ h_t = h + q \tag{2.24}$$

It is possible to show that the eigenvalues of this matrix are :

$$\lambda_i = \begin{bmatrix} V_n & V_n & V_n + a & V_n - a \end{bmatrix} \tag{2.25}$$

where a is the sound speed. These characteristics lines provide the right direction for upwind discretization. Several classes of upwind methods such as flux vector splitting [57, 58], flux difference splitting [59] and advection upstream splitting methods [60, 61] have been proposed to discretize the hyperbolic system of compressible flows.

In our solver, we use Roe's approximate Riemann solver [62] for inviscid flux discretizations. The Roe flux function, which is classified as one of the flux difference splitting methods, includes the average of the two convective flux vectors computed from each side minus a dissipation term which splits (upwinds) the difference of the two fluxes:

$$F_c\,(U_L, U_R) = \frac{1}{2}\left(F_c\,(U_L) + F_c\,(U_R)\right) - \frac{1}{2}\left|\tilde{A}\right|(U_R - U_L) \tag{2.26}$$

The details of the flux difference splitting is very complicated and beyond the scope of this section. In this formulation, $\left|\tilde{A}\right|$ is the flux Jacobian matrix in diagonalized form as:

$$\left|\tilde{A}\right| = \tilde{X}^{-1}\left|\tilde{\Lambda}\right|\tilde{X},\ \left|\tilde{\Lambda}\right| = Diag\left(\left|\tilde{\lambda}_i\right|\right) \tag{2.27}$$

Note that $\tilde{()}$ means evaluations at Roe's average state defined as:

$$\tilde{\rho} = \sqrt{\rho_L \rho_R}$$

$$\tilde{u} = \frac{\sqrt{\rho_L} u_L + \sqrt{\rho_R} u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}$$

$$\tilde{v} = \frac{\sqrt{\rho_L} v_L + \sqrt{\rho_R} v_R}{\sqrt{\rho_L} + \sqrt{\rho_R}}$$

$$\tilde{h}_t = \frac{\sqrt{\rho_L} h_{t,L} + \sqrt{\rho_R} h_{t,R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{2.28}$$

2.3.2 Viscous Fluxes

As described earlier, the evaluation of diffusive fluxes requires estimates of the solution gradient at the quadrature points of a face. The diffusive (viscous) part of the governing equations by itself can be considered as a system of elliptic PDEs. It is well-known that central differencing is an optimal choice for the discretization of elliptic problems such as Poisson's equation. Even though for a structured scheme, a central differencing based on the cell averages of neighboring control volumes is usually computed to the desired order [63], this approach is problematic for unstructured grids, because the cell centroids are often far from lying on the perpendicular bisector of the face between them. This drawback decreases the order of accuracy of the computed gradient and therefore affects the total accuracy of solvers.

Instead, the face gradient can be determined by averaging the two gradients reconstructed from each side of a quadrature point using Equations 2.21 and 2.22. The most obvious way is arithmetic averaging in which equal weights are set for the two cell gradients; however, the weight of averaging can be tuned by some geometrical considerations within the cell. Volume weighted averaging (area weighted in 2D) or linear interpolation based on the distance of the opposite cell centroid to the face midpoint can be used as alternatives [64]. In ANSLib, we calculate the interface gradients of the primitive variables as the arithmetic average of the two reconstructed gradients:

$$(\nabla\phi)_F = \frac{1}{2}\left((\nabla\phi)_R + (\nabla\phi)_L\right) \tag{2.29}$$

Likewise, the values of those primitive variables involved in the calculation of viscous

fluxes are obtained by arithmetic averaging:

$$\phi_F = \frac{1}{2}\left(\phi_R + \phi_L\right) \tag{2.30}$$

Using the averaged gradients and values of Equations 2.29 and 2.30, the unique components of viscous flux vectors in Equation 2.6 can be calculated easily. Note that this strategy is robust and efficient for low-Reynolds laminar problems where nearly isotropic meshes are used. In those cases where highly anisotropic meshes are required, such a scheme leads to instability as will be shown later.

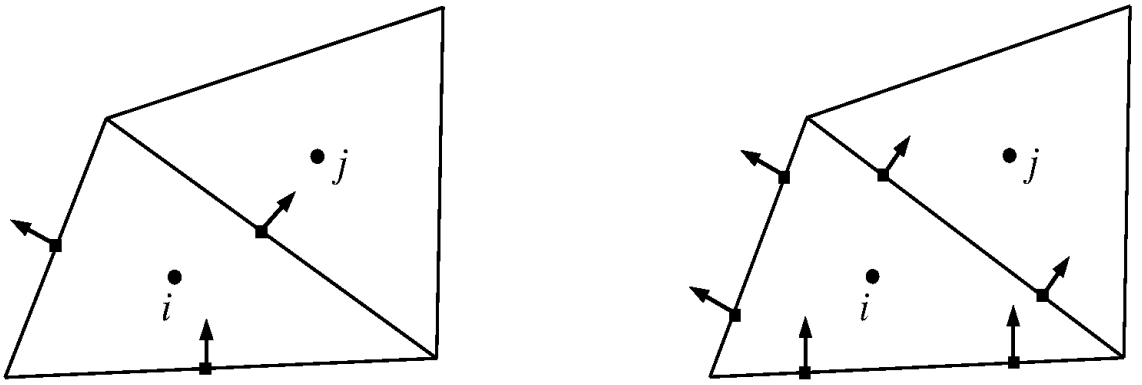
2.4 Integration

Numerical integration is involved in several parts such as residual evaluation, source term integration and control volume moment calculation. In all of these elements, numerical integration must be accurate enough not to degrade the overall accuracy of the discretization. A summary of quadrature rules and integration method used for each one is given in this section.

2.4.1 Flux Integral

To compute the flux integral for each control volume, numerical fluxes should be integrated over control volume faces. The accuracy of flux integration should be equal or higher than the accuracy of solution reconstruction. In other words, for a $p+1$-order accurate solution, we should be able to integrate a polynomial of degree p accurately. Gauss quadrature gives the capability of evaluating a definite integral with the integrand evaluated at only a few points [65]. For instance, one quadrature point located at the mid-face is sufficient for a linear reconstruction, i.e., second-order solution. For higher than second-order, more quadrature points per edge are required. Figure 2.5 shows the quadrature points for the flux integration schematically. It is possible to verify that these quadrature points are sufficient to obtain higher-order accurate flux integrals [66]. Comprehensive information regarding the locations and weights of the Gauss quadrature integration points has been discussed in the work of Stroud and Secrest [65].

The control volume flux integral in Equation 2.3 is approximated as the summation of flux integrals over the faces. The flux is integrated along each interior face only



(a) 2^{nd} order/1 Gauss point per face

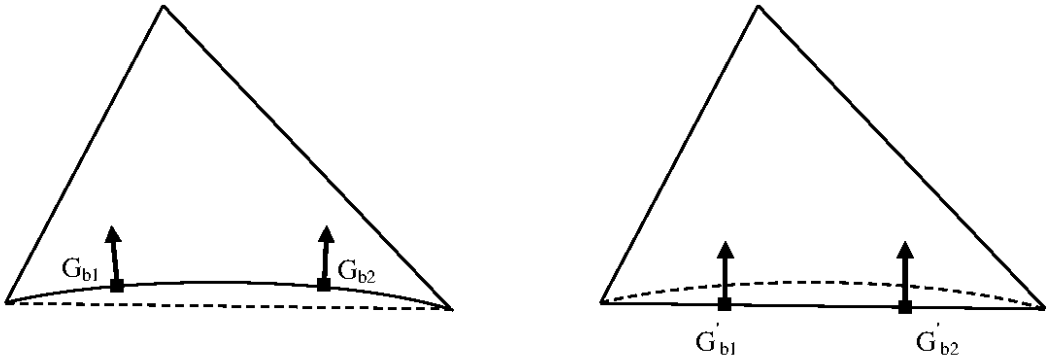
(b) 3^{rd} and 4^{th} order/2 Gauss points per face

Figure 2.5: Schematic illustration of Gauss quadrature points on straight faces

once and then added to the control volume with outward unit normal vector and subtracted from the other one. In this way, we can decrease the cost of flux integration by a factor of two, as well as ensuring local conservation to machine precision.

For those problems with curved boundaries, a piecewise linear representation of the boundary is not adequate for higher-order discretizations. This comes from the fact that the line segments between boundary vertices are separated from the actual curve by a distance that is $\mathcal{O}(h^2)$ for face length h. In general, the discrete approximation of the boundary must converge to the real boundary shape with the same order of accuracy as the order of discretization. For flux integration along curved boundaries, special care must be taken that the locations, unit normals and integration weights reflect the shape of the boundary appropriately. For general curved boundaries such as aerodynamic configurations in two-dimensions, we represent the geometry by a piecewise cubic spline which is sufficient up to fourth-order discretization (the distance from the actual curve is $\mathcal{O}(h^4)$). Flux integration along such curves uses the boundary representation directly in setting up the information of Gauss integration points [66]. It should be noted that the locations of the quadrature points are determined based on fractions of arc length, which are calculated iteratively for a cubic curve. Having the parametric representation of the curved geometry, the unit normal vectors can be obtained. Finally, the weights of integration are assigned based on arc length. Figure 2.6 shows the correct (points G_{b1} and G_{b2}) and wrong (points G'_{b1} and G'_{b2}) locations of Gauss points along a curved boundary face with their corresponding

normals. It is worth mentioning that two quadrature points along curved boundaries are used for flux integration of third- and fourth-order methods.



(a) Correct quadrature points (b) Wrong quadrature points

Figure 2.6: Schematic illustration of higher than second-order quadrature points on curved boundary faces

2.4.2 Source Term

Even though the Euler and Navier-Stokes equations do not have a source term, we need the capability of accurate source term integration for some model problems (e.g., Poisson's equation) and also testing manufactured solutions. In the case of having a source term, $S(U, \nabla U)$, the residual evaluation includes both the flux and source term integrals on control volumes:

$$R\left(\overline{U}_i\right) = \oint_{CS} \vec{F} \cdot \hat{n}\, ds - \int_{CV} S(U, \nabla U)\, dA \tag{2.31}$$

In contrast with the flux integral that is evaluated along the interfaces of a control volume, source term integral requires accurate numerical integration over control volumes. In this way, the source term integral can be approximated as:

$$\int_{CV} S(U, \nabla U)\, dA \approx \sum_{q=1}^{N_q} w_q\, S(x_q, y_q)\, A_{CV} \tag{2.32}$$

where N_q represents the number of quadrature points and w_q is the corresponding weight of integration for quadrature point q. Table 2.1 shows the number of quadra-

ture points required for each order of accuracy for different types of cell-centered elements in 2D and 3D. It is possible to verify the nominal accuracy of source term integration numerically using the reported number of quadrature points. Note that the locations of quadrature points for quads and hexes are obtained from the tensor product of Gauss points in 1D on a mapped reference element. The details about the location of quadrature points and weights of integration for triangles and tetrahedra can be found in Ref. [66]. The quadrature rule used for source term integration in 2D is exactly valid for flux integration in 3D where the faces of control volumes are triangles and quadrilaterals. Figure 2.7 shows the location of these points in 2D schematically for a fourth-order scheme.

	2nd order	3rd order	4th order
		2D	
triangle	1	3	4
quadrilateral	1	4	4
		3D	
tetrahedron	1	4	5
hex	1	8	8

Table 2.1: Number of quadrature points required for source term integration on cell-centered meshes

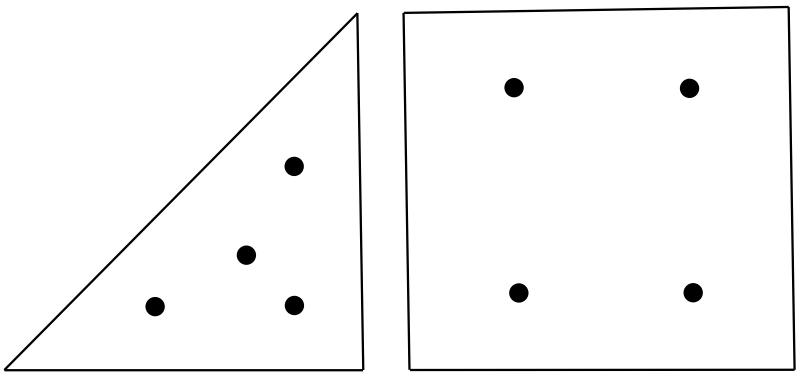


Figure 2.7: Schematic illustration of quadrature points for fourth-order integration of source terms over 2D cells

For accurate integration over those boundary cells that are adjacent to a curved

boundary, we can use the same approach employing ideas from isoparametric finite elements. However, we choose instead to use an approach for integration over irregularly shaped control volumes which guarantees exact integration for polynomials of degree p (and thus is $p+1$-order accurate). Such a strategy has the advantage of reducing the number of quadrature points for vertex-centered meshes in 2D as it removes the necessity of breaking a vertex-centered control volume into constituent parts. Furthermore, it only requires the calculation of moments of area, which must be computed for reconstruction anyway. More information about the implementation of this method for arbitrary control volumes is available in Ref. [66].

2.4.3 Moments of Area

The computation of moments of area, which is necessary for reconstruction, requires integration over the surface of each control volumes. To have a high fidelity solution reconstruction, we need to calculate these moments exactly. It is possible to calculate the moments using the same rule as source term integration:

$$\overline{x^n y^m}_i = \frac{1}{A_i} \sum_{q=1}^{N_q} w_q \, (x_q - x_i)^n \, (y_q - y_i)^m A_i \tag{2.33}$$

However, the integration rule for arbitrarily shaped control volumes such as vertex-centered ones or those adjacent to curved boundaries needs the computation of moments beforehand as described earlier. As a result, we compute the moment by using the Gauss's theorem to convert them into integrals around each control volume:

$$\overline{x^n y^m}_i = \frac{1}{(n+1)\, A_i} \oint_{CS} (x - x_i)^{n+1} \, (y - y_i)^m \, \hat{n}_x \, ds \tag{2.34}$$

where $\hat{n}_x$ is the x-component of the outward unit normal vector on the control volume boundary. In this way, we will have a unified framework for the calculation of moments for all types of control volumes as opposed to computing them using Equation 2.33 for straight cell-centered control volumes and using Equation 2.34 for irregularly shaped ones. The integral of Equation 2.34 is evaluated by using a Gaussian quadrature of appropriate order along each segment of the surface of each control volume as summarized in Table 2.2. The values given in this table are for straight control volume boundaries where the normal vector is constant. For control volumes

with curved boundaries, three Gauss points are used along the boundary face for all moments to account for variations in the direction of the unit normal vector.

Moment	Required for order $\geqslant$	No. of Gauss points
Area	1	1
$\bar{x}$, $\bar{y}$	2	2
$\overline{x^2}$, $\overline{xy}$,$\overline{y^2}$	3	2
$\overline{x^3}$, $\overline{x^2y}$,$\overline{xy^2}$,$\overline{y^3}$	4	3

Table 2.2: Number of quadrature points required for computing moments by integration around the control volumes

2.5 Time Advance Schemes

For time-dependent solutions, Equation 2.4 must be integrated in time to yield the solution at different time levels. This time integration can be done either explicitly or implicitly. In explicit time advance schemes, the spatial discretization represented by the residual vector is performed based on the solution at the current time level. The easiest explicit time integration scheme is known as explicit Euler where the known solution data at the current time level n is used to approximate the time derivative of the solution vector at control volume i:

$$A_i \frac{\bar{U}_i^{n+1} - \bar{U}_i^n}{\Delta t} = -R\left(\bar{U}_i^n\right) \tag{2.35}$$

The solution in the next time level $n+1$ is obtained directly by solving the above equation. Explicit Euler is only first-order in time and will degrade the overall accuracy of time-dependent solutions if combined with higher than first-order spatial discretizations. In general, the same order of time integration as the order of spatial discretization is required to obtain a high-order time-dependent solution. In ANSLib, we use multi-stage explicit Runge-Kutta schemes with appropriate order for time integration. Equation 2.36 gives a two-stage scheme which is second-order accurate in time and can be used for spatial discretizations up to second-order:

$$A_i \frac{\bar{U}_i^{n+1} - \bar{U}_i^*}{\Delta t} = -R\left(\bar{U}_i^*\right) \tag{2.36}$$

$$A_i \frac{\bar{U}_i^* - \bar{U}_i^n}{\Delta t/2} = -R\left(\bar{U}_i^n\right)$$

In this formulation, the intermediate state, $\bar{U}_i^*$, is obtained based on the solution at current time level n; the solution in the next time level requires another residual evaluation at the intermediate time level. Explicit time advance schemes are easy to program and exhibit robustness in many cases. However, they suffer from time-step restrictions enforced by the Courant-Friedrichs-Lewy (CFL) stability constraint that result in taking very small time steps, particularly for higher-order discretizations and fine meshes. In many engineering applications in which we seek the steady-state solution of the system, taking small time steps prohibitively slows down the convergence rate.

On the other hand, implicit time advance schemes use the next time level solution for spatial discretization. Implicit Euler, which is again first-order in time, can be written as:

$$\begin{aligned} A_i \frac{\bar{U}_i^{n+1} - \bar{U}_i^n}{\Delta t} &= -R\left(\bar{U}_i^{n+1}\right) \\ &= -\left[R\left(\bar{U}_i^n\right) + \frac{\partial R}{\partial U}\left(\bar{U}_i^{n+1} - \bar{U}_i^n\right) + \mathcal{O}\left(\left(\bar{U}_i^{n+1} - \bar{U}_i^n\right)^2\right)\right] \end{aligned} \tag{2.37}$$

or

$$\left(\frac{I}{\Delta t/A_i} + \frac{\partial R}{\partial \overline{\overline{U}}}\right)\delta U_i = -R\left(\bar{U}_i^n\right), \qquad \bar{U}_i^{n+1} = \bar{U}_i^n + \delta U_i \tag{2.38}$$

where $\frac{\partial R}{\partial \overline{\overline{U}}}$ is a large sparse matrix called the flux Jacobian representing the sensitivity of the spatial discretization to the control volume averages exist in the stencil and I is the identity matrix. The solution at the next time level is obtained by solving the linear system of Equation 2.38 and computing the update vector, δU_i.

For steady-state problems, $R\left(\bar{U}_i\right) = 0$, we do not care about the temporal accuracy of the solution. In this case, one may want to solve the non-linear system of equations by the direct application of Newton's methods for steady-state problems as:

$$\frac{\partial R}{\partial \overline{\overline{U}}}\delta U_i = -R\left(\bar{U}_i^n\right), \qquad \bar{U}_i^{n+1} = \bar{U}_i^n + \delta U_i \tag{2.39}$$

However, Newton's method will diverge if the initial guess is too far from the real solution. As a result, the linear system is augmented by a damping term which mimics the time derivative in the original time-dependent equations and prevents

the evolution of non-physical solution at each iteration. This is called the implicit pseudo time-stepping method, which has considerably less strict time-step restrictions compared to explicit methods and offers a significantly faster convergence to the steady-state solution.

Nevertheless, we must solve the large sparse linear system of Equation 2.38. The direct solution of the linear system is computationally intensive, so we use iterative methods for solving the linear system. In particular, we use the Generalized Minimum Residual (GMRES) method, which is a preferable choice for non-symmetric linear systems. GMRES, which approximates the solution by a vector with minimal residual in the Krylov subspace [67], has shown good performance for complex CFD problems [68, 69, 47]. The convergence properties of GMRES are highly sensitive to the conditioning of the matrix. As a result, a good approximation of the left-hand side matrix and a robust preconditioning strategy are required. Stationary methods such as Gauss Seidel and SOR are easy to implement and they are effective in damping high frequency errors. However, they often have restrictive stability conditions reducing the benefits of Newton's method. Alternatively, Incomplete Lower-Upper factorization methods, ILU(k), have proven to be a robust strategy for GMRES preconditioning [70, 7]. The number p shows the fill-level which determines the memory usage and the accuracy of ILU decomposition; using larger fill-level often leads to a more accurate factorization increasing the quality of preconditioning. However, there is a restriction in increasing the fill-level in practice due to memory limitations.

2.5.1 Jacobian Matrix

In addition, we need to obtain the global Jacobian matrix from our spatial discretization. This section describes how we calculate this matrix explicitly [49]. As stated before, the global Jacobian matrix represents the sensitivity of the residual vector to the control volume averages in the conservative form. For now, we assume that the residual vector only consists of the flux integral. In the case of having a source term that is dependent on the solution and/or gradient, the same procedure can be generalized. Using the chain rule, the global Jacobian can be expanded as:

$$\begin{aligned}\frac{\partial R}{\partial \bar{U}} &= \frac{\partial \text{FluxInt}}{\partial \text{CVar}} \\ &= \frac{\partial \text{FluxInt}}{\partial \text{Flux}} \frac{\partial \text{Flux}}{\partial \text{RecSol}} \frac{\partial \text{RecSol}}{\partial \text{RecCoef}} \frac{\partial \text{RecCoef}}{\partial \text{PVar}} \frac{\partial \text{PVar}}{\partial \text{CVar}}\end{aligned} \tag{2.40}$$

in which FluxInt is the flux integral, Flux is the numerical normal flux obtained by the application flux functions, RecSol are the reconstructed solutions and derivatives at Gauss points, RecCoef are the coefficients of reconstructed polynomial obtained by solving the least-squares system, PVar and CVar are the average of primitive and conserved variables over control volumes. To compute the Jacobian, each term is found for each Gauss point along the interfaces of a control volume using the following procedure:

1. $\frac{\partial \text{PVar}}{\partial \text{CVar}}$ is computed for each problem based on the relation between primitive and conserved variables.

2. The$\frac{\partial \text{RecCoef}}{\partial \text{PVar}}$ term can be found by using the pseudoinverse of the reconstruction matrix in the least-squares system of Equation 2.18. The pseudoinverse can be obtained by the SVD method once and stored since it is only dependent on geometric terms and does not change between iterations.

3. $\frac{\partial \text{RecSol}}{\partial \text{RecCoef}}$ term, which is also a geometric term and only dependent on the location of the Gauss point, is found using Equations 2.20 to 2.22.

4. Flux is defined as the dot product of numerical fluxes and unit normal vectors, $\vec{F} \cdot \hat{n}$, at each Gauss point. Therefore, $\frac{\partial \text{Flux}}{\partial \text{RecSol}}$ is found based on the choices of flux function for inviscid and viscous fluxes described in Section 2.3.

5. $\frac{\partial \text{FluxInt}}{\partial \text{Flux}}$ is easily computed as the weight of integration for the corresponding Gauss point.

Table 2.3 gives the breakdown of memory requirement for a simple subsonic inviscid flow around a multi-element airfoil with zero angle of attack [49]. The values in the table show the memory requirement in terms of the number of floating point numbers per control volume for the reconstruction matrix pseudoinverse, global Jacobian matrix, ILU factorized matrix and Krylov subspace.

2.6 Extension to Turbulent Flows

All the elements described so far lead to a higher-order unstructured finite volume solver for compressible inviscid and viscous laminar flows which is efficient in computing steady-state solutions. Figure 2.8 shows how the major elements of the solver

Order	Preconditioner	Recon.	Jacobian	ILU matrix	Krylov subsp.	Total
2	ILU(0)	11.7	299.2	299.2	252	862.1
2	ILU(1)	11.7	299.2	480.3	156	947.2
3	ILU(0)	88	585	585	172	1430
3	ILU(1)	88	585	1011.7	124	1808.6
4	ILU(0)	179.2	615.2	615.2	164	1573.7
4	ILU(1)	179.2	615.2	1087.6	108	1990

Table 2.3: Breakdown of memory requirement for an inviscid subsonic flow problem

are combined together to obtain the solution. In the pre-processing stage, an unstructured mesh of arbitrary type (e.g., cell-centered) and a desired physics, which includes all the information about flux functions and source term, are used to reconstruct an appropriate initial solution field to a desired order. Note that all those reconstruction elements (e.g., pseudoinverse and geometric terms) that are independent of solution are created in this stage and stored. The reconstructed solution is used to compute the residual vector and global Jacobian matrix using an appropriate quadrature rule. The residual and Jacobian are exported to the implicit pseudo time-stepper to update the solution vector. The new solution vector is brought back to recompute the reconstructed solution and thus residual (fluxes and source term) for the next iteration. This process continues until we converge to the steady-state solution which satisfies the discrete residual operator to some tolerance.

As stated earlier, one objective of this thesis is to extend the capabilities of the current flow solver so that we can compute subsonic and transonic turbulent flows on mixed-element meshes over aerodynamic configurations in two dimensions. For this purpose, we need several additional pieces highlighted by the gray boxes in Figure 2.8.

For turbulent flow simulations, it is common to have highly anisotropic cells in the mesh as the change of properties in one direction is considerably larger than the other one in turbulent boundary layer and wake regions. So the first required element is the ability to curve the interior faces of a mesh. As discussed, a high-order scheme needs a high-order representation of curved surfaces. For isotropic cells or cells with small aspect ratio which are preferable for the Euler and laminar Navier-Stokes equations, we can use the high-order representation in setting up the Gauss integration points of individual boundary cells with the method described in Section

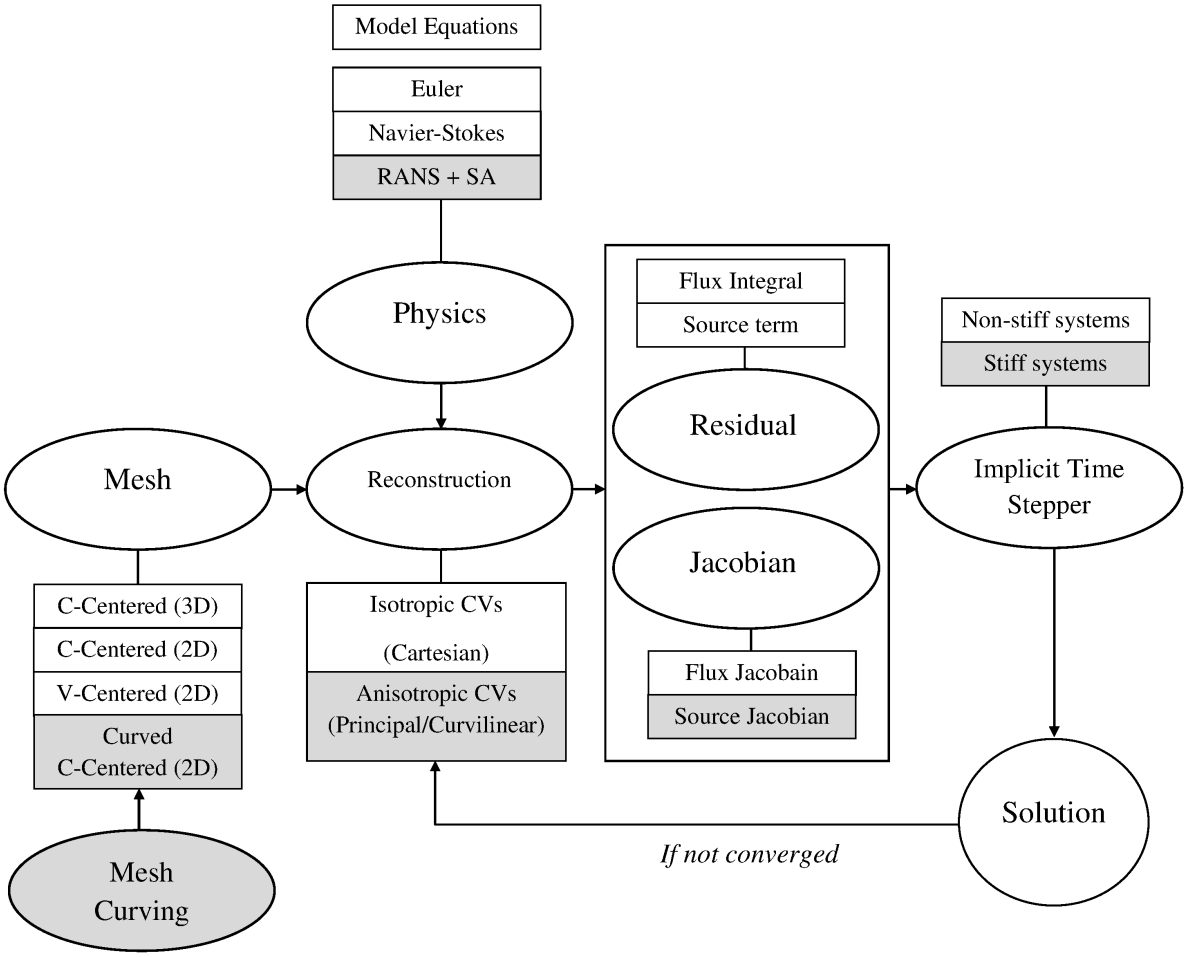


Figure 2.8: Combination of unstructured finite volume solver elements and required pieces for extension to turbulent flows

2.4. For turbulent flows with high aspect ratio cells, on the other hand, this strategy results in the intersection of boundary face with interior faces and thus non-physical meshes as will be shown later. Therefore, we need a mechanism to propagate the boundary curvature to interior faces. The details of this mechanism is described in Chapter 3.

Having curved the interior faces, we should pre-process the mesh. Considering that the median-duals for curved vertex-centered control volumes are difficult even to define uniquely, we choose to add the capability of handling curved meshes (with

curved interior faces) for cell-centered control volumes. Although there is no difference in the connectivity information between curved and straight cell-centered cases, some other geometric properties (such as moments, quadrature points, etc.) are computed differently, as discussed in Chapter 3.

It is also well-known that solution reconstruction on high aspect ratio meshes suffers from two known issues: poor conditioning of the least-squares system and poor accuracy for high-aspect ratio meshes in the presence of curvature. We will show later that both of these problems originate from the natural choice of Cartesian coordinate system for reconstruction. To resolve the issues, we need the capability of performing accurate solution reconstruction in other coordinate systems such as local principal and curvilinear systems. For this purpose, we must be able to find the mapping coefficients from Cartesian coordinates and also compute the moments of area in the appropriate coordinate system. The details of such a treatment are explained in Chapter 3.

In addition, we need a RANS turbulence model for the simulation of turbulent flows. In computational aerodynamics, it is quite standard to use either the Spalart-Allmaras one equation turbulence model [19] or some variant of the $k-\omega$ two-equation turbulence model [71, 72]. Both of these models have non-linear source terms dependent on the solution and gradient of the flow field, which is not the case for the other governing equations solved by our solver. Therefore, we need to enhance the ability of Jacobian calculations to include the sensitivity of the source term with respect to solution using the strategy described in Section 2.5. In this thesis, we use a variant of the Spalart-Allmaras (SA) model designed for higher-order discretizations and fully couple the evolution equation of turbulence working variable with RANS equations as will be discussed in Chapter 4.

The last element that needs to be modified is the implicit solver. For turbulent flow simulations, highly anisotropic cells induce significant stiffness into the discrete equations and have the potential to hamper the solution procedure. This also becomes worse for higher-order discretizations with smaller numerical dissipation. In addition, there is significant difference in the order of magnitude of flow field variables. Consequently, special attention should be taken to the solution strategy for these problems. This will be done by a robust selection of time-step, appropriate under-relaxation of the solution update and proper scaling of governing equations as will be described in Chapter 4.

Chapter 3

Anisotropic Mesh Treatment

In the context of turbulent flows for aerodynamic applications, it is quite common to have cells with high aspect ratio to resolve the flow properties in one direction as opposed to the direction where quantities change slowly. A typical example is turbulent boundary layer where the gradient of velocity components is quite high across the boundary layer whereas the gradients are small along the direction of flow. Obtaining sufficient resolution in such regions using isotropic cells leads to additional unnecessary work including huge computational cost and memory usage due to the large number of degrees of freedom.

As mentioned earlier, the presence of anisotropic cells in the mesh requires special treatment. An effective interior curving strategy, pre-computing geometrical properties of a mesh with curved faces and finally accurate and well-conditioned solution reconstruction on anisotropic cells are among those elements that need to be considered. This chapter explains each of these in detail.

3.1 Interior Curving Strategy

The effect of curvature must be taken into account in higher-order solvers by a more accurate representation of boundary faces. When the mesh is comprised of isotropic triangles, it is typically possible to just deform the boundary of the elements which are in contact with the curved boundary. However, this strategy fails when the cells are highly anisotropic as is common in boundary layer regions. Figure 3.1 shows the layers of anisotropic triangles over a curved boundary represented by the dashed red line. It can be seen that even for such a moderate aspect ratio, curving the boundary faces causes intersection with the interior faces and this will be more severe for triangles/quadrilaterals with higher aspect ratio. In this situation, it is necessary to propagate the mesh deformation into the domain interior to prevent faces from intersecting near curved boundaries.

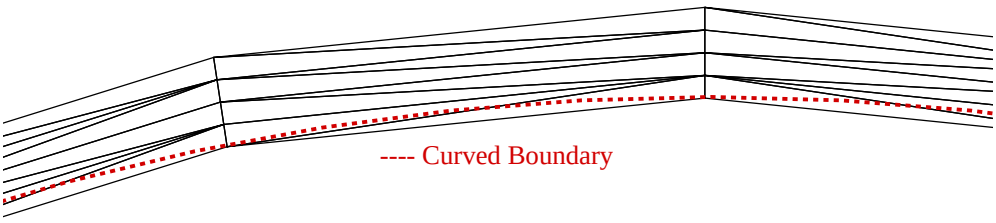


Figure 3.1: Curved boundary intersection with interior faces for anisotropic meshes

To curve the interior faces of a mesh, Luo [73] suggested a Bezier-based approach to assign curvature to interior faces based on the curvature of the boundary. Also, Sherwin *et al.* [74] proposed a type of hybrid meshing using quad elements close to the curved boundary and a curvature based refinement procedure. More recently, Persson and Peraire [75] devised a method for generating well-shaped curved unstructured meshes using a non-linear elasticity analogy. They used the high-order finite element method to solve a non-linear elasticity problem with the boundary curvature acting as an input deformation; the equilibrium displacement from the elasticity problem defines the curvature of internal faces.

In the current thesis, a modified linear elasticity method is used to project the boundary curvature into the interior edges. In this method, which has been used by Wang *et al.* [76], the geometry of the domain to be meshed is considered as an elastic solid that obeys the isotropic linear elasticity relations. For the initial linear mesh in which the Cartesian coordinates are denoted by (X, Y), the elasticity equation can be recast in the following form:

$$\begin{aligned} \frac{\partial}{\partial X}\left(d_{11}\frac{\partial \delta_X}{\partial X}+d_{12}\frac{\partial \delta_Y}{\partial Y}\right)+\frac{\partial}{\partial Y}\left(d_{33}\left(\frac{\partial \delta_X}{\partial Y}+\frac{\partial \delta_Y}{\partial X}\right)\right) &= 0 \\ \frac{\partial}{\partial X}\left(d_{33}\left(\frac{\partial \delta_X}{\partial Y}+\frac{\partial \delta_Y}{\partial X}\right)\right)+\frac{\partial}{\partial Y}\left(d_{21}\frac{\partial \delta_X}{\partial X}+d_{22}\frac{\partial \delta_Y}{\partial Y}\right) &= 0 \end{aligned} \tag{3.1}$$

$\delta = (\delta_X, \delta_Y)$ represents the nodal displacement vector in the Cartesian coordinate

direction and the coefficients, d, are defined as follows:

$$\begin{aligned} d_{11} &= d_{22} = \frac{E\,(1-\nu)}{(1+\nu)\,(1-2\nu)} \\ d_{12} &= d_{21} = \frac{E\nu}{(1+\nu)\,(1-2\nu)} \\ d_{33} &= \frac{E}{2\,(1+\nu)} \end{aligned} \tag{3.2}$$

in which E is assumed to be a constant throughout the domain and thus canceled and ν denotes Poisson's ratio, which is set to be 0.25. Equation 3.1 can be re-written in the following form:

$$\nabla \cdot S = 0 \tag{3.3}$$

where S represents the stress tensor.

$$S = \begin{bmatrix} \sigma_{XX} & \sigma_{XY} \\ \sigma_{YX} & \sigma_{YY} \end{bmatrix} \tag{3.4}$$

For the finite element discretization, Equation 3.3 is multiplied by an arbitrary test function, $z\,(X,Y)$, and integrated over the domain A. Integration by parts gives the weak formulation of the system of equations:

$$\int_A S\,(\delta_X, \delta_Y) \cdot \nabla z\,(X,Y)\; dA = 0 \tag{3.5}$$

where the boundary term is not present because the test functions vanish on boundaries due to essential Dirichlet boundary conditions. Following the standard finite element technique, the discrete displacement vector is represented at the nodes using nodal basis functions, $\phi\,(X,Y)$:

$$\vec{\delta}\,(x,y) = \sum_k \vec{\delta}_k \phi_k\,(X,Y) \tag{3.6}$$

Using the Galerkin finite element method, the test function is set equal to the nodal basis function at each node and thus the system of Equation 3.5 can be written as $K\delta = f$ where the stiffness matrix entries are 2×2 matrices:

$$K_{ij} = \begin{bmatrix} \int_A \left(d_{11} \frac{\partial \phi_i}{\partial X} \frac{\partial \phi_j}{\partial X} + d_{33} \frac{\partial \phi_i}{\partial Y} \frac{\partial \phi_j}{\partial Y} \right) dA & \int_A \left(d_{12} \frac{\partial \phi_i}{\partial X} \frac{\partial \phi_j}{\partial Y} + d_{33} \frac{\partial \phi_i}{\partial Y} \frac{\partial \phi_j}{\partial X} \right) dA \\ \int_A \left(d_{33} \frac{\partial \phi_i}{\partial X} \frac{\partial \phi_j}{\partial Y} + d_{21} \frac{\partial \phi_i}{\partial Y} \frac{\partial \phi_j}{\partial X} \right) dA & \int_A \left(d_{33} \frac{\partial \phi_i}{\partial X} \frac{\partial \phi_j}{\partial X} + d_{22} \frac{\partial \phi_i}{\partial Y} \frac{\partial \phi_j}{\partial Y} \right) dA \end{bmatrix} \tag{3.7}$$

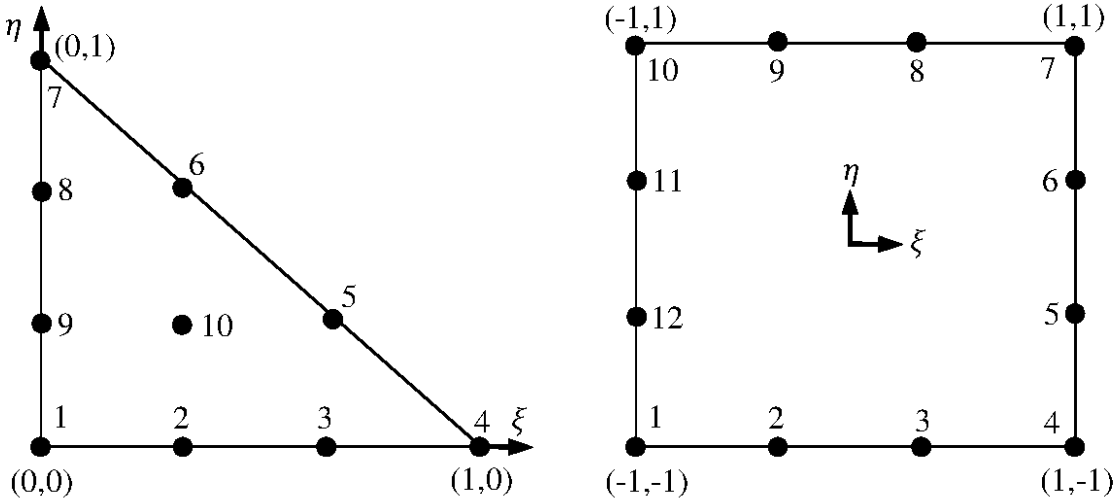
To compute the stiffness matrix, nodal basis functions are found for each element up to the desired order of accuracy. For this purpose, Lagrange cubic (10-node) elements for triangles and serendipity cubic (12-node) elements for quadrilaterals are used to represent the boundary geometry up to fourth-order accuracy. The serendipity cubic (12-node) element is chosen instead of a tensor-product cubic (16-node) element to decrease the cost of solving the linear elasticity problem. Figure 3.2 shows the reference element for both cases in the reference space of (ξ, η). The nodal basis functions for both the cubic reference elements are given in Appendix A. Note that the derivatives of basis functions in the physical space, $\left(\frac{\partial \phi}{\partial X}, \frac{\partial \phi}{\partial Y}\right)$, are needed in Equation 3.7 rather than those in the reference space, $\left(\frac{\partial \phi}{\partial \xi}, \frac{\partial \phi}{\partial \eta}\right)$. By knowing the mapping from the reference element to any of the physical cells, $(\xi, \eta) \rightarrow (X, Y)$, which is linear for triangles and bilinear for quadrilaterals, these derivatives can be computed at any arbitrary point by using:

$$\begin{bmatrix} \frac{\partial \phi}{\partial X} \\ \\ \frac{\partial \phi}{\partial Y} \end{bmatrix} = \begin{bmatrix} \frac{\partial X}{\partial \xi} & \frac{\partial Y}{\partial \xi} \\ \\ \frac{\partial X}{\partial \eta} & \frac{\partial Y}{\partial \eta} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial \phi}{\partial \xi} \\ \\ \frac{\partial \phi}{\partial \eta} \end{bmatrix} \tag{3.8}$$

The integrals of Equation 3.7 are computed using sixth-order quadrature rules [65] by evaluating Equation 3.8 in quadrature points of the reference elements, (ξ_q, η_q). The computed elemental stiffness matrices are assembled into a global matrix to form the left-hand side of the system of equations.

On the boundary of the domain, the displacement can be obtained for the two middle nodes of each boundary face by representing the geometry by a higher-order curve (Figure 3.3). The location of these nodes in the linear mesh is found by linear interpolation between the two end points of the corresponding face:

$$X_k = X_a + k\frac{X_b - X_a}{3},\ Y_k = Y_a + k\frac{Y_b - Y_a}{3} \ \text{ for } k = 1,\ 2 \tag{3.9}$$



(a) Triangle (b) Quadrilateral

Figure 3.2: Illustration of cubic reference elements

Using piecewise cubic splines for the geometry $((x_B(t), y_B(t))$, where $t = [0, 1]$ for each curved boundary face), the two middle nodes are assumed to be displaced to the locations obtained by interpolation in the parametric space of the same boundary face, $(x_{B,i}(t), y_{B,i}(t))$:

$$x_k = x_{B,i}\left(t = \frac{k}{3}\right),\ y_k = y_{B,i}\left(t = \frac{k}{3}\right) \quad \text{for } k = 1,\ 2 \tag{3.10}$$

Therefore, the boundary displacement which is imposed as a Dirichlet boundary condition is found as $\delta_i = (x_i - X_i, y_i - Y_i)$. Solving the system of equations arising from Equation 3.1 gives the displacement, and thus the location of nodes in the curved mesh. The linear system is solved via GMRES method preconditioned by ILU(1). Note that the Cartesian coordinates are represented by (x, y) in the higher-order curved mesh and by (X, Y) in the initial linear mesh. Figure 3.4 shows higher-order elements near the leading edge of the NACA 0012 geometry obtained by curving the initial linear mesh shown by dashed lines. As seen, all the interior faces are curved in this way to prevent the intersection of boundary and interior. Far from the curved surface, the deformations are considerably smaller although the cells are still represented by cubic faces.

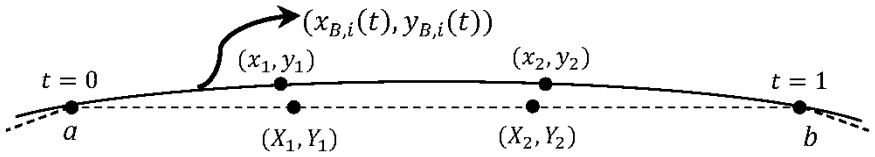


Figure 3.3: Schematic representation of boundary displacement for curving the interior faces of a linear mesh

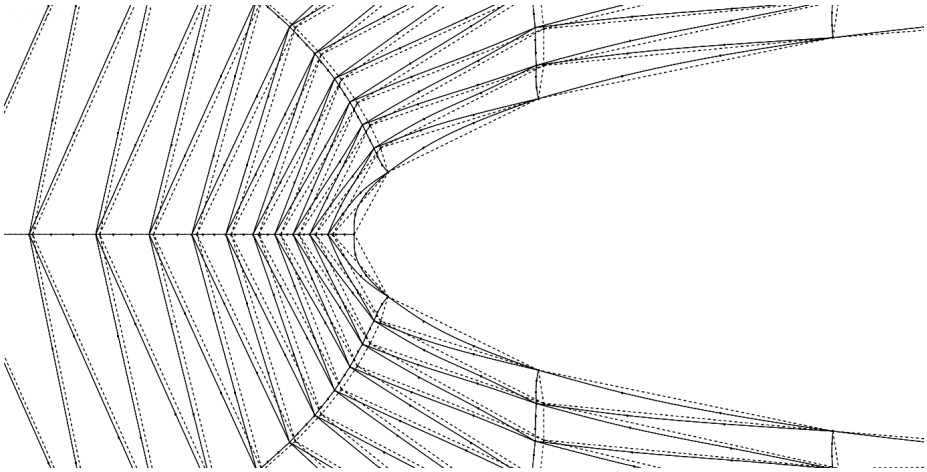


Figure 3.4: Representation of cubic cells obtained by interior curving of a linear mesh around the geometry of NACA 0012 (dashed lines: linear mesh, solid lines: cubic mesh)

3.2 Curved Mesh Support

After interior curving, the cells of the mesh are not triangles/quadrilaterals with straight faces. Therefore, special care should be taken to compute those geometrical features required for a finite volume solver. These include flux integral quadrature points on the interfaces along with corresponding normal vector and integration weight, and quadrature points and weights required for source term integration and moments of area, which are essential for solution reconstruction.

3.2.1 Flux Integral

A face of a higher-order mesh is comprised of four nodes coming directly from solving the linear elasticity equations at the nodes of each cubic element. Therefore, a cubic mapping from a reference straight line segment into each curved face (Figure 3.5) can

be created as:

$$x(\xi) = \sum_{k=1}^{4} x_k \phi_k(\xi) \ , \ y(\xi) = \sum_{k=1}^{4} y_k \phi_k(\xi) \tag{3.11}$$

where (x_i, y_i) are the coordinates of the nodes and $\phi_i(\xi)$ are the Lagrangian cubic interpolation functions in 1D (see Appendix A). Having this, we can also map flux integration along curved faces into flux integration along the straight reference line segment as:

$$\oint_{CS} \vec{F} \cdot \hat{n} \, ds = \oint_{CS} \vec{F} \cdot \hat{n} \, \sqrt{dx^2 + dy^2} = \oint_{CS} \vec{F} \cdot \hat{n} \, \sqrt{\left(\frac{dx}{d\xi}\right)^2 + \left(\frac{dy}{d\xi}\right)^2} \, d\xi \tag{3.12}$$

To evaluate the integral of Equation 3.12, we can use appropriate Gauss quadrature rules for the reference line segment. However, the flux vector must be evaluated at the quadrature points mapped to the physical space whose locations are obtained from Equation 3.11. The unit normal vectors at the quadrature points are found as $(-dy/d\xi, dx/d\xi)$. Due to the changes in the direction of unit normal vector, we use a quadrature rule that is able to integrate a polynomial of degree $2p+1$ exactly on the reference line segment for a $(p+1)$-accurate discretization. Our numerical experiments showed that this number of quadrature points is always sufficient.

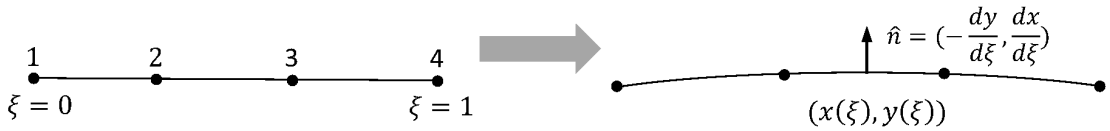


Figure 3.5: Illustration of mapping from a reference line segment into a general cubic face

3.2.2 Source Term

To compute the source term integral over a curved cell, we use the idea of isoparametric finite elements in which element geometry and displacement vector components are represented by the same type of basis functions. In other words, the Cartesian coordinates inside each cubic triangle/quadrilateral can be interpolated by the same

cubic basis functions used to solve the linear elasticity problem:

$$x\left(\xi,\eta\right)=\sum_{k=1}^{N}x_k\phi_k\left(\xi,\eta\right)\ ,\ y\left(\xi\right)=\sum_{k=1}^{N}y_k\phi_k\left(\xi,\eta\right) \tag{3.13}$$

Note that the number of nodes, N, is 10 for a cubic triangle and 12 for a cubic serendipity quadrilateral. Figure 3.6 shows such a mapping for cubic triangles schematically. To calculate the integral of a source term over a curved cell, we can use coordinate transformation to the reference element space as:

$$\int_{CV}S\left(U,\nabla U\right)\ dA=\int_{CV}S\left(U,\nabla U\right)\ dxdy=\int_{CV}S\left(U,\nabla U\right)\ \left|\frac{\partial\left(x,y\right)}{\partial\left(\xi,\eta\right)}\right|\ d\xi d\eta \tag{3.14}$$

in which $\left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right|=|\,J\,|$ denotes the determinant for the Jacobian of the coordinate transformation:

$$|\,J\,|=\left|\frac{\partial x}{\partial\xi}\frac{\partial y}{\partial\eta}-\frac{\partial y}{\partial\xi}\frac{\partial x}{\partial\eta}\right| \tag{3.15}$$

Therefore, the integral of Equation 3.14 can be evaluated using a quadrature rule that is able to integrate a polynomial of degree $2p$ exactly over the reference element. Note that the source term, $S\left(U,\nabla U\right)$, must be evaluated at the quadrature points mapped to the physical space whose coordinates are found using Equation 3.13.

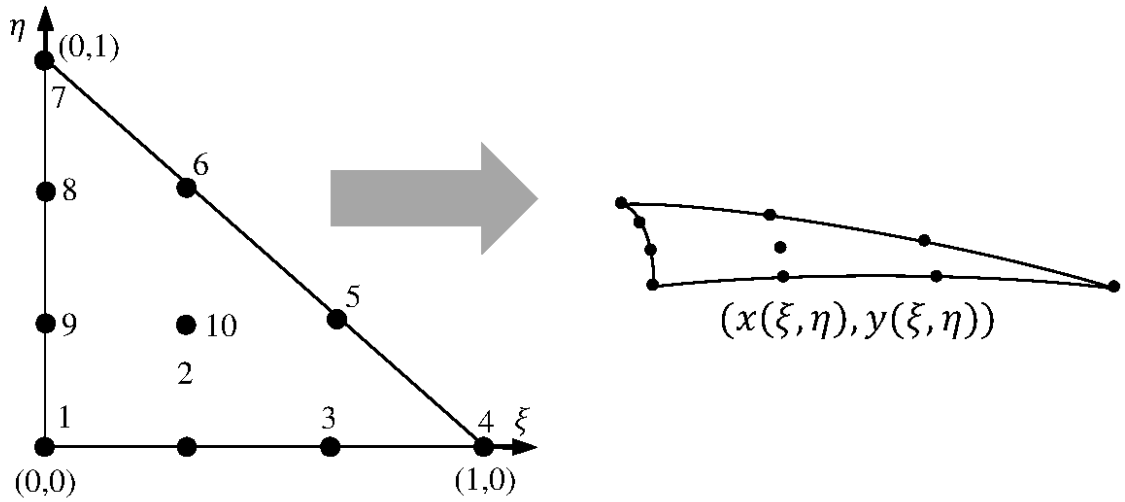


Figure 3.6: Illustration of mapping from a reference triangle into an arbitrary cubic triangular cell

3.2.3 Moments of Area

As described in Chapter 2, the least-squares solution reconstruction requires the moments of each control volume about its reference point. For this purpose, we can use the same strategy as source term integral since the moments of area are defined as volume integrals:

$$\overline{x^n y^m}_i = \frac{1}{A_i} \int_{CV_i} (x - x_i)^n (y - y_i)^m |\, J \,|\, d\xi d\eta \tag{3.16}$$

where the (x_i, y_i) is the coordinate of the reference point of a control volume. This point is obtained by mapping point $\left(\frac{1}{3}, \frac{1}{3}\right)$ for triangles and point $(0, 0)$ for quadrilaterals from the reference element to the physical space using Equation 3.13. To obtain sufficiently accurate moments, we use a quadrature rule that is able to integrate a monomial of degree $2p + 1$ exactly on the reference element for a $(p + 1)$-accurate reconstruction.

To assess the accuracy of curved boundary representation and moment calculations, the moments of area of a quarter-annulus, whose inner and outer radii are 7/8 and 9/8, are found about the origin and compared with the exact values. For this purpose, the moments of each element are computed first and then translated to the origin by the use of the parallel axis theorem. The error involved in computing the moments has been tabulated in Table 3.1 for two different N values which represent the number of divisions in both the radial and azimuthal directions. The ratio of error observed for different moments verifies the fourth-order accuracy of the curved boundary representation and also sufficient accuracy of numerical integration up to fourth order.

3.2.4 Distance Function

Another geometrical property that must be computed for anisotropic curved meshes is the distance of the cell reference location from the closest curved boundary. This quantity is needed for accurate solution reconstruction on high aspect ratio meshes near curved geometries as will be shown in the next section. In addition, the source term of the Spalart-Allmaras turbulence model is dependent on the distance from the closest wall. As a result, we need to pre-compute and store the distance from the closest wall boundary for all of the quadrature points used to compute the integral

Moment	$\lvert E\rvert_N$		$\frac{\lvert E\rvert_{20}}{\lvert E\rvert_{40}}$
	$N=20$	$N=40$	
A	7.37×10^{-8}	4.61×10^{-9}	15.97
$\overline{x}$	5.71×10^{-8}	3.66×10^{-9}	15.60
$\overline{y}$	5.71×10^{-8}	3.66×10^{-9}	15.60
$\overline{x^2}$	1.04×10^{-7}	6.24×10^{-9}	16.61
$\overline{xy}$	4.80×10^{-8}	3.38×10^{-9}	14.19
$\overline{y^2}$	1.03×10^{-7}	6.23×10^{-9}	16.48
$\overline{x^3}$	1.34×10^{-7}	8.05×10^{-9}	16.69
$\overline{x^2y}$	5.67×10^{-8}	3.70×10^{-9}	15.31
$\overline{xy^2}$	5.62×10^{-8}	3.70×10^{-9}	15.19
$\overline{y^3}$	1.33×10^{-7}	8.05×10^{-9}	16.49

Table 3.1: Error in calculating the moments of area for a quarter-annulus

of the source term. In this section, the method by which the distance from a higher-order curve is calculated for an arbitrary point (reference location and quadrature points) is described.

As discussed, we represent curved geometries with a piecewise cubic spline. Note that the cubic polynomial representing boundary face i (one piece of the spline) is different from other boundary faces:

$$x_{B,i}(t) = \alpha_{3,i}t^3 + \alpha_{2,i}t^2 + \alpha_{1,i}t + \alpha_{0,i} \tag{3.17}$$

$$y_{B,i}(t) = \beta_{3,i}t^3 + \beta_{2,i}t^2 + \beta_{1,i}t + \beta_{0,i}$$

To find the closest point to an arbitrary point of (x_0, y_0) on the curved geometry, we need to identify the boundary face on which the closest point exists (Figure 3.7). For this purpose, we compute the unit normal vectors at the two end points of each boundary face ($\hat{n}_1$ and $\hat{n}_2$). It is worth-mentioning that the first derivative and thus unit normal vector of a cubic spline are continuous at the points. We also find the vectors that connect each of the two end points to point (x_0, y_0). Computing the cross product of this vector, $\vec{r_i}$, and corresponding unit normal vector, $\hat{n}_i$, the closest point lies on the boundary face where the two cross products have different signs, i.e., $((\vec{r}_1 \times \hat{n}_1) \cdot (\vec{r}_2 \times \hat{n}_2)) < 0$. Note that the cross product of two vectors in two-dimensions is a scalar. Having found the appropriate boundary faces, the distance

squared of each point on that face from (x_0, y_0) is:

$$D^2 = \left(\alpha_{3,i}t^3 + \alpha_{2,i}t^2 + \alpha_{1,i}t + \alpha_{0,i} - x_0\right)^2 + \left(\beta_{3,i}t^3 + \beta_{2,i}t^2 + \beta_{1,i}t + \beta_{0,i} - y_0\right)^2 \tag{3.18}$$

The closest point is obtained as the one with minimum distance squared whose corresponding parameter satisfies $\frac{dD^2}{dt} = 0$. This minimization problem is solved for parameter t_c via Newton's method to machine zero tolerance. The corresponding parameter is substituted in Equation 3.17 to find the Cartesian coordinates of the closest point on the curved geometry. If we do not find any boundary face where the cross product changes sign, the closest point on the surfaces will be one of the corner points of the cubic spline (e.g., trailing edge). The distances from all of the corner points are measured and compared to find the closest point on the wall.

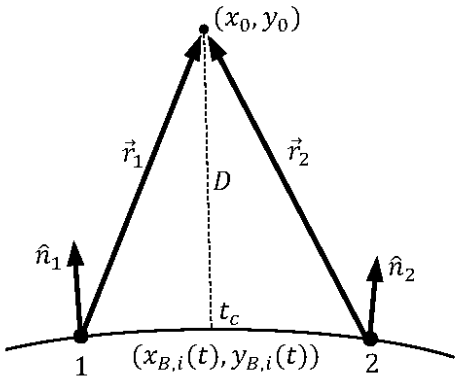


Figure 3.7: Illustration of distance function calculation for an arbitrary point

3.3 Reconstruction on Anisotropic Meshes

A known issue for k-exact solution reconstruction on highly anisotropic meshes is the poor conditioning of the least-squares (LS) system. This is exacerbated for higher-order computations as the presence of geometric terms with higher exponents results in a considerable difference in the magnitude of matrix columns. Furthermore, aerodynamic configurations typically consist of curved surfaces on which a flow boundary layer develops. Previous studies have demonstrated that least-squares reconstruction

on high aspect ratio meshes with finite curvature suffers from poor accuracy, even with straight-sided cells. Mavriplis [77] examined the accuracy of the least-squares technique for second-order discretization on unstructured meshes. He used a non-linear function of distance to the wall as a test function to evaluate the accuracy of the reconstruction on highly stretched meshes in the presence of surface curvature. The results showed that unweighted least-squares reconstruction fails to estimate the solution gradient; however, the accuracy can be recovered using an inverse distance weighting for vertex-centered discretizations. For a cell-centered method on triangular meshes, none of the control volumes within the stencil may be sufficiently close to the cell center under consideration. Therefore, inverse-distance weighting does not help and least-squares reconstruction exhibits poor accuracy. To improve the accuracy of reconstructed gradient on the curved meshes with high aspect ratio, Diskin *et al.* [78] proposed a least-squares minimization in a mapped domain using a local curvilinear coordinate system aligned with the wall. The mapped domain is constructed using the distance function, which is the nearest distance to the boundary, to estimate the solution and gradient at the face mid-points. Although this method produces more accurate gradient for second-order schemes, it does not satisfy conservation of mean within each control volume nor can it be used for higher-order methods. Petrovskaya introduced the concept of numerically distant points to explain the poor accuracy of a weighted LS method on stretched meshes [79]. She also studied quadratic least-squares reconstruction on triangular cells with high aspect ratio and concluded that the weighting of distant stencil points does not lead to more accurate approximation because of numerically distant points in the stencil [80]. More recently, Petrovskaya [81] designed a new approach that allows one to measure the distance between points in the data space instead of the physical space for second-order accuracy and devised a new weighting function that de-emphasizes the points which are far from the original point in the data space.

In this and the next sections, we will investigate the accuracy of high-order solution reconstruction on highly anisotropic meshes with or without curvature for cell-centered discretization along with the conditioning of the least-squares problems. We will show that the condition number of the LS system grows rapidly with mesh refinement and increasing aspect ratio even on meshes without curvature. We will suggest some modification for solving the LS system that improves the conditioning considerably. In addition, our numerical results reinforce the notion that the

cell-centered least-squares reconstruction in its traditional form degrades solution accuracy on high aspect ratio meshes with finite curvature in the sense that the error in the reconstructed coefficients are relatively large compared to the exact values and do not attain the expected order of accuracy. While this has already been proven for second-order methods on triangles, we will demonstrate that the issue remains for higher-order LS reconstruction and adding a weight function does not improve the accuracy. Consequently, a new least-squares reconstruction framework is proposed in which the solution and derivatives are approximated in a local tangential-normal coordinate system obtained by solving an auxiliary LS problem. We will show that the k-exact reconstruction in the new coordinate system improves the accuracy and conditioning significantly for high aspect ratio cells with finite curvature and satisfies the expected order of error reduction with mesh refinement.

3.3.1 Least-Squares Conditioning

The reduced least-squares system of Equation 2.18 can be written in the form of

$$A\,x = b \tag{3.19}$$

where A is the reconstruction matrix after one step of the Gaussian elimination, x is the reconstruction coefficients vector and b is the weighted/unweighted difference of control volume averages of the solution within the stencil. Given an SVD of the reconstruction matrix, the pseudoinverse $A^{\dagger}$ can be obtained

$$A^{\dagger} = V\Sigma^{\dagger}U^{T} \tag{3.20}$$

in which the columns of U and V are the left and right singular vectors of A and $\Sigma^{\dagger}$ is a diagonal matrix containing the reciprocal of singular values of A. Since the reconstruction matrix only includes geometric weights, it does not change between iterations [49]. Therefore, its pseudoinverse is precomputed and stored to yield the reconstruction coefficients by a matrix-vector multiplication

$$x = A^{\dagger}b \tag{3.21}$$

To have an accurate estimate of the reconstruction coefficients, the least-squares

problem should be well-conditioned in the sense that a small error in b (the control volume averages in this case) does not cause a large error in x. This property is typically investigated by the condition number of the reconstruction matrix A (equivalent to the condition number of $A^{\dagger}$) which is defined as the ratio of the maximum to the minimum singular value. The condition number bounds the relative inaccuracy in the estimate of the unknown components. A least-squares problem with a large condition number is said to be ill-conditioned.

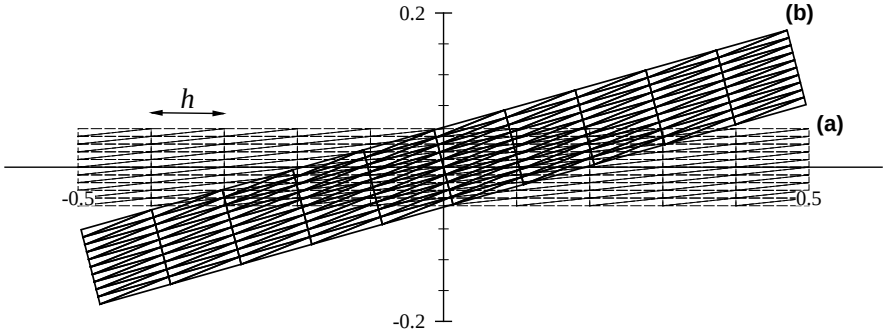
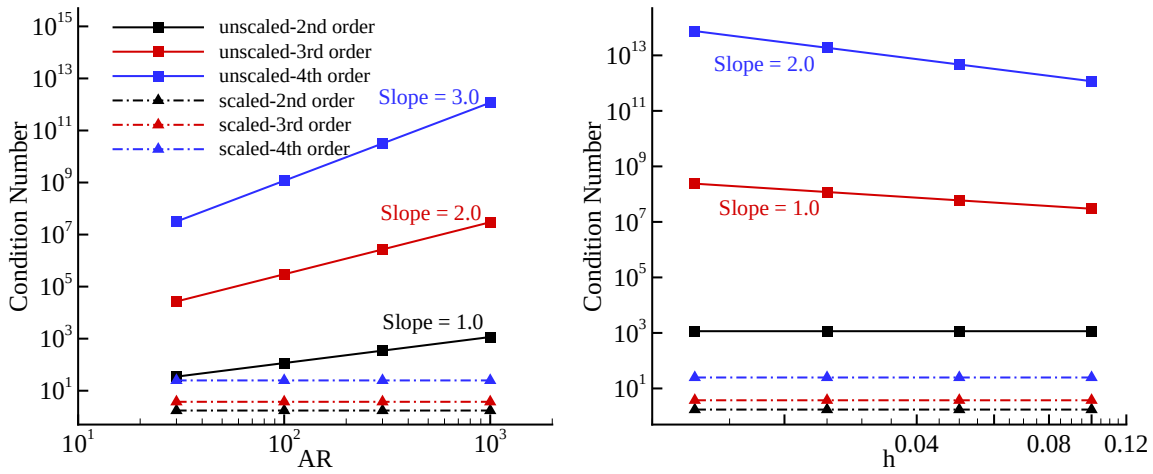


Figure 3.8: Anisotropic triangular meshes with uniform stencil: (a) aligned with the Cartesian coordinate system; (b) rotated 15° in the counter clock-wise direction

As a preliminary test, we consider the least-squares reconstruction on anisotropic triangles obtained from similar splitting of quadrilaterals as shown in Figure 3.8. We employ the unweighted LS method to investigate the effect of aspect ratio and mesh size on the condition number. Note that in such a mesh, the stencil is uniform throughout all interior triangles and thus all control volumes have the same condition number. The columns of the reconstruction matrix of Equation 2.18 contain the moments of area that are scaled by both the size and aspect ratio of the triangular cells. Increasing the aspect ratio of the cells and/or decreasing the mesh spacing results in a huge difference in the order of magnitude of matrix columns that in turn leads to large condition numbers. This difference is asymptotically larger for higher order reconstruction whose matrix includes higher moments of area. This brief discussion suggests the idea of matrix scaling with the largest entry of each column [82] which also was used by Ivan and Groth to improve the conditioning of a central ENO reconstruction for viscous flows [83]. In this way, all the entries of the reconstruction matrix are in the range of $[-1, 1]$ and the conditioning of the least-squares problem improves. Also, the entry with the maximum absolute value

is stored to recover the right reconstruction coefficients later. Figure 3.9 shows how the condition number of the unweighted reconstruction matrix changes with aspect ratio and mesh spacing with/without column scaling for the mesh that is aligned with the Cartesian coordinate system. Our results show that the condition number of matrix A is proportional to AR^{k-1}/h^{k-2} for a k-th order reconstruction. However, the column scaling makes the conditioning independent of the mesh size and aspect ratio.



(a) Effect of AR, $h = 0.1$

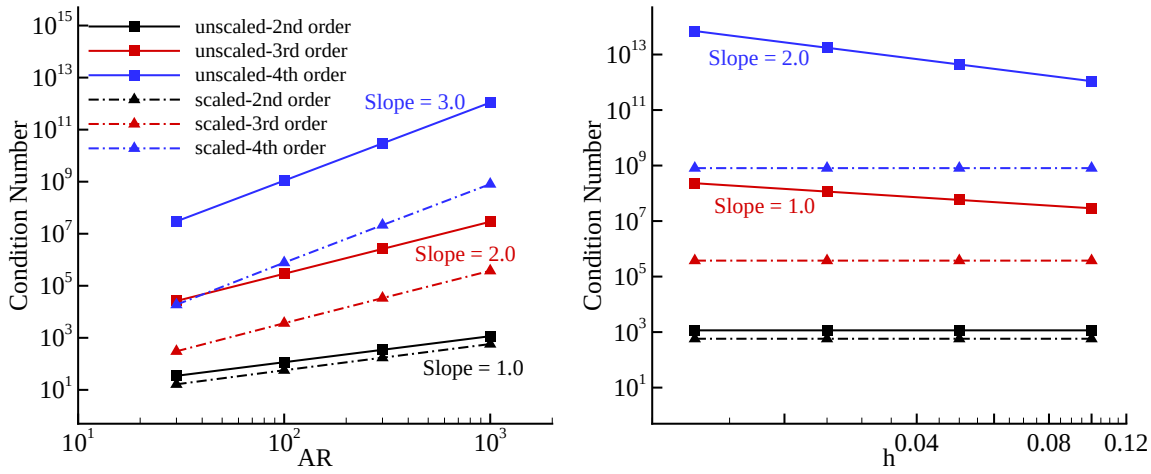
(b) Effect of h, $AR = 1000$

Figure 3.9: Condition number of reconstruction matrix for uniform stencil meshes aligned with the Cartesian coordinate system

Now, we turn our attention to the mesh being rotated 15° in Figure 3.8. For this case, the condition number has been plotted in Figure 3.10 for the same variations of aspect ratios and mesh sizes. As seen, the condition number of the scaled reconstruction matrix still grows with the aspect ratio in a way analogous to the unscaled system although the values are smaller by several orders of magnitude. This is in contrast with the case where the cells were aligned with the Cartesian coordinate system and column scaling led to independence from both the aspect ratio and mesh size. This can be explained by the fact that the Cartesian coordinate system is almost aligned with the principal axes[1] as the aspect ratio increases in that mesh. So the combination of column scaling and principal coordinate system makes the mesh

[1]Principal axes are the two directions around which the second moments of area are minimum and maximum and obtained via Mohr's circle in 2D

isotropic and resolves the conditioning issue for high-order reconstruction on high aspect ratio meshes. It is worth mentioning that this happens just for meshes where the principal axes are almost parallel for the cells within the stencil (Figure 3.8). Nevertheless, for more general cases such as anisotropic meshes over curved surfaces where the orientation of principal axes changes across the stencil, we will show that the combination of column scaling and principal coordinate system is still helpful in reducing the condition number.



(a) Effect of AR, $h = 0.1$

(b) Effect of h, $AR = 1000$

Figure 3.10: Condition number of reconstruction matrix for uniform stencil meshes aligned rotated 15°

3.3.2 Curvilinear Coordinates

As mentioned earlier, the simulation of high Reynolds number turbulent flows requires sufficiently accurate polynomial approximation on high aspect ratio meshes with finite curvature. It is well understood that the cell-centered k-exact LS reconstruction suffers from poor accuracy on highly anisotropic meshes over curved surfaces for second-order. Our numerical results in Section 3.4 yield the same conclusion for higher than second-order reconstruction even after weighting the least-squares system by geometrically close data. This problem is alleviated for those cells that are far from the wall and thus have smaller curvature.

To improve the accuracy of the cell-centered reconstructed gradient on curved meshes with high aspect ratio, the local mapping proposed by Diskin *et al.* [78] is quite inspiring. However, this method is only applicable to second-order accurate reconstruction as it only gives the point-wise first-order face gradient in the curvilinear coordinate system constructed at the middle of each face. This also takes the advantage of the fact that the solution values at the cell centroids are equal to the control volume averages for second-order and the reconstruction matrix is only comprised of Δx and Δy. For higher-order methods, this scheme cannot be incorporated with the moments of area and real control volume averages that differ from the point-wise values at the centroid. Therefore, a higher-order polynomial cannot be reconstructed within a cell to give the solution and the gradient vector.

Instead, we construct a higher-order local curvilinear coordinate system at the reference point of each control volume. For this purpose, we define a mapping from the physical space into a tangential-normal coordinate system for cells with high curvature near the walls:

$$\begin{aligned} t &= a_1 (x - x_i) + a_2 (y - y_i) + a_3 (x - x_i)^2 + a_4 (x - x_i)(y - y_i) + ... \\ n &= b_1 (x - x_i) + b_2 (y - y_i) + b_3 (x - x_i)^2 + b_4 (x - x_i)(y - y_i) + ... \end{aligned} \tag{3.22}$$

A cubic mapping from (x, y) to (t, n) is sufficient for reconstruction up to fourth-order. The mapping is obtained by the distance function and constructed tangential direction at the cell's reference points. The difference in distance from the wall determines the normal coordinate, $n_j^{(i)} = D_j - D_i$, while the tangent coordinate is obtained by the projection of the vector connecting two reference points on the constructed tangential direction, $t_j^{(i)} = \vec{r}_{ij} \cdot \hat{t}_i$. This tangential direction is defined as the perpendicular direction to the normal to the wall direction as seen in Figure 3.11. The two sides of Equation 3.22 are evaluated for the same handful of control volumes used in the reconstruction stencil of a particular control volume. To find the mapping coefficients, an auxiliary least-squares system is solved to give the values of a_i and b_i in Equation 3.22.

It should be noted that the auxiliary least-squares problem will suffer from the same conditioning issue explained before since the moments of area in the reconstruction matrix are replaced by some other geometric terms that have the same order of magnitude and are scaled in an unsatisfactory manner with mesh refinement and

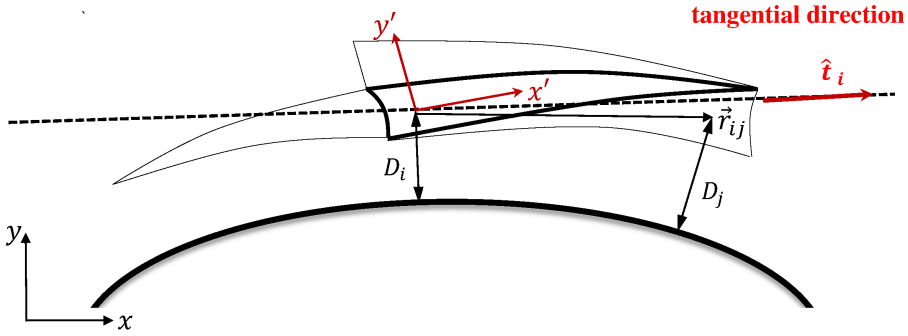


Figure 3.11: Illustration of tangential-normal coordinate construction

increasing aspect ratio. Consequently, the auxiliary LS problem is also brought to the principal coordinate system denoted by (x', y') and its origin is fixed on the cell's reference point (Figure 3.11). The new mapping is defined from (x', y') to (t, n) as

$$\begin{aligned} t &= a'_1 x' + a'_2 y' + a'_3 x'^2 + a'_4 x' y' + ... \\ n &= b'_1 x' + b'_2 y' + b'_3 x'^2 + b'_4 x' y' + ... \end{aligned} \tag{3.23}$$

and column scaling is applied to improve the conditioning.

Having computed the mapping coefficients, it is possible to find the coordinates of any arbitrary point with respect to the local tangential-normal system of a certain control volume by applying the curvilinear mapping of Equation 3.23. For solution reconstruction in the new curvilinear coordinates, we require the moments of control volume i about its reference point:

$$\overline{t^n n^m}_i = \frac{1}{A_i} \int_{V_i} t^n n^m dA \tag{3.24}$$

The integration is performed over the curved cells using the method described in Section 3.2.3. For this purpose, the location of each quadrature point in the physical space (x_q, y_q) is obtained by Equation 3.13 with its corresponding Jacobian of transformation Then, the appropriate rotation transformation is applied to map $(x_q - x_i, y_q - y_i)$ to $\left(x'_q, y'_q\right)$ and finally the non-linear mapping of Equation 3.23 is employed to find (t_q, n_q). This procedure can also be extended to compute the mo-

ments of control volume j about the reference point of control volume i, i.e., $\widehat{t^n n^m}_{ij}$. For these moments, the quadrature points of control volume j must be combined with the curvilinear mapping coefficients of control volume i. Contrary to the reconstruction in the Cartesian coordinate system, these moments need to be calculated one by one since the parallel axis theorem cannot be used for non-parallel local curvilinear coordinates.

The final point about the curvilinear coordinate system is the transformation of solution (or derivatives) to an arbitrary location in the physical space, (x_p, y_p). As described for the quadrature points, it is possible to find the curvilinear coordinates of the desired point, (t_p, n_p). Therefore, the solution can be found as:

$$\begin{aligned} u_p(t,n) &= u|_i + \left.\frac{\partial u}{\partial t}\right|_i t_p + \left.\frac{\partial u}{\partial n}\right|_i n_p + \left.\frac{\partial^2 u}{\partial t^2}\right|_i \frac{t_p^2}{2} \\ &\quad + \left.\frac{\partial^2 u}{\partial t \partial n}\right|_i t_p n_p + \left.\frac{\partial^2 u}{\partial n^2}\right|_i \frac{n_p^2}{2} + ... \end{aligned} \tag{3.25}$$

Similarly, the gradient at any location can be computed in the local coordinate system, $(\partial u/\partial t, \partial u/\partial n)_p$. One can transform the gradient to the Cartesian coordinate system by applying the change of variables and the chain rule:

$$\begin{aligned} \left.\frac{\partial u}{\partial x}\right|_p &= \left.\frac{\partial u}{\partial t}\right|_p \cdot \left.\frac{\partial t}{\partial x'}\right|_p \cdot \left.\frac{\partial x'}{\partial x}\right|_p + \left.\frac{\partial u}{\partial t}\right|_p \cdot \left.\frac{\partial t}{\partial y'}\right|_p \cdot \left.\frac{\partial y'}{\partial x}\right|_p \\ &+ \left.\frac{\partial u}{\partial n}\right|_p \cdot \left.\frac{\partial n}{\partial x'}\right|_p \cdot \left.\frac{\partial x'}{\partial x}\right|_p + \left.\frac{\partial u}{\partial n}\right|_p \cdot \left.\frac{\partial n}{\partial y'}\right|_p \cdot \left.\frac{\partial y'}{\partial x}\right|_p \end{aligned} \tag{3.26}$$

$$\begin{aligned} \left.\frac{\partial u}{\partial y}\right|_p &= \left.\frac{\partial u}{\partial t}\right|_p \cdot \left.\frac{\partial t}{\partial x'}\right|_p \cdot \left.\frac{\partial x'}{\partial y}\right|_p + \left.\frac{\partial u}{\partial t}\right|_p \cdot \left.\frac{\partial t}{\partial y'}\right|_p \cdot \left.\frac{\partial y'}{\partial y}\right|_p \\ &+ \left.\frac{\partial u}{\partial n}\right|_p \cdot \left.\frac{\partial n}{\partial x'}\right|_p \cdot \left.\frac{\partial x'}{\partial y}\right|_p + \left.\frac{\partial u}{\partial n}\right|_p \cdot \left.\frac{\partial n}{\partial y'}\right|_p \cdot \left.\frac{\partial y'}{\partial y}\right|_p \end{aligned}$$

3.4 Numerical Tests

In this section, we present the results for k-exact least-squares reconstruction on anisotropic meshes. First, we show the accuracy and conditioning results corresponding to unweighted and weighted LS reconstructions of an empirical flat plate boundary

layer profile on straight meshes where the variation of principal axis orientation is small between adjacent triangles. Then, we consider anisotropic triangular meshes over curved surfaces. For this purpose, we generate anisotropic triangular meshes over a circular arc and reconstruct a boundary layer type function. The reconstruction procedure both in its traditional form but along principal axes and on the new curvilinear coordinate system is performed and the results are compared. Then, we extend the analysis to more general meshes by testing reconstruction scenarios on the anisotropic boundary layer region over the NACA 0012 airfoil.

3.4.1 Straight Meshes

We start the analysis with anisotropic straight meshes where the alignment changes slowly when moving from one triangle to another. These meshes are typically used for anisotropic flows over non-curved surfaces on which a boundary layer develops (e.g., flat plate) and/or wake regions behind single objects. Anisotropic meshes over curved surfaces can fall in this category provided that the radius of curvature is considerably larger than the cell spacing.

To investigate the accuracy of reconstructed derivatives, the method of manufactured solution is employed. Considering that anisotropic meshes are used to capture solution anisotropy, an anisotropic function must be manufactured for reconstruction tests. One of the obvious candidates for such a function is the turbulent boundary layer velocity profile. In this paper, we make use of Reichardt's empirical boundary layer profile [84] :

$$u = \left\{ 2.5 \, \ln\left(1 + 0.4\, y^+\right) + 7.8 \, \left(1 - \exp\left(\frac{-y^+}{11}\right) - \frac{y^+}{11} \, \exp\left(-0.33\, y^+\right)\right)\right\} u^* \quad (3.27)$$

where y^+ is the non-dimensional distance from the wall and u^* is the friction velocity:

$$y^+ = \frac{y u^*}{\nu} \quad u^* = \sqrt{\frac{\tau_w}{\rho}} \quad (3.28)$$

In Equation 3.28, τ_w is the wall shear stress which can be related to the friction factor

$$C_{f,x} = \frac{\tau_w}{\frac{1}{2}\rho U_\infty^2} \quad (3.29)$$

and approximated by the one-seventh power law:

$$C_{f,x} = \frac{0.027}{(Re_x)^{1/7}} \ , \ Re_x = \frac{U_\infty x}{\nu} \tag{3.30}$$

Using Equations 3.28 to 3.30, it is possible to simplify u^* and y^+ :

$$u^* = \sqrt{\frac{0.0135}{(Re_L.\bar{x})^{1/7}}} \ , \ y^+ = \bar{y}.Re_L.\sqrt{\frac{0.0135}{(Re_L.\bar{x})^{1/7}}} \tag{3.31}$$

where $\bar{x}$ and $\bar{y}$ are the non-dimensional coordinates based on the length of plate, L, for a plate whose leading edge is at the origin. We assume that $L = 1$ and take $Re_L = 10^7$ to provide highly anisotropic behavior for $\bar{u} = u/U_\infty$ near the wall. Note that u^* and y^+ are both singular at $x = 0$ and thus the region close to the leading edge must be avoided. For our reconstruction tests, we assume that the leading edge is at $(-1, 0)$ and only consider the zone between $x = -0.5$ and $x = 0.5$. Therefore, $\bar{x}$ is replaced with $x+1$ in Equation 3.31. The grid is generated with $(N+1)\times(2N+1)$ nodes which are uniformly distributed along the plate but stretched out from the wall with a factor of s. The thickness of the first layer is set such that one control volume exists in the viscous sublayer ($y^+ < 5$) for the coarsest mesh; this is shrunk by a factor of two at each level of refinement. The quadrilaterals formed by these nodes are randomly divided into two triangles and nodes are perturbed in both directions with a factor of rh_y where r is a random number in $[-0.25, 0.25]$ and h_y is local vertical spacing. We consider three different mesh sizes in which $N = 8$, 16, 32 and $s = 1.6$, 1.25, 1.12, respectively. It is worth mentioning that for all these meshes, the maximum aspect ratio is about $10,000$. This value is obtained by considering the number of divisions in the horizontal direction and the necessity of having one cell in the viscous sublayer region for the coarsest mesh. Figure 3.12 shows the anisotropic solution on a randomly triangulated mesh where $N = 16$ both in a far view and very close to the wall.

To assess the accuracy of reconstruction, the L_2-norm of error in the reconstructed solution and first derivatives at the reference point of each triangle is considered for both unweighted and weighted least-squares systems. In all these cases, the reconstruction has been performed along the principal axes of each control volume to improve the conditioning of the least-squares problem. In this framework, the deriva-

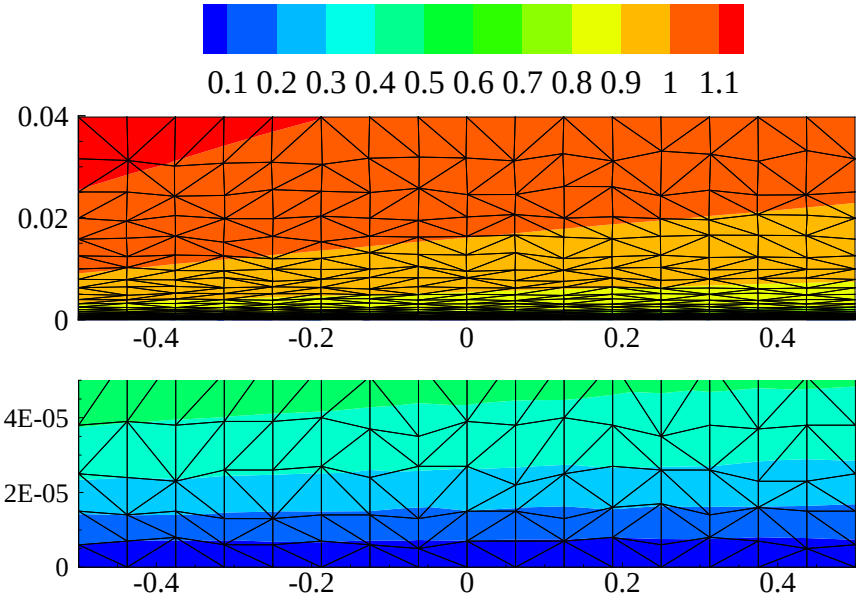
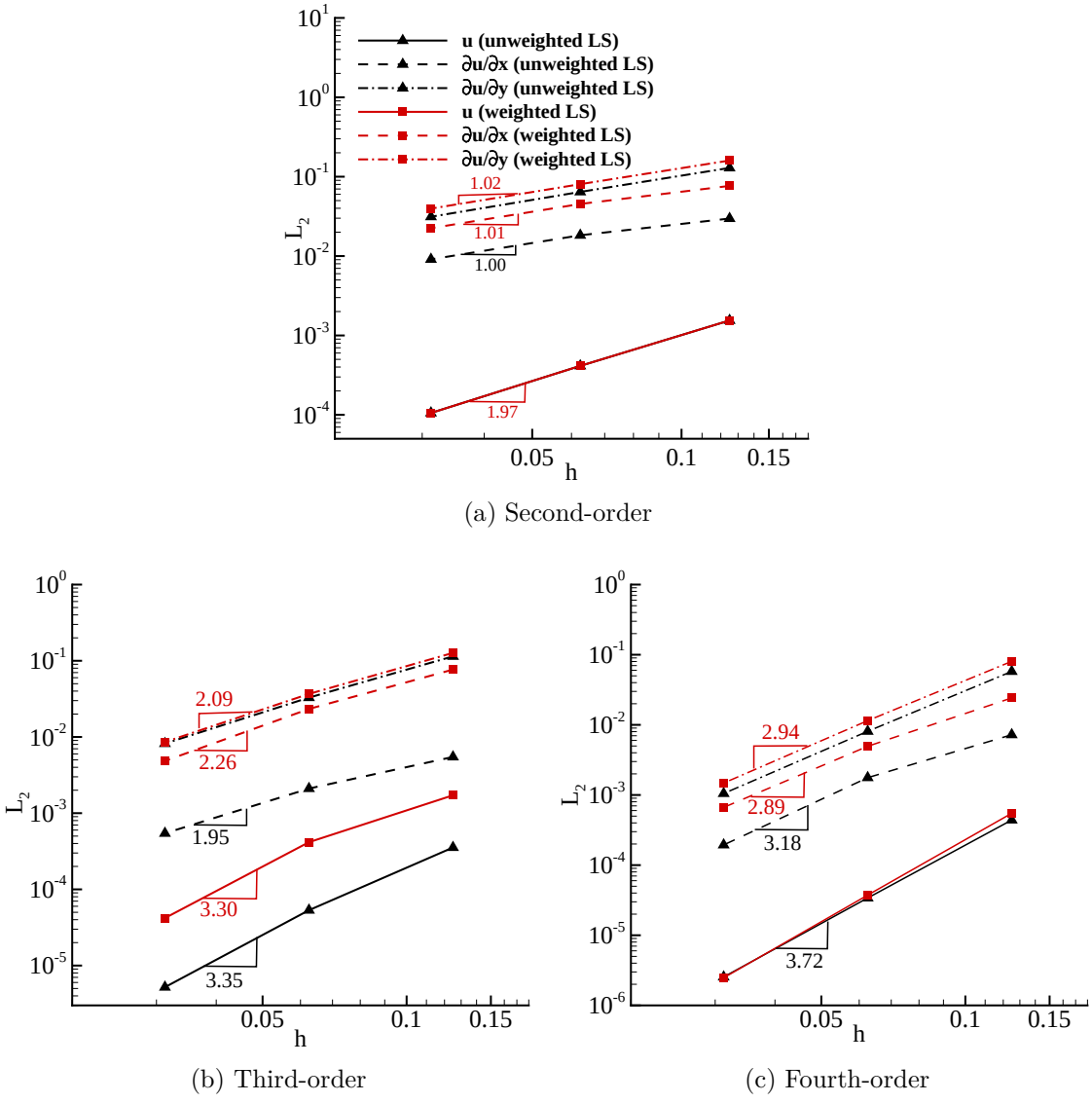


Figure 3.12: Anisotropic manufactured solution for a straight mesh (y-axis has been scaled)

tives have been brought to the Cartesian coordinate system and compared against the exact derivatives obtained from the manufactured solution. Figure 3.13 shows asymptotic error convergence for second-, third- and fourth-order solution reconstruction. In all cases, the order of error in the reconstructed solution matches the expected order of reconstruction. Likewise, the reconstructed derivatives are one order less accurate than the solution, as expected. Note that the $y-$derivatives are larger due to the anisotropic property of the test function and thus their relative errors are reported to be comparable with the $x-$derivatives. As seen in this figure, the difference in solution error is less noticeable between weighted (red) and unweighted (black) for second- and fourth-order but is about one order of magnitude for third-order reconstruction. Moreover, the unweighted LS yields more accurate derivatives particularly in the $x-$direction for all orders of accuracy. These results suggest that the unweighted LS provides more accurate reconstructed values for anisotropic straight meshes.

In addition to error norms, the local accuracy of reconstruction is important. Figure 3.14 compares the solutions and y-derivatives obtained by second- and fourth-order unweighted LS reconstruction with the exact values at different distances from the wall at $x = 0.1$. As expected, the reconstructed values match well with the exact profiles even at small distances in which the triangles are highly skewed. Also,



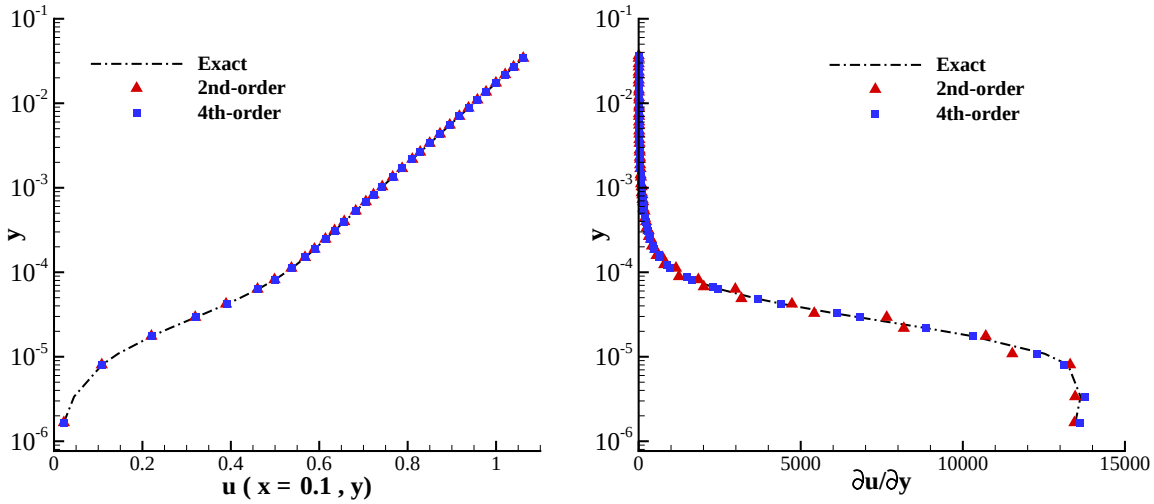
(a) Second-order

(b) Third-order

(c) Fourth-order

Figure 3.13: Accuracy of LS reconstruction for anisotropic straight meshes

the error is reduced by increasing the order of accuracy. This can be more clearly noticed for the y-derivatives that are one order less accurate and more scattered in magnitude.



(a) Solution (b) y-derivative

Figure 3.14: Reconstructed vs. exact values at different distances from the wall ($x = 0.1$)

To address the conditioning of the LS reconstruction for straight meshes, we consider the maximum condition number, $\kappa_\infty(A)$, corresponding to the least-squares systems. This is given by Table 3.2 for different conditions. In this table, $AR_{\max}$ refers to the maximum aspect ratio which occurs in the first layer. Increasing Re_L in the solution reduces the thickness of viscous sub-layer and thus the first interior layer which increases the maximum aspect ratio in the mesh. Clearly, column scaling improves the conditioning of the LS system in all cases. Also, the unweighted system after column scaling exhibits remarkably better conditioning for higher than second-order as it leads to considerably smaller condition numbers for similar meshes and the condition number remains almost constant when changing the maximum aspect ratio and/or mesh size. On the other hand, the weighted system even after column scaling produces condition numbers proportional to aspect ratio for third- and fourth-order reconstructions. This result implies more accurate reconstruction at extremely anisotropic meshes using unweighted LS combined with column scaling.

$AR_{\max}$	h	Col. Scale	$\kappa_\infty(A)$		
			$k=2$	$k=3$	$k=4$
unweighted LS					
3.15×10^3	0.0625	yes	2.21	13.78	107.12
9.65×10^3	0.125	yes	2.00	16.69	107.46
9.65×10^3	0.0625	yes	2.20	17.35	106.77
9.65×10^3	0.0625	no	1.43×10^4	1.44×10^{10}	6.19×10^{15}
9.65×10^3	0.03125	yes	2.39	15.55	104.71
2.67×10^4	0.0625	yes	2.21	15.53	95.52
2.67×10^4	0.0625	no	4.07×10^4	1.26×10^{11}	1.39×10^{17}
weighted LS					
3.15×10^3	0.0625	yes	2.74	1.41×10^4	6.48×10^4
3.15×10^3	0.03125	yes	2.76	1.43×10^4	6.83×10^4
9.65×10^3	0.0625	yes	2.78	4.28×10^4	2.07×10^5
9.65×10^3	0.0625	no	8.10×10^3	5.96×10^9	2.67×10^{15}
2.67×10^4	0.03125	yes	2.76	1.27×10^5	6.20×10^5
2.67×10^4	0.03125	no	2.48×10^4	1.17×10^{11}	3.26×10^{17}

Table 3.2: Maximum condition number for reconstruction along principal axes on straight meshes

3.4.2 Curved Meshes

Now, we focus on meshes varying anisotropically over curved surfaces. For these meshes, the alignment varies considerably between adjacent triangles. This is usually encountered in high-Reynolds turbulent flow simulations over curved walls on which a boundary layer exists and/or wake regions of upstream objects in multi-element configurations.

As the first step for testing different reconstruction scenarios, we generate anisotropic cells over a circular arc and reconstruct a boundary layer type function. For this purpose, we use the boundary layer profile of Equation 3.27 and replace y with distance from the wall and x with the horizontal distance from the leading edge. Even though this function does not give the velocity profile over a cylinder with stream-wise pressure gradient, it has similar characteristics and exhibits strong normal gradients, and vanishing stream-wise gradients. To prevent singularities given by Equation 3.27

at the cylinder's leading edge, we consider a 40° circular arc whose initial edge is sufficiently away from the leading edge, $\theta = -\pi/2$ (Figure 3.15). Again, we use $(N+1) \times (2N+1)$ nodes with the aforementioned values for N and diagonalize quadrilaterals randomly (in the case of triangular meshes) along with a node perturbation based on local radial spacing. Note that the nodes are placed with uniform spacing along the arc but are stretched towards the wall with the stretching factors already mentioned for straight meshes. We still ensure the presence of at least one control volume in the region where y^+ based on distance from the wall is less than 5 and set $Re_L = 10^7$. Figure 3.15 shows the anisotropic solution and mesh where $AR_{\max} \simeq 6320$, $N = 16$ and $s = 1.25$.

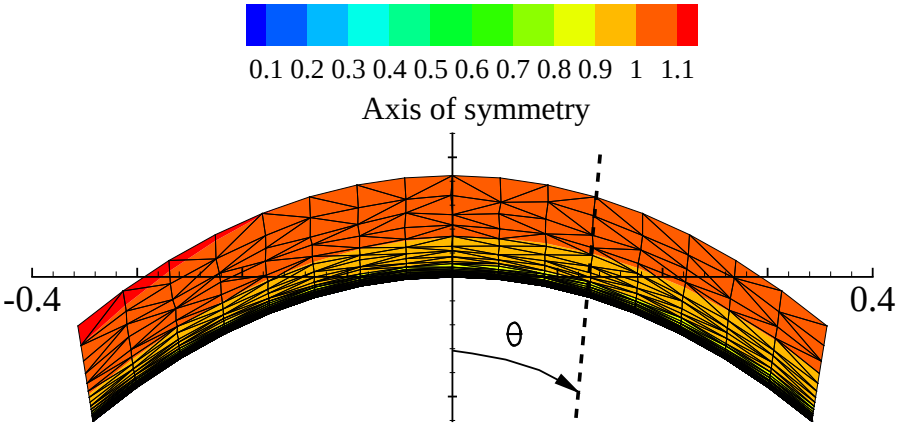


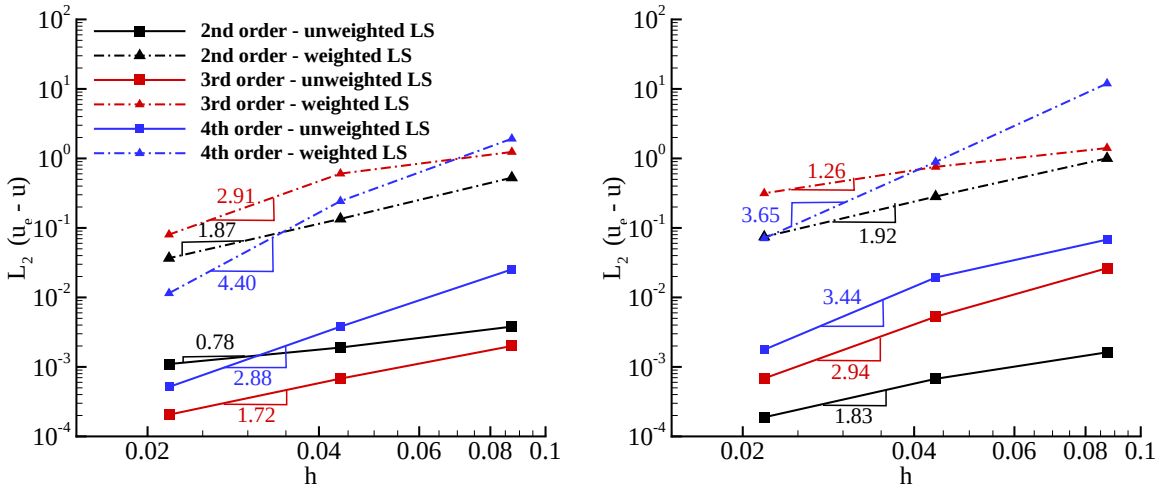
Figure 3.15: Anisotropic boundary layer type solution on a circular arc

Having manufactured an appropriate solution, we examine the accuracy of higher-order reconstruction in its traditional form described in Chapter 2 but using the principal coordinate system to make the conditioning better. For the results shown in this section, the boundary faces on the circular arc are curved using the method described in Section 3.1 which is accurate up to fourth-order. Figure 3.16 shows the L_2 norm of solution reconstruction error for the manufactured function of Figure 3.15 and the three mesh sizes described earlier for both the triangular and quadrilateral meshes.

For triangles, it is seen that the asymptotic order of error in the unweighted LS is not consistent with the order of reconstruction in the range of mesh sizes considered here. Even though there is a chance of achieving the order expected in reconstruction with more levels of refinement, this negates the advantage of higher-

order computations. Weighting the least-squares system corrects the asymptotic order of error but at the same time is associated with a noticeable increase in the value of error. Comparing the error values with the solution magnitude illustrated in Figure 3.15 implies a large relative error which is not acceptable.

For quadrilateral meshes, the unweighted LS delivers the right asymptotic order of convergence. However, increasing the order of accuracy degrades the accuracy of reconstructed solution at the cell reference points. Likewise, adding the distance weight to the LS system increases the magnitudes of reconstruction error. Therefore, it is accurate to say that traditional higher-order least-squares reconstruction suffers from accuracy issues for anisotropic meshes over curved surface regardless of the weighting function and cell type.



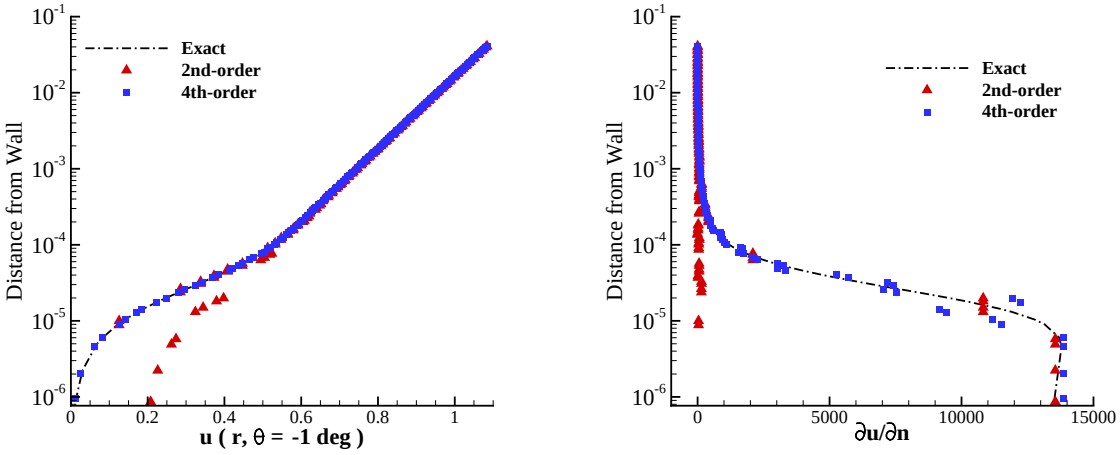
(a) Triangular meshes

(b) Quadrilateral meshes

Figure 3.16: Solution reconstruction error using principal coordinate system

To shed more light on this issue, the reconstructed solution and normal derivatives on a fixed position across the arc, $\theta = -1°$, have been plotted against the exact values in Figure 3.17 for the second triangular mesh. These are computed at different distances from the wall for a weighted LS reconstruction (Figure 3.16). At small distances from the wall where high aspect ratio triangles exist, the second-order approximation of solution and corresponding normal derivatives are highly off the exact solution in some locations. This can be explained by the lack of sufficiently close control volumes to the cell center under consideration as Mavriplis showed in his

work [77] for second-order reconstruction on triangular meshes. For fourth-order, the stencil becomes larger and includes geometrically close control volumes that improve the estimate of normal derivatives; however, the reconstructed values are notably off the exact curve for the weighted LS reconstruction.

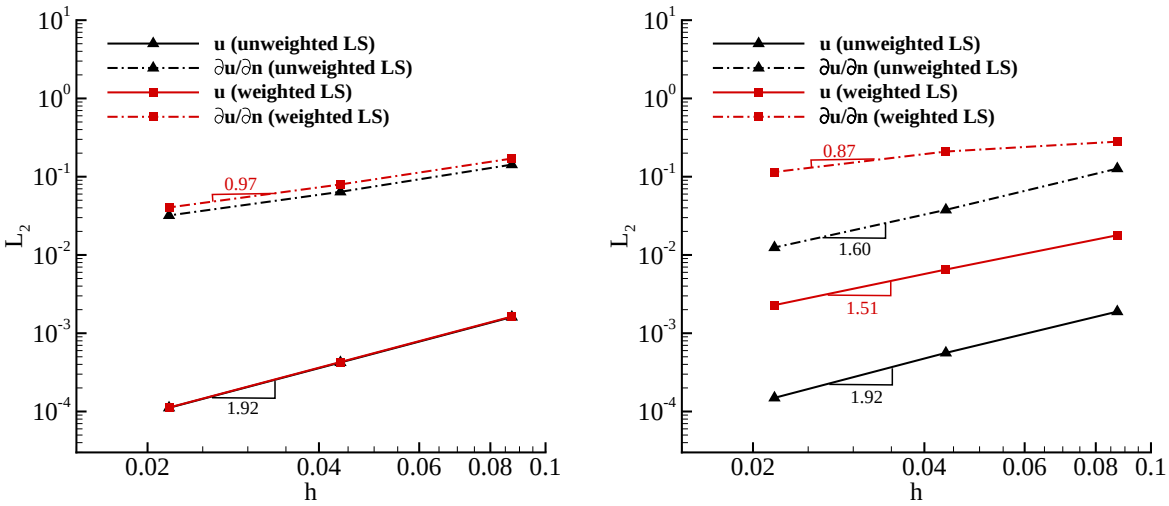


(a) Solution (b) Normal derivative

Figure 3.17: Weighted LS reconstructed values in the principal coordinate system against exact values at different distances from the wall ($\theta = -1°$)

On the other hand, the asymptotic order of reconstruction error using the new curvilinear coordinate system is consistent with the order of reconstruction and the error values are smaller by several orders of magnitude compared to the principal coordinates. Figure 3.18 shows the reconstruction error for second- to fourth-order reconstruction in the tangential-normal coordinate system for triangular and quadrilateral cells. Also, the first derivative in the normal direction is one order less accurate than the solution, as expected. For both types of cells, the unweighted LS produces more accurate reconstruction coefficients, particularly for quadrilateral cells where the difference in the error of reconstructed solution is more than one order of magnitude in some cases. For normal derivatives, the differences between weighted and unweighted LS become smaller by increasing the order of reconstruction although the unweighted LS outperforms in all of the cases. In general, the unweighted LS reconstruction in the curvilinear coordinate system solves the accuracy issues related to high aspect ratio meshes in the presence of curvature.

Again, the solution and normal derivatives reconstructed by unweighted LS on the curvilinear coordinate system at $\theta = -1°$ are plotted against the exact values

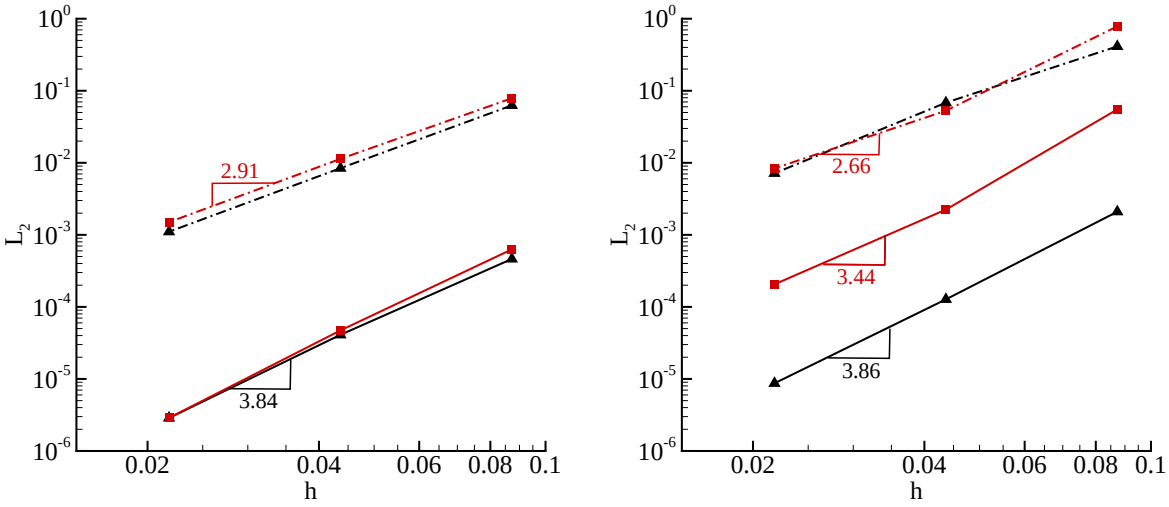


(a) Second-order/triangles

(b) Second-order/quads

(c) Third-order/triangles

(d) Third-order/quads



(e) Fourth-order/triangles

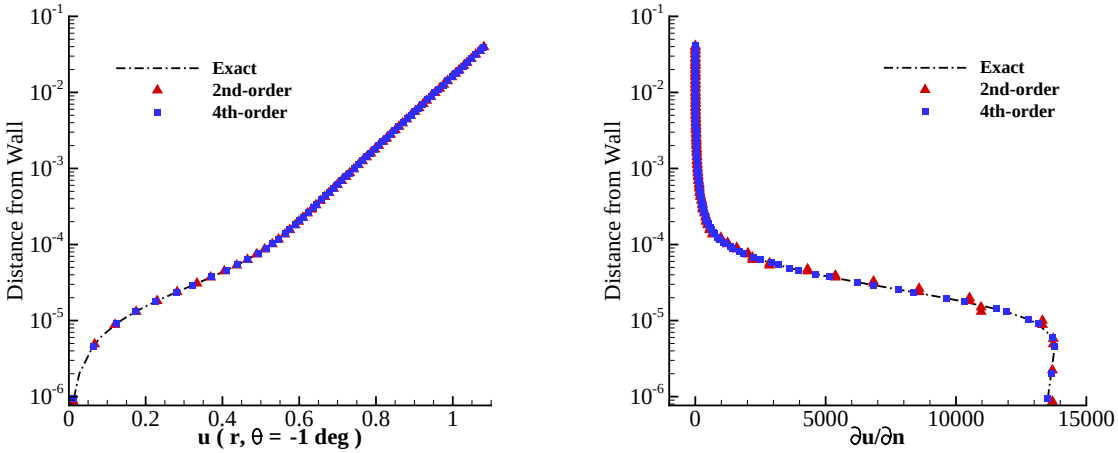
(f) Fourth-order/quads

Figure 3.18: Reconstruction error (relative for normal derivatives) using curvilinear $t-n$ coordinate system

on the second triangular mesh (Figure 3.19). In contrast to the results obtained by reconstruction on principal coordinates, both the solution and normal derivatives are reasonably close to the exact values and become more accurate by increasing the order of accuracy.

Now, we turn our attention to the conditioning of the least-squares reconstruction in each coordinate system. Table 3.3 gives the maximum condition number of LS system for reconstruction along principal or Cartesian coordinates for triangular meshes. Note that "no" in principal coordinates column implies the traditional reconstruction in the Cartesian coordinate system. Clearly, the use of principal coordinates and column scaling highly enhance the conditioning as they reduce $\kappa_\infty(A)$ particularly for higher orders by several orders of magnitude. Even though the combination of these two does not make the condition number independent of mesh properties in this case, it is highly recommended even for the auxiliary LS problem having the same features.

Having this in mind, one can compare the maximum condition numbers for the scaled system in principal coordinates for different $AR_{\max}$ and h. Second-order reconstruction is always well-conditioned regardless of weight function and mesh properties. For higher-order, the unweighted LS gives the condition number being scaled linearly



(a) Solution (b) Normal derivative

Figure 3.19: Reconstructed values in the curvilinear coordinate system against exact values at different distances from the wall ($\theta = -1^{\circ}$)

with mesh spacing at a constant aspect ratio whereas the changes are smaller for the weighted system. Likewise, the changes with $AR_{\max}$ at a constant h are more severe for the unweighted LS. In general, weighting gives smaller condition numbers for higher-order on curved meshes; however, it does not guarantee a perfect conditioning particularly for fourth-order reconstruction on high aspect ratio meshes.

On the other hand, the unweighted LS along the curvilinear coordinate system is always well-conditioned as shown by Table 3.4. Based on the conditioning results reported so far, we know that column scaling is essential for higher-order reconstruction on anisotropic meshes. Therefore, we just consider the maximum condition number for the scaled LS system. The conditioning results for the curvilinear coordinates are completely similar to those obtained for principal axes on straight meshes. The condition numbers become independent of mesh properties for all orders of accuracy for the unweighted system. On the other hand, they are scaled with $AR_{\max}$ but remain independent of h for the weighted case. As a result, the unweighted LS on the curvilinear coordinates outperforms other schemes for anisotropic curved meshes in terms of both accuracy and conditioning. Note that the same type of conclusion can be drawn for the conditioning of solution reconstruction on quadrilateral meshes although we skip their results for brevity.

$AR_{\max}$	h	Col. Scale	Princ. Coord.	$\kappa_\infty(A)$		
				$k=2$	$k=3$	$k=4$
unweighted LS						
2.06×10^3	0.0436	yes	yes	2.43	254.94	3.46×10^3
2.06×10^3	0.0436	yes	no	54.53	5.46×10^3	2.81×10^6
2.06×10^3	0.0218	yes	yes	2.43	135.15	1.88×10^3
2.06×10^3	0.0218	no	yes	193.84	6.14×10^6	4.57×10^{11}
6.32×10^3	0.0436	yes	yes	4.98	842.41	1.05×10^4
6.32×10^3	0.0436	yes	no	79.49	5.61×10^3	8.36×10^6
6.32×10^3	0.0436	no	no	158.90	8.02×10^5	6.84×10^{10}
1.75×10^4	0.0218	yes	yes	2.44	1.15×10^3	1.56×10^4
1.75×10^4	0.0218	yes	no	117.13	2.55×10^4	1.56×10^8
weighted LS						
2.06×10^3	0.0436	yes	yes	2.21	30.65	561.58
2.06×10^3	0.0218	yes	yes	2.22	42.88	501.29
6.32×10^3	0.0872	yes	yes	15.50	53.99	6.64×10^3
6.32×10^3	0.0436	yes	yes	17.35	162.56	3.81×10^3
6.32×10^3	0.0436	yes	no	55.85	3.53×10^3	2.25×10^6
6.32×10^3	0.0436	no	yes	119.23	2.42×10^6	4.35×10^{10}
6.32×10^3	0.0218	yes	yes	13.70	399.98	2.74×10^3
1.75×10^4	0.0436	yes	yes	2.22	48.93	4.77×10^3
1.75×10^4	0.0436	yes	no	53.69	3.00×10^3	6.52×10^6

Table 3.3: Maximum condition number for reconstruction along principal(Cartesian) axes on curved meshes

3.4.3 General Meshes

As a more general case for anisotropic meshes over curved surfaces, we use the structured grids generated for turbulent RANS simulations by NASA Langley Research Center [85]. For experiments on triangular meshes, we also randomly triangulate the quadrilaterals (Figure 3.20). We curve the interior faces of the mesh to fourth-order using a cubic spline representation of the airfoil surface. We only consider the unweighted LS because of the conditioning issues discussed earlier.

The meshes typically used for viscous flow simulations are comprised of different regions: anisotropic cells with curvature very close to solid walls (R1), anisotropic

AR_{max}	h	column scaling	$\kappa_\infty(A)$		
			$k=2$	$k=3$	$k=4$
unweighted LS					
2.06×10^3	0.0436	yes	1.74	13.99	101.81
2.06×10^3	0.0218	yes	1.74	13.86	97.48
6.32×10^3	0.0872	yes	2.06	13.58	91.49
6.32×10^3	0.0436	yes	2.23	15.61	97.17
6.32×10^3	0.0218	yes	2.31	17.47	105.36
1.75×10^4	0.0218	yes	1.73	13.84	97.43
weighted LS					
2.06×10^3	0.0436	yes	2.21	4.85×10^3	4.17×10^4
2.06×10^3	0.0218	yes	2.22	5.29×10^3	4.50×10^4
6.32×10^3	0.0436	yes	2.20	2.86×10^4	1.51×10^5
1.75×10^4	0.0436	yes	2.22	4.07×10^4	3.53×10^5

Table 3.4: Maximum condition number for reconstruction along local curvilinear axes on curved meshes

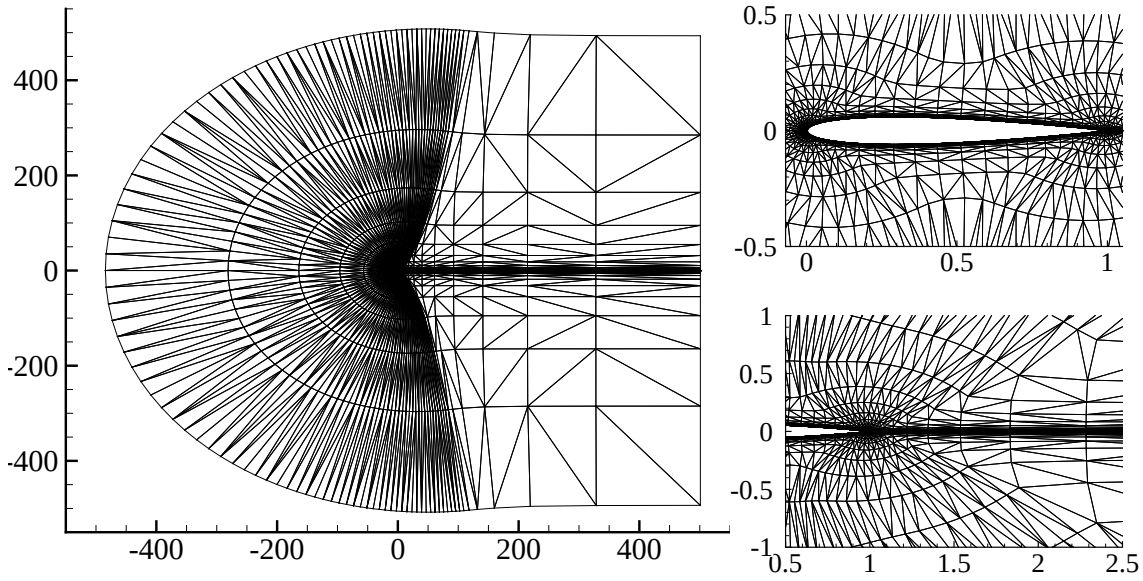


Figure 3.20: Unstructured triangular mesh over NACA 0012

straight cells often seen in wake regions (R2) and isotropic cells sufficiently far from the walls (R3). Since the calculation of mapping coefficients and moments of area are computationally more expensive for the curvilinear coordinate system, we need to isolate the regions that require this treatment. We already know that higher-order least-squares reconstruction even in Cartesian coordinate system provides accurate results along with good conditioning for isotropic cells. In addition, our previous results for anisotropic straight meshes suggest a principal coordinate system and column scaling for accurate and well-conditioned LS reconstruction. As a results, only cells with high aspect ratio and significant curvature demand curvilinear coordinates. Such cells can be recognized as those that have a large aspect ratio ($AR > 10$ here) and are fairly well aligned with a wall. The latter can be understood by the angle between the line that connects the reference location to the closest point on the wall and the principal axis around which the second moment of area is minimum ($\gamma < 10°$). Note that the cells in R2 only satisfy the first criterion. Figure 3.21 shows the separation of regions for NACA 0012 meshes using the mentioned procedure.

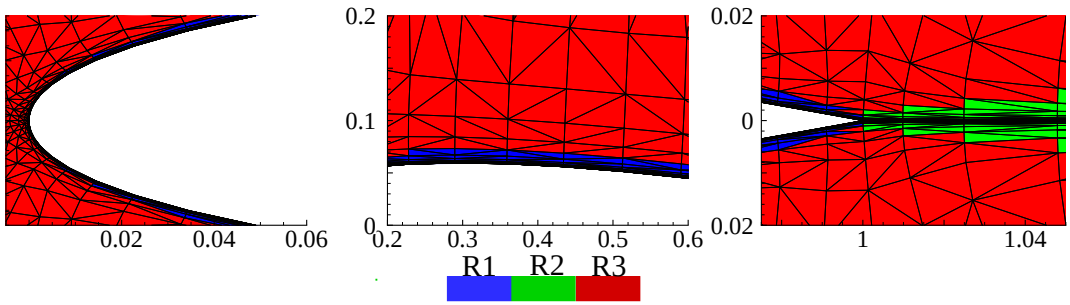


Figure 3.21: Separate regions for NACA 0012 meshes

For reconstruction tests, we just consider R1 colored by blue because that is the region where traditional LS suffers from accuracy issues. Note that the cells in R2 are identical to straight meshes we considered in Section 3.4.1. To prove that the improvements in LS reconstruction on curvilinear coordinate system is not limited only to simple geometries, we perform our accuracy tests on the cells of R1 for the unstructured meshes around the NACA 0012. For this purpose, an anisotropic solution is manufactured for this region and some layers of neighbors in other regions. We re-employ the boundary layer profile of Equation 3.27 for the cells with a reference location in $-0.1 < y < 0.1$ and $x > -0.15$ and initialize other control volume averages with zero. In Equation 3.31, $\bar{y}$ is replaced with distance from the wall for the cells

near the wall with a reference location in $0 < x < 1$ and $\bar{x}$ is replaced with $x + 0.2$ to prevent singularities near the leading edge:

$$u_i^e(x,y) = \left\{2.5\,\ln\left(1+0.4\,y^+\right) + 7.8\left(1-\exp\left(\frac{-y^+}{11}\right) - \frac{y^+}{11}\exp\left(-0.33\,y^+\right)\right)\right\} u^*$$
$$u^* = \sqrt{\frac{0.0135}{(Re_L.(x+0.2))^{1/7}}},\ y^+ = d.Re_L.u^* \tag{3.32}$$

Also, $Re_L = 6 \times 10^6$ as this is the Reynolds number for which numerical results have been reported in Ref. [85] and thus matches the anisotropy in the mesh. Figure 3.22 illustrates the manufactured solution at different locations near the wall on the coarsest mesh.

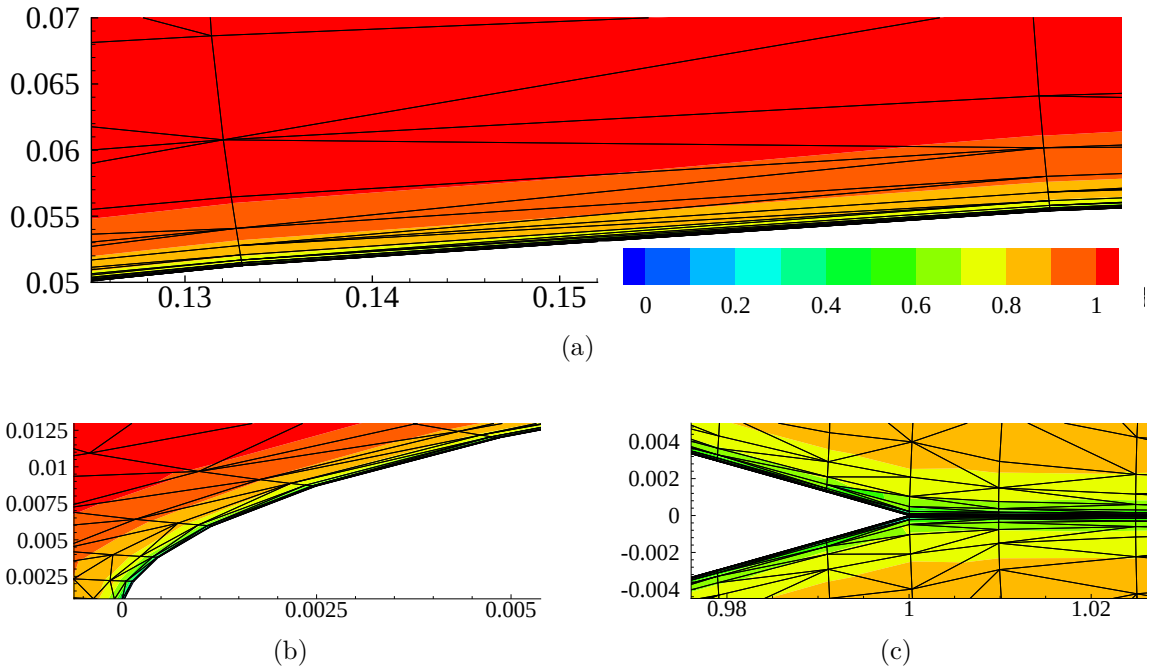
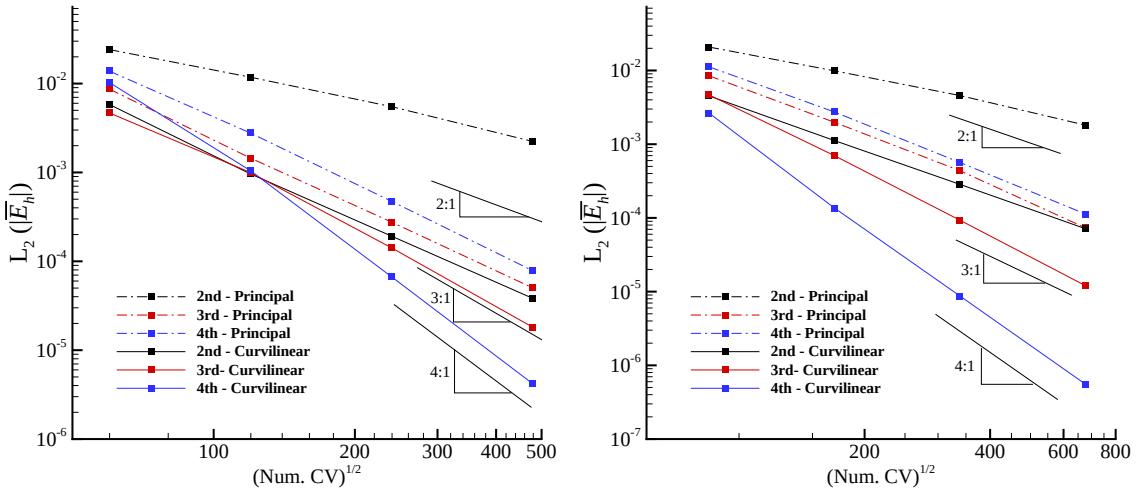


Figure 3.22: Manufactured solution for reconstruction accuracy test on NACA 0012

The accuracy test is performed on a sequence of quadrilateral and triangular meshes. For this purpose, the average of the absolute value of the reconstruction error, $\bar{E}_h$, is calculated over each control volume as:

$$\left|\bar{E}_h\right|_i = \frac{1}{A_{CV_i}} \int\int_{CV_i} \left|u_i^R(x,y) - u_i^e(x,y)\right| dA \tag{3.33}$$



(a) Quadrilateral cells (b) Triangular cells

Figure 3.23: Error norms of reconstructed solution on anisotropic cells of NACA 0012

where A_{CV_i} is the surface are for control volume i, $u_i^R(x,y)$ is the reconstructed solution profile in the control volume and $u_i^e(x,y)$ is the exact solution profile obtained by Equation 3.32 at each point of the control volume. The integral of Equation 3.33 is computed by employing a sixth-order quadrature rule and evaluating the difference between exact and reconstructed values at the quadrature points of each control volume.

One can compare the error norms obtained for unweighted LS reconstruction using principal and curvilinear coordinate systems. Using the local curvilinear coordinate system leads to a significant improvement in the accuracy of reconstructed solutions as seen in Figure 3.23. The error values become considerably smaller for both the quad and triangular anisotropic cells and the nominal order of reconstruction is obtained by mesh refinement. On the other hand, higher than second-order reconstruction yields slightly better than second-order reconstructed solutions using principal coordinates. It is clear that the approximation of the flux vector and source term with low-order reconstructed solutions cannot lead to a higher-order accurate solution at the end. Although this improvement was shown only for the reconstruction of a particular manufactured solution, its advantage in evaluating more accurate solutions and outputs will be shown later in the next chapter.

Chapter 4

RANS Simulation of Turbulent Flows

With the success of Discontinuous Galerkin (DG) methods in obtaining more accurate solutions in computational aerodynamics, more realistic flow conditions such as turbulent flows governed by the Reynolds Averaged Navier-Stokes (RANS) equations have been tackled in recent years. The earliest success in this area was the implicit higher-order RANS solver of Bassi et al. [86] based on the $k-\omega$ turbulence model in which the solutions of turbulent flows on a flat plate and turbine blades were investigated. Later, Nguyen *et al.* [87] used the one equation turbulence model of Spalart-Allmaras (SA) in their higher-order DG solver for the solution of turbulent flat plate and 2D airfoil test cases. In recent years, a great deal of effort has been devoted to increase the efficiency and robustness of higher-order DG RANS solvers [88, 89, 90] and also to expand the capability of solving turbulent flow problems with more complex geometries [91]. In addition to these, some other discretization methods such as streamline upwind/Petrov Galerkin (SUPG) [92] or residual distribution (RD) [93] schemes have been used to deliver a higher-order solution of RANS equations. These methods are less expensive as the solution is assumed to be continuous across the interfaces and thus the number of degrees of freedom grows less rapidly when increasing the order of polynomial.

On the other hand, the higher-order finite volume methods, which have shown their ability for efficient computations of inviscid and viscous laminar flow with promising accuracy on irregular meshes, have not been used for the solution of turbulent compressible flows. In this chapter, we describe the remaining challenges encountered in the extension of higher-order unstructured finite volume methods to RANS simulations and also the treatments required to tackle each of these issues. This chapter presents the development of an implicit higher-order unstructured finite volume solver for turbulent aerodynamic flows.

One of the challenges in the application of higher-order methods to RANS simulations with the Spalart-Allmaras model is the abrupt change in the slope of the turbulence working variable at the edge of boundary layer. This behavior results in negative values of the turbulence working variable and ultimately causes solver failure. Several modifications [88, 94] have been proposed for this model to make it benign for higher-order discretizations across the slope discontinuity. In the present work, we use the SA-neg model developed by Allmaras *et al.* [95]. We also fully couple the turbulence model equation with the mean flow equations and the RANS-SA system is considered as a complete system of equations.

Another important task in a higher-order flow solver is robust and efficient convergence to the steady-state solution. For a turbulent flow simulation, anisotropic cells induce significant stiffness into the discrete equations and hamper solution convergence. Consequently, special attention should be paid to the solution strategy for these problems. In this work, we adopt a solution strategy originally developed for a higher-order DG RANS solver [90], with minor modifications.

An outline of this chapter is as follows. In Section 4.1, the governing equations including the negative variant of the SA model are described. We present the numerical flux functions used for the fully-coupled RANS-SA system in Section 2.3. Section 4.3 briefly reviews the solution strategy used for the steady-state solution of turbulent flows. Four numerical examples, including a turbulent flat plate, subsonic and transonic airfoils, and finally a multi-element configuration are presented in Section 4.4 to highlight the ability of higher-order methods to obtain a more accurate solution on coarser meshes and also efficient convergence to the steady-state solution.

4.1 Governing Equations

In this work, the RANS equations are coupled to the one equation turbulence model of Spalart and Allmaras (SA model) [19]. The original SA model admits only non-negative values of the working variable. Recent studies in the context of higher-order DG methods have found difficulties in the robust application of the Spalart-Allmaras model due to non-smooth solution behavior of this turbulence model at the edge of the flow boundary layer [89]. This behavior results in negative values of the SA working variable and ultimately causes solver failure. These negative values are generated by Gibbs phenomena that stem from employing high-order approximations across a

discontinuity in slope. Alternatively, we use the negative variant of the SA model proposed by Allmaras *et al.* [95]:

$$\frac{\partial \rho\tilde{\nu}}{\partial t} + \nabla \cdot \left(\rho\tilde{\nu}\vec{V}\right) - \nabla \cdot \left(\frac{\left(\mu + \mu' f_n \rho\tilde{\nu}\right)\nabla\tilde{\nu}}{\sigma}\right) = \rho\left(P - D\right) + \frac{\mu'}{\sigma}c_{b2}\rho\nabla\tilde{\nu}\cdot\nabla\tilde{\nu} - \frac{1}{\sigma}\left(\nu + \mu'\tilde{\nu}\right)\nabla\rho\cdot\nabla\tilde{\nu} \quad (4.1)$$

where $\mu' = 1000$ is a scaling factor used to improve the convergence rate of the implicit Newton-Krylov solver by making the turbulence working variable comparable in magnitude to other physical quantities [96]. In Equation 4.1, P and D denote the production and destruction of eddy viscosity which depend on the sign of the eddy viscosity and are defined as:

$$P = \begin{cases} c_{b1}\left(1 - f_{t2}\right)\tilde{S}\tilde{\nu} & \tilde{\nu} \geq 0 \\ c_{b1}\left(1 - c_{t3}\right)S\tilde{\nu} & \tilde{\nu} < 0 \end{cases}, \; D = \begin{cases} \mu'\left(c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right)\left(\frac{\tilde{\nu}}{d}\right)^2 & \tilde{\nu} \geq 0 \\ -\mu' c_{w1}\left(\frac{\tilde{\nu}}{d}\right)^2 & \tilde{\nu} < 0 \end{cases} \quad (4.2)$$

In Equation 4.2, $\tilde{S}$ is the modified vorticity, which must always remain positive:

$$\tilde{S} = \begin{cases} S + \bar{S} & \bar{S} \geq -c_{v2}S \\ S + \frac{S\left(c_{v2}^2 S + c_{v3}\bar{S}\right)}{\left(c_{v3} - 2c_{v2}\right)S - \bar{S}} & \bar{S} \leq -c_{v2}S \end{cases} \quad (4.3)$$

$$\begin{aligned} S &= \left|\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right| \\ \bar{S} &= \frac{\mu'\tilde{\nu}f_{v2}}{\kappa^2 d^2} \end{aligned} \quad (4.4)$$

and d is the distance to the closest wall. The functions f_{v1} and f_{v2} are :

$$f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \; f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \; \chi = \frac{\mu'\tilde{\nu}}{\nu} \quad (4.5)$$

and functions f_n and f_{t2} are defined as:

$$f_n = \frac{c_{n1} + \chi^3}{c_{n1} - \chi^3}, \; f_{t2} = c_{t3}\exp\left(-c_{t4}\chi^2\right) \quad (4.6)$$

The destruction term coefficients are:

$$\begin{aligned} r &= \min\left(\frac{\mu'\tilde{\nu}}{\tilde{S}\kappa^2 d^2}, 10\right) \\ g &= r + c_{w2}\left(r^6 - r\right) \\ f_w &= g\left[\frac{1+c_{w3}^6}{g^6+c_{w3}^6}\right] \end{aligned} \tag{4.7}$$

The constant values used in this model are as follows: $\sigma = 0.66$, $\kappa = 0.41$, $c_{v1} = 7.1$, $c_{v2} = 0.7$, $c_{v3} = 0.9$, $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $c_{w1} = \frac{c_{b1}}{\kappa 2} + \frac{1+c_{b2}}{\sigma}$, $c_{w2} = 0.3$, $c_{w3} = 2.0$, $c_{t3} = 1.2$, $c_{t4} = 0.5$, $c_{n1} = 16$.

The conservative form of the compressible Reynolds Averaged Navier-Stokes (RANS) equations combined with the SA-neg turbulence model equation can be re-arranged as:

$$\frac{\partial U}{\partial t} + \nabla \cdot \left(\vec{F}_c\left(U\right) - \vec{F}_v\left(U, \nabla U\right)\right) = S\left(U, \nabla U\right) \tag{4.8}$$

The solution, flux and source term vectors for the coupled system of RANS-SA in two-dimensions are given as:

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E_t \\ \rho\tilde{\nu} \end{pmatrix}, \quad F_c^x = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u\left(E_t + P\right) \\ \rho u\tilde{\nu} \end{pmatrix}, \quad F_c^y = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ v\left(E_t + P\right) \\ \rho v\tilde{\nu} \end{pmatrix}$$

$$F_v^x = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} + c_p\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right)\frac{\partial T}{\partial x} \\ \frac{1}{\sigma}\left(\mu + \mu' f_n \rho\tilde{\nu}\right)\frac{\partial \tilde{\nu}}{\partial x} \end{pmatrix}, \quad F_v^y = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} + c_p\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right)\frac{\partial T}{\partial y} \\ \frac{1}{\sigma}\left(\mu + \mu' f_n \rho\tilde{\nu}\right)\frac{\partial \tilde{\nu}}{\partial y} \end{pmatrix}$$

$$S = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \rho\left(P - D\right) + \frac{\mu'}{\sigma}c_{b2}\rho\nabla\tilde{\nu}\cdot\nabla\tilde{\nu} - \frac{1}{\sigma}\left(\nu + \mu'\tilde{\nu}\right)\nabla\rho\cdot\nabla\tilde{\nu} \end{pmatrix} \tag{4.9}$$

where $Pr_{_T}$ is the turbulent Prandtl number, μ_T is the turbulent eddy viscosity and the other variables are the same as those described for the Navier-Stokes equations in Chapter 2. We assume that the working fluid is air with $Pr_{_T} = 0.9$; the turbulent eddy viscosity is given as:

$$\mu_T = \begin{cases} \mu' \rho \tilde{\nu} f_{v1} & \tilde{\nu} \geq 0 \\ 0 & \tilde{\nu} < 0 \end{cases} \tag{4.10}$$

Also, the total viscous stress tensor, τ_{ij}, including the Boussinesq approximated Reynolds stresses is found as:

$$\tau_{ij} = 2\left(\mu + \mu_T\right)\dot{\gamma}_{ij} \tag{4.11}$$

4.2 Flux Functions

Many turbulent flow solvers discretize the turbulence model equation in a loosely-coupled manner, which treats the convection term as though it were a scalar transport equation evolving with a prescribed velocity field. However, this treatment neglects the fact that the velocity field, which convects the turbulence model quantities, is heavily influenced by the turbulence model solution. Recent work by Burgess and Mavriplis [45] demonstrated the importance of full coupling of the RANS equations with the one equation SA model. In this section, we describe the numerical flux functions used for the discretization of convective and viscous fluxes.

4.2.1 Convective Fluxes

The RANS equations closed with the SA turbulence model results in a total of five equations in two spatial dimensions whose flux function must be re-derived. To use a flux-difference splitting method, we need the eigenvalues of the Jacobian of the normal convective flux for the system of RANS equations:

$$F_c^n = \begin{pmatrix} \rho u \hat{n}_x + \rho v \hat{n}_y \\ \left(\rho u^2 + P\right)\hat{n}_x + \rho u v \hat{n}_y \\ \rho u v \hat{n}_x + \left(\rho v^2 + P\right)\hat{n}_y \\ u\left(E_t + P\right)\hat{n}_x + v\left(E_t + P\right)\hat{n}_y \\ \rho u \tilde{\nu} \hat{n}_x + \rho v \tilde{\nu} \hat{n}_y \end{pmatrix} \tag{4.12}$$

It is possible to show that the eigenvalues of $\frac{\partial F_c^n}{\partial U}$ are:

$$\lambda_1 = \vec{V} \cdot \hat{n} - a \;\; \lambda_2 = \lambda_3 = \lambda_4 = \vec{V} \cdot \hat{n} \;\; \lambda_5 = \vec{V} \cdot \hat{n} + a \tag{4.13}$$

in which a is the sound speed. We use the numerical flux function of Roe and Pike [97] that was derived by Burgess and Mavriplis [45] for the RANS-SA system:

$$F_c\left(U_L, U_R\right) = \frac{1}{2}\left(F_c\left(U_L\right) + F_c\left(U_R\right) - D\right) \tag{4.14}$$

where D is the dissipative component of the numerical flux, which is given as:

$$D = \begin{pmatrix} \left|\tilde{\lambda}_2\right|\left(\rho_L - \rho_R\right) + \delta_1 \\ \left|\tilde{\lambda}_2\right|\left(\rho u_L - \rho u_R\right) + \delta_1 \tilde{u} + \delta_2 \hat{n}_x \\ \left|\tilde{\lambda}_2\right|\left(\rho v_L - \rho v_R\right) + \delta_1 \tilde{v} + \delta_2 \hat{n}_y \\ \left|\tilde{\lambda}_2\right|\left(E_{t,L} - E_{t,R}\right) + \delta_1 \tilde{H} + \delta_2\left(\tilde{u}\hat{n}_x + \tilde{v}\hat{n}_y\right) \\ \left|\tilde{\lambda}_2\right|\left(\rho\tilde{\nu}_L - \rho\tilde{\nu}_R\right) + \delta_1 \tilde{\tilde{\nu}} \end{pmatrix} \tag{4.15}$$

$$\delta_1 = \left(-\left|\tilde{\lambda}_2\right| + \frac{\left|\tilde{\lambda}_1\right| + \left|\tilde{\lambda}_5\right|}{2}\right)\frac{\Delta P}{\tilde{a}^2} + \frac{\left|\tilde{\lambda}_5\right| - \left|\tilde{\lambda}_1\right|}{2}\frac{\tilde{\rho}}{\tilde{a}}\left(\hat{n}_x \Delta u + \hat{n}_y \Delta v\right)$$

$$\delta_2 = \left(-\left|\tilde{\lambda}_2\right| + \frac{\left|\tilde{\lambda}_1\right| + \left|\tilde{\lambda}_5\right|}{2}\right)\tilde{\rho}\left(\hat{n}_x \Delta u + \hat{n}_y \Delta v\right) + \frac{\left|\tilde{\lambda}_5\right| - \left|\tilde{\lambda}_1\right|}{2}\frac{\Delta P}{\tilde{a}}$$

Note that in Equation 4.15, $\Delta\left(\right) = \left(\right)_L - \left(\right)_R$ and $\tilde{X}$ refers to quantity X evaluated at the Roe state [59]. In particular, $\tilde{\tilde{\nu}}$ is the turbulence model working variable at Roe's state. This flux function is computationally less expensive compared to the implementation of Roe's flux function available in our solver as it does not require any matrix-matrix or matrix-vector multiplication.

4.2.2 Viscous Fluxes

As described in Chapter 2, the gradient at the face quadrature points can be calculated by the average of the two reconstructed gradients and that value is employed to calculate the diffusive fluxes. However, this strategy fails for highly anisotropic meshes. To shed light on this issue, we consider the Poisson equation, $\nabla^2\phi = S$, as a model of viscous discretization along with the method of manufactured solutions

on anisotropic meshes. For this purpose, we create irregular anisotropic meshes on a rectangular domain $(x, y) \in [0, 1] \times [0, 0.05]$ using the procedure that has been described in detail by Diskin *et al* [98]. Figure 4.1 depicts the anisotropic mesh test case with 20×20 cells. For generating the test case, the first step is stretching a regular rectangular grid with $(N + 1) \times (N + 1)$ nodes toward the bottom line $y = 0$ using a stretching factor s and the maximum aspect ratio $AR_{\max}$. The y-coordinates of the horizontal grid lines in the top half of the domain are defined as:

$$y_k = y_{k-1} + \frac{s^{k}}{AR_{\max} \cdot \ N} \tag{4.16}$$

in which

$$y_0 = 0\,, \quad k = 1,\, 2,\, ...,\, N \tag{4.17}$$

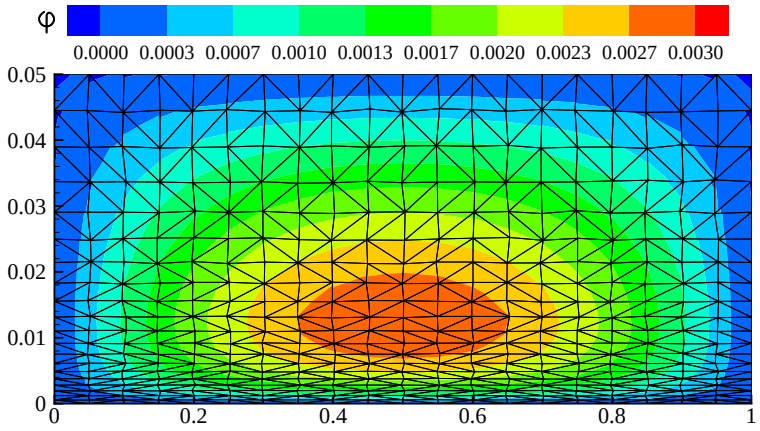
Then, irregularities are introduced by random shifts on interior nodes in vertical and horizontal directions and finally each perturbed quadrilateral is randomly triangulated with one of the two choices for its diagonal. In Figure 4.1 where $N = 20$, the stretching factor and the maximum aspect ratio are set as $s = 1.14$ and $AR_{\max} = 100$ to yield triangles whose aspect ratios vary approximately between 10 and 100. This mesh consists of anisotropic triangles which are long in the x-direction and thin in the y-direction.

In this study, these meshes with $N = 10,\, 20,\, 40,\, 80$ are used where the stretching factors are $s = 1.28,\, 1.14,\, 1.065,\, 1.0335$, respectively. The s values are chosen so as to keep the the length scale reduction in the y-direction the same as the $x-$direction. Since the anisotropic meshes are used for anisotropic solutions, the function being used for test purposes is manufactured to comply with the cells' aspect ratio. The anisotropic mesh of Figure 4.1 is stretched toward the horizontal line $y = 0$ using Equation 4.16 which is strongly dependent on the y-coordinates. The variations in the x-direction are introduced by an exponential function to yield isotropic behavior in this direction and homogeneous boundary conditions:

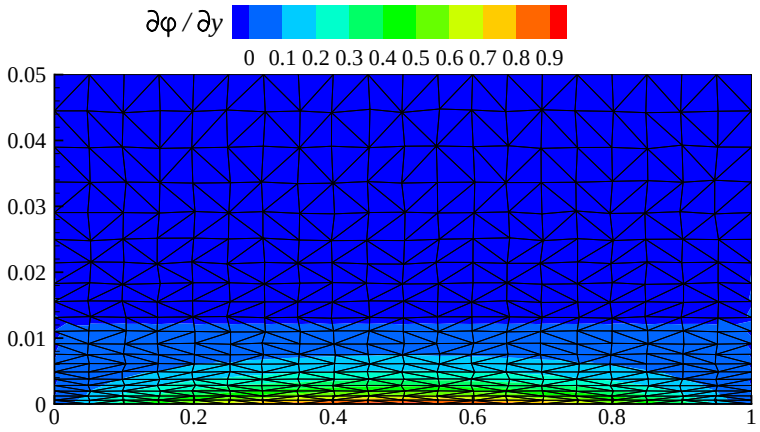
$$\phi\,(x, y) = \frac{\exp\left(x\,(x-1)\,y\,(y-0.05)\right) - 1}{\frac{s}{AR_{\max}} + N\left(1 - \frac{1}{s}\right) y} \tag{4.18}$$

In this part, Equation 4.18 is used as the test function with geometric values corresponding to the coarsest mesh ($N = 10$ and $s = 1.28$). Figure 4.1 depicts the

manufactured solution on one of the constructed anisotropic mesh along with its y-derivative which is too large near the bottom line with highly stretched cells and is reduced smoothly in the vertical direction compliant with cells' aspect ratio.



(a) Solution



(b) y-derivative

Figure 4.1: Anisotropic solution and derivative on a stretched grid with 20×20 cells

Figure 4.2 shows the eigenvalues of the global Jacobian matrix for a second-order discretization of Poisson's equation on this mesh. It is worth-mentioning that the global Jacobian matrix of a linear problem is independent of the solution. Note that pure averaging of the two reconstructed gradients for viscous flux discretization leads to the appearance of eigenvalues in the right-half plane that has potential to make the solution procedure unstable. However, a jump term which comes from the

discontinuous solution at the Gauss points can be added to the face gradient [99]:

$$(\nabla\phi)_F = \frac{1}{2}\left((\nabla\phi)_R + (\nabla\phi)_L\right) + \frac{\alpha}{|\vec{r}_{ij}.\hat{n}|}\left(\phi_L - \phi_R\right)\hat{n} \tag{4.19}$$

where α is an arbitrary value (we use $\alpha = 1$), $\vec{r}_{ij}$ is the vector that connects the reference points of the two control volumes that share the face and $\hat{n}$ is the outward unit normal vector at the Gauss point. The jump term has the role of stabilizing by damping high-frequency errors. This can also be seen in Figure 4.2 as adding this term pushes all the eigenvalues to the left-half plane [100]. We generalize this idea to the complicated case of the non-linear RANS equations on anisotropic meshes and use Equation 4.19 to compute the gradients involved in the viscous fluxes of Equation 4.9.

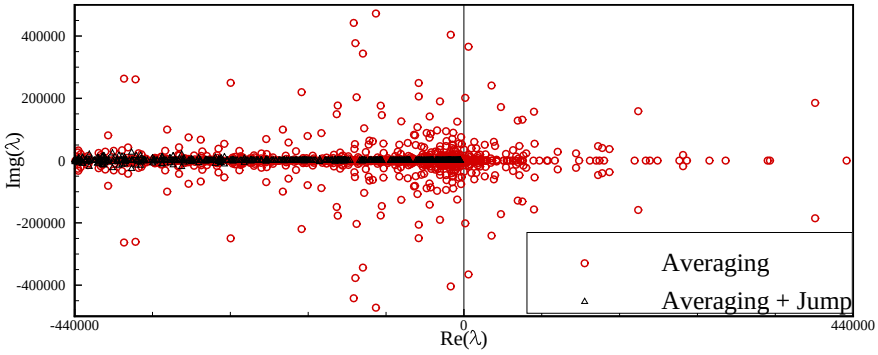


Figure 4.2: Comparison of eigenvalue spectra for the discretization of viscous fluxes on anisotropic meshes

4.3 Solution Method

In the context of RANS simulations with finite volume solvers, it is common to use explicit Runge-Kutta time-stepping even for the steady-state solution of the flow field. However, these methods slow down the convergence significantly unless combined with multigrid solution techniques [101]. Alternatively, we use the implicit pseudo time-stepping method described in Section 2.5 to accelerate the convergence of our flow solver:

$$\left(\frac{I}{\Delta t/A} + \frac{\partial R}{\partial \bar{U}}\right)\delta U = -R\left(\bar{U}^n\right) \tag{4.20}$$

For the sake of brevity, we re-write this equation as:

$$L\,\delta U_i = -R\left(\bar{U}^n\right) \tag{4.21}$$

The convergence to the steady-state solution can be posed as an optimization problem where we search for the minimum of an objective function. We define our objective function as:

$$f\left(\tilde{U}\right) = \frac{1}{2}\left|R_t\left(\tilde{U}\right)\right|^2 = \frac{1}{2}R_t^T\left(\tilde{U}\right)R_t\left(\tilde{U}\right) \tag{4.22}$$

where the unsteady residual is defined as:

$$R_t\left(\tilde{U}\right) = \frac{I}{\Delta t/A}\left(\tilde{U} - \bar{U}^n\right) + R\left(\tilde{U}\right) \tag{4.23}$$

In other words, we seek a trial solution, $\tilde{U}$, that minimizes the norm of the unsteady residual vector. Multiplying both sides of Equation 4.21 by the transpose of its left-hand side gives

$$\delta U^T\,L^T\,L\,\delta U = -\delta U^T\,L^T\,R\left(\bar{U}_i^n\right) = -\delta U^T\,\left.\frac{\partial f}{\partial \tilde{U}}\right|_{\bar{U}^n} > 0 \tag{4.24}$$

where the inequality comes from the fact that the left-hand side is the dot product of a non-zero vector with itself. Considering that

$$\delta U^T\,\left.\frac{\partial f}{\partial \tilde{U}}\right|_{\bar{U}^n} < 0, \tag{4.25}$$

δU is a descent direction for the scalar function $f\left(\tilde{U}\right)$. As a result, we can incorporate a line search algorithm with the implicit solver to enhance the robustness. For this purpose, an under-relaxation factor, ω^n , is used at each iteration to update the solution:

$$\bar{U}^{n+1} = \bar{U}^n + \omega^n \delta U \tag{4.26}$$

The under-relaxation factor can be found in a way to guarantee sufficient decrease in the norm of the unsteady residual vector such that $\left\|R_t\left(\tilde{U}\right)\right\|_2 < \left\|R\left(\bar{U}^n\right)\right\|_2$. However,

Ceze and Fidkowski showed that relaxing the sufficient decrease condition to

$$\left\| R_t\left(\tilde{U}\right)\right\|_2 < \kappa_{LS} \left\| R\left(\bar{U}^n\right)\right\|_2 \tag{4.27}$$

with $\kappa_{LS} > 1$ is helpful in accelerating the convergence of a fully coupled RANS solver [90]. In our work, we use $\kappa_{LS} = 1.2$ as the relaxation parameter for the sufficient non-increase condition. We also need to choose the under-relaxation factor so as to ensure a physical solution at the next iteration. Note that the physicality of the solution state is verified by checking the positivity of density and pressure at all quadrature points. Therefore, we start with the initial guess of $\omega^n = 1$ and check if the new solution state is physical *and* satisfies the relaxed decrease condition. If not, we halve the under-relaxation factor every time until the both conditions are satisfied.

For time accurate solutions, the time step in Equation 4.20 must be the same for all control volumes. However, local time-stepping with a global CFL number can be used when we seek the steady-state solution to accelerate the convergence process. The control volume wise time steps at each iteration are obtained as:

$$\Delta t_i^n = \frac{CFL^n \cdot h_i}{\lambda_{\max,\, i}} \tag{4.28}$$

where $\lambda_{\max,\, i}$ is the eigenvalue of the convective flux Jacobian with the largest magnitude obtained from Equation 4.13, and h_i is a characteristic size for the control volume. In the first stages of the computations, the initial solution does not satisfy the boundary conditions and thus strong transients take place in the solution. It is important to keep the global CFL number small at these stages to make the solution follow a physical path. As we proceed, the CFL number must be increased to make the convergence to the steady-state solution faster. Therefore, an evolution strategy is required for the CFL from its initial value to a large value such that Equation 4.20 becomes Newton's method and the solution approaches the steady state. We use the exponential progression with under-relaxation strategy proposed by Ceze and Fidkowski [96] in which the size of CFL growth is tuned by the under-relaxation factor. This strategy can be summarized as:

$$CFL^{n+1} = \begin{cases} \beta \cdot CFL^n & \omega^n = 1 \\ CFL^n & \omega_{min} < \omega^n < 1 \\ \kappa \cdot CFL^n & \omega^n < \omega_{min} \end{cases} \tag{4.29}$$

In our work, we set the parameters to $\beta = 1.5$, $\kappa = 0.1$ and $\omega_{min} = 0.01$. Note that when we search for the appropriate under-relaxation factor which satisfies both the relaxed sufficient decrease and physicality conditions, we terminate the search process for $\omega^n < \omega_{min}$ and cut the global CFL number according to Equation 4.29.

4.4 Results

In this section, we present our results for four fully turbulent cases: subsonic flow over a flat plate, subsonic flow over the NACA 0012 airfoil, transonic flow over the RAE 2822 airfoil and flow around a high-lift multi-element configuration [102]. Our results include verification of the turbulence model discretization, accuracy advantages obtained by higher-order discretizations and the convergence behavior of each case. For all these test cases, the no-slip and zero turbulent viscosity boundary conditions on the walls are applied by adding an extra constraint to the least-squares system for solution reconstruction for each boundary Gauss point. In addition, the adiabatic wall condition is applied weakly by zeroing the heat flux components in Equation 4.9 for wall boundary faces. For inflow, we specify the values of total pressure, velocity components and turbulent viscosity while the static pressure is the only pre-specified condition at the outflow; other quantities are obtained from solution reconstruction. No point vortex correction is done for the far-field boundaries. Also, the dynamic viscosity is obtained by Sutherland's law.

4.4.1 High Reynolds Number Flow Over a Flat Plate

For this test case, we consider the 2D flat plate verification case from the NASA Turbulence Modeling Resource (TMR) website [85]. Figure 4.3 shows the problem geometry and boundary conditions for this problem with $Re = 5\times10^6$ and $Ma_\infty = 0.2$. The range of the computational domain in the x-direction is $[-0.33, 2]$ with the leading edge of the flat plate at $x = 0$. The size of the domain in the y-direction is 1. At $y = 0$, the symmetry boundary condition is applied for $-0.33 \leq x \leq 0$ weakly by

forcing the normal derivatives to be zero in the flux vectors and the no slip adiabatic boundary condition is imposed for $0 \leq x \leq 2$. Note that this causes a singularity in the solution at the leading edge. The free-stream value of the turbulent working variable is $\tilde{\nu}/\nu_\infty = 3.0$.

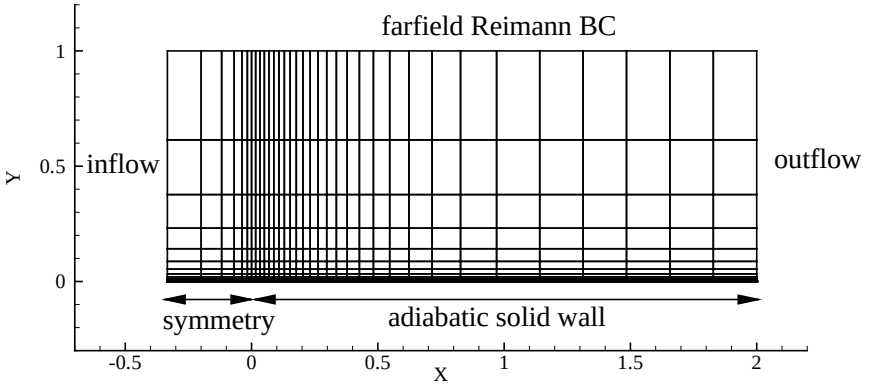


Figure 4.3: Turbulent flat plate test case mesh and geometry

For this test case, we generated a sequence of nested grids with quadrilateral cells. The coarsest mesh, which has 2040 cells has 61 nodes in the x- and 35 nodes in the y-direction. The grid is highly stretched near the wall such that the height of the first cell is 9.2×10^{-6}. Considering that our higher-order discretization employs a larger stencil for the Taylor series expansion, having anisotropic cells near singularities causes instability in the solution process. Therefore, we cluster the cells near the leading edge to produce nearly isotropic cells ($AR = \mathcal{O}(1)$) in that region. Finer meshes are obtained by uniform refinement of the coarsest mesh.

Figure 4.4 shows the skin friction along the wall and turbulent viscosity at $x = 0.97$ near the wall for second- to fourth-order discretizations on a mesh with $32,640$ control volumes. These profiles are converging to the grid-converged values obtained by the FUN3D solver from NASA Langley [85], which uses the SA-Neg model, on a much finer mesh. As expected, higher-order discretizations lead to more accurate estimates of the flow field on the same mesh. Closer views of the plots show that third- and fourth-order results are closer to the expected values. In addition, the fourth-order results for $32,640$ control volumes are comparable with second-order results on the next finer mesh (n.DoF $= 130,560$). Note that in the context of finite volume methods, the number of degrees of freedom (n.DoF) per equation is defined as the number of control volumes in the mesh independent of the order of

accuracy. These figures verify the correctness of our RANS flow solver and also show the accuracy advantages obtained by employing higher-order discretizations.

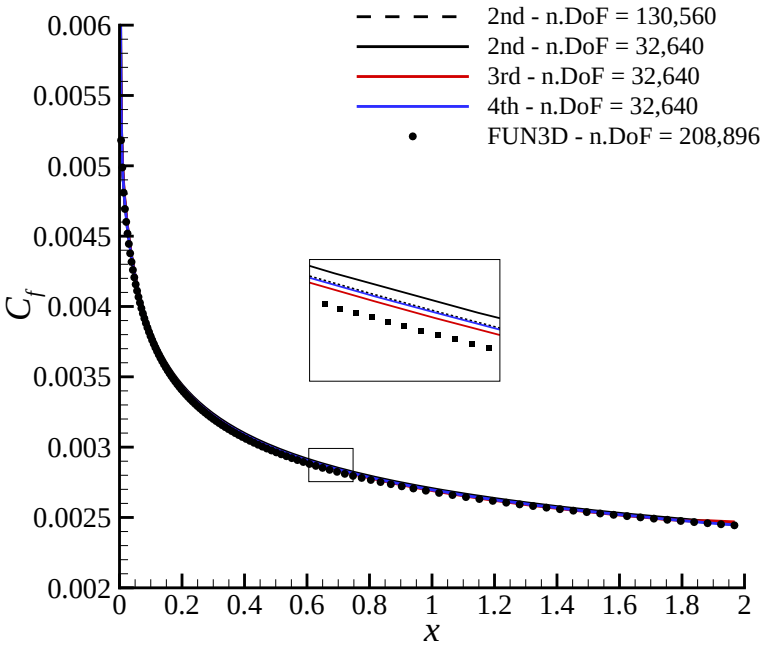
We also examine the convergence of friction drag coefficient on the sequence of systematically refined meshes for different orders. Table 4.1 shows the values of C_D on different meshes and error in the drag coefficient, defined as $C_{D,\,Err} = |C_D - C_{D,\,Ref}|$. Note that the reference value is obtained by fourth-order discretization on the finest mesh with $130,560$ control volumes. As expected, the error decreases by increasing the order of accuracy on the same mesh. Also, the drag values converge faster using a higher-order discretization so that the error associated with fourth-order on the second finest mesh is smaller than the error of the second-order scheme on the finest mesh.

n.DoF	C_D			$C_{D,Err}$		
	2nd	3rd	4th	2nd	3rd	4th
2040	0.003617	0.002163	0.002329	7.54×10^{-4}	7.00×10^{-4}	5.34×10^{-4}
8160	0.002979	0.002803	0.002849	1.16×10^{-4}	6.0×10^{-5}	1.4×10^{-5}
32640	0.002873	0.002862	0.002862	1.0×10^{-5}	2×10^{-6}	1×10^{-6}
130560	0.002865	0.002864	0.002863	2×10^{-6}	1×10^{-6}	0

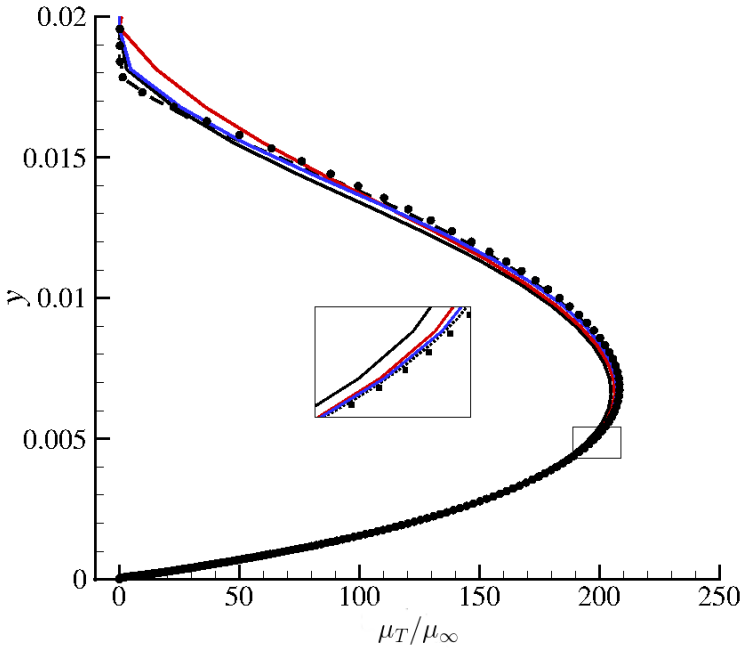
Table 4.1: Friction drag coefficient of turbulent flat plate for different orders and mesh sizes

To investigate the iterative convergence properties of our flow solver, relevant parameters have been tabulated in Table 4.2 for each order and three mesh sizes. This table gives the number of linear and non-linear iterations, total CPU time on a single core of an i7-4790 (3.60 GHz) CPU, and total number of work units until convergence (CPU time divided by the time of a single residual evaluation). In addition, the time for residual evaluation, Jacobian matrix formation and linear system solution averaged over the non-linear iterations is listed. Note that the Jacobian matrix is calculated to full order to enhance the convergence speed of the solver. The initial solution is the uniform free stream condition everywhere and the linear system solver is GMRES preconditioned with ILU(2). The size of the Krylov subspace is 100 for all the cases and the relative residual of the linear solver is $\eta_l = 10^{-3}$. Quotient minimum degree (QMD) re-ordering is applied at the time of preconditioning. The convergence process terminates when the L_2-norm of the residual vector drops below 10^{-8}.

An interesting observation is that the number of non-linear iterations is constant



(a) Wall friction coefficient



(b) Turbulent viscosity at $x = 0.97$

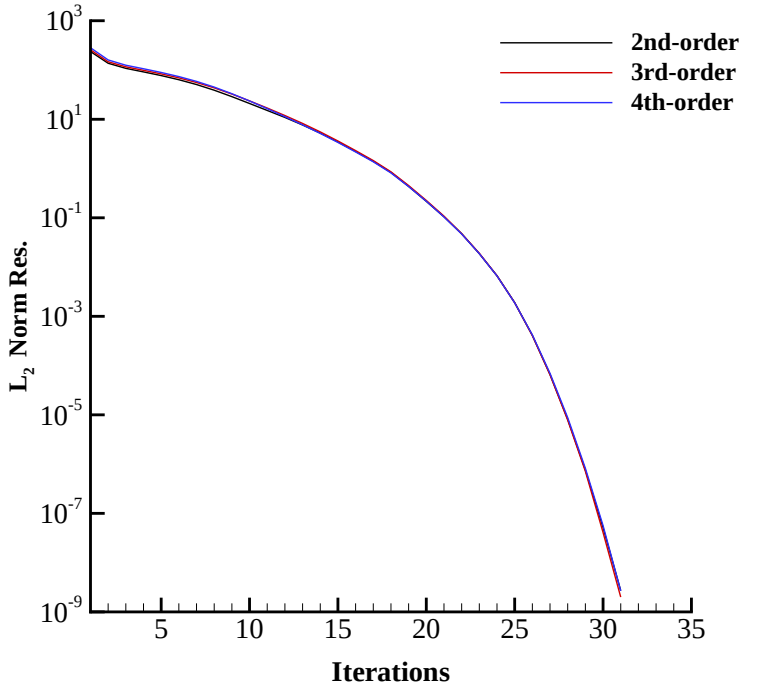
Figure 4.4: Distribution of wall friction factor and turbulent viscosity for flat plate

independent of the order on the same mesh and the number of linear iterations decreases with increasing order. With mesh refinement, the number of non-linear iterations slightly increases although considerably more linear iterations are required. As expected, the total time of computation increases with mesh and order refinement. It should be noted that the time of residual evaluation and Jacobian matrix formation are fully scalable as they increase by a factor of 4 in each level of mesh refinement. However, the time spent to solve the linear system does not linearly increase with the number of degrees of freedom. This is also expected since mesh refinement leads to a larger linear system with larger condition number while the size of the Krylov subspace is fixed.

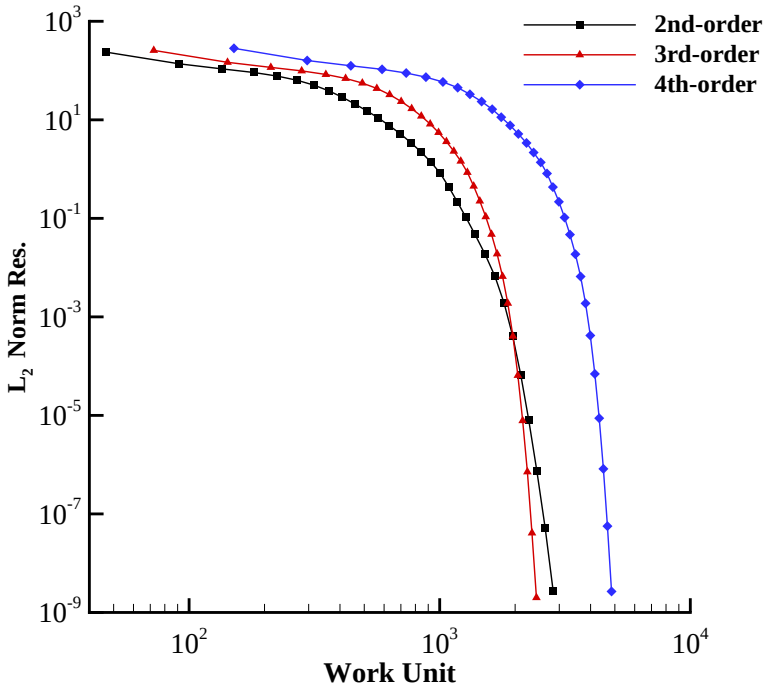
Figure 4.5 shows the convergence history for the flat plate test case on a mesh with $32,640$ cells. The drop in the L_2-norm of the residual vector in terms of non-linear iteration count is very similar for different orders. However, the difference can be found by considering the numbers of work unit. For all meshes, the number of work units is almost the same between second- and third-order and is about twice as large for fourth-order. This can be explained by the extra time required to form the Jacobian matrix with more non-zeros and factorize it for preconditioning in the case of fourth-order.

Order	n.DoF	No. of iterations		CPU Time (Sec.)	Work Unit	Average Time (Sec.)		
		Linear	Non-linear			Res. Eval.	Jac. Form.	LS Solver
2nd	$2,040$	291	28	6.08	1330	0.004	0.112	0.099
	$8,160$	696	29	27.28	1690	0.016	0.411	0.510
	$32,640$	2701	31	207.03	2840	0.073	1.644	4.954
3rd	$2,040$	155	28	16.68	1770	0.009	0.274	0.311
	$8,160$	318	29	74.64	2030	0.036	1.071	1.466
	$32,640$	668	31	366.17	2430	0.150	4.327	7.323
4th	$2,040$	98	28	31.70	1810	0.017	0.347	0.772
	$8,160$	269	29	164.82	3630	0.045	1.379	4.256
	$32,640$	576	31	865.84	4850	0.178	5.653	22.078

Table 4.2: Convergence properties of turbulent RANS solver for high Reynolds number flat plate



(a) Non-linear iterations



(b) Work units

Figure 4.5: Convergence history of finite volume solver for high-Reynolds turbulent flat plate for n.DoF = 32,640

4.4.2 Subsonic Flow Over a NACA 0012 Airfoil

The second test case considered is the subsonic flow over a NACA 0012 airfoil at angle of attack $\alpha = 10°$; the Mach number based on the free stream condition is $Ma_\infty = 0.15$ and the Reynolds number based on the airfoil chord is $Re_c = 6 \times 10^6$. This is also among the verification test cases of the TMR, which documents results obtained by established codes for different families of structured meshes. We use the meshes with the smallest spacing near the trailing edge (meshes of Family II [85]) where the solution is singular. We also convert the structured quad meshes into hybrid meshes comprised of a few quad layers near the wall and triangles (obtained by splitting quads) in other regions to have truly unstructured meshes. Computations are performed on a sequence of four meshes with $6,272$, $25,088$, $100,352$, $401,408$ control volumes. Figure 4.6 shows an example of these meshes. No-slip adiabatic boundary conditions are imposed on the airfoil surface and far-field conditions (inflow/outflow based on the direction of velocity vector) are applied on the outer boundary located $500c$ away from the airfoil. Again, the free-stream value of the turbulent working variable is $\tilde{\nu}/\nu_\infty = 3.0$.

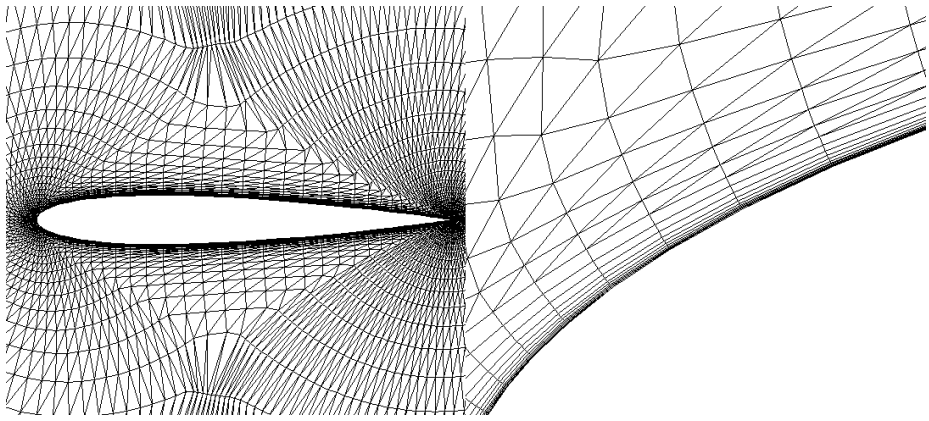


Figure 4.6: Example of unstructured hybrid meshes used for subsonic flow over NACA 0012 airfoil with $25,088$ degrees of freedom

Figure 4.7 shows the contours of turbulence working variable obtained by second- and fourth-order discretizations on a hybrid mesh with $25,088$ degrees of freedom. The fourth-order method resolves the wake region more accurately and provides a smoother distribution of scaled turbulent viscosity.

Figure 4.8 shows the distribution of pressure and friction coefficients on the wall for second- and fourth-order methods using a mesh with $100,352$ cells. For both

methods, the values are consistent with the distributions obtained by FUN3D on a super fine mesh (with about 1M cells); fourth-order gives more accurate values near the leading and trailing edges as expected. Note that these results are obtained by employing a curvilinear coordinate system for highly anisotropic cells near the wall with the procedure explained in Section 3.3.

Table 4.3 gives the values of pressure drag, viscous drag and lift coefficients on different meshes for all discretization orders using curvilinear and Cartesian coordinate systems. This table highlights the importance of accurate reconstruction on the final solution obtained by the solver. Comparing the values of coefficients with the reference values reported by TMR in the limit of mesh refinement reveals the fact that reconstruction in a curvilinear coordinate system leads to grid converged values considerably faster. As shown in Table 4.3a, coefficients of lift and drag converge to the reference values with mesh and order refinement. Also, the convergence rate increases for higher-order discretizations. Note that the second- and third-order results on the finest mesh and fourth-order result on the second finest mesh are very close to the reference values. On the other hand, using a Cartesian coordinate system gives poor output values, particularly for second- and fourth-order, even on very fine meshes (Table 4.3b). Although the values of drag and lift coefficients may converge to the reference values on super fine meshes, this is in contrast with the motivation of higher-order methods, which is supposed to produce more accurate results on coarser meshes.

Similarly, parameters related to the iterative convergence of our flow solver is listed in Table 4.4. The linear solver settings are the same as those reported for the flat plate test case and initial solution is set to uniform free stream values for all cases. The number of work units changes more slowly with mesh refinement for third-order than for second- and fourth-order. This comes from the larger number of linear iterations required for these cases and is related to the structure of Jacobian matrix. For all the cases, the solver converges in a reasonable number of non-linear iterations and work units although the linear solver is not perfectly scalable.

Figure 4.9 shows the convergence histories in terms of non-linear iteration count and number of work units for a mesh with $100,352$ control volumes.

n.DoF	$C_{D,p}$	$C_{D,v}$	C_L
Reference	0.00606	0.00620	1.0910
2nd order			
$6,272$	0.02748	0.00998	1.0466
$25,088$	0.01095	0.00704	1.0778
$100,352$	0.00685	0.00623	1.0881
$401,408$	0.00614	0.00617	1.0908
3rd order			
$6,272$	0.00019	0.00220	0.9432
$25,088$	0.00380	0.00556	1.0817
$100,352$	0.00587	0.00615	1.0877
$401,408$	0.00607	0.00619	1.0904
4th order			
$6,272$	0.00090	0.06200	0.5813
$25,088$	0.00544	0.00582	1.0951
$100,352$	0.00615	0.00623	1.0905

(a) Curvilinear coordinates

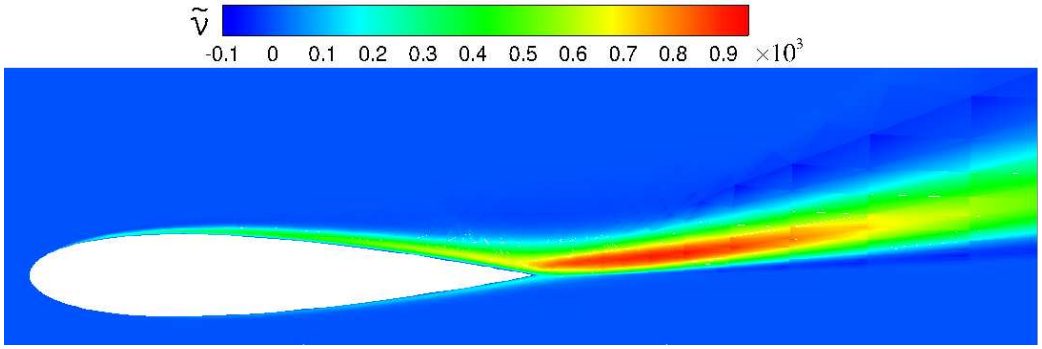
n.DoF	$C_{D,p}$	$C_{D,v}$	C_L
Reference	0.00606	0.00620	1.0910
2nd order			
$6,272$	0.02221	0.00084	1.1384
$25,088$	0.00443	0.00103	1.1765
$100,352$	0.00204	0.00207	1.1625
$401,408$	0.00321	0.00417	1.1353
3rd order			
$6,272$	0.00443	0.00818	0.8166
$25,088$	0.00740	0.00789	1.0199
$100,352$	0.00644	0.00662	1.0776
$401,408$	0.00612	0.00619	1.0899
4th order			
$6,272$	0.03832	0.00477	0.6857
$25,088$	0.01084	0.00862	1.0257
$100,352$	0.00749	0.00695	1.0724

(b) Cartesian coordinates

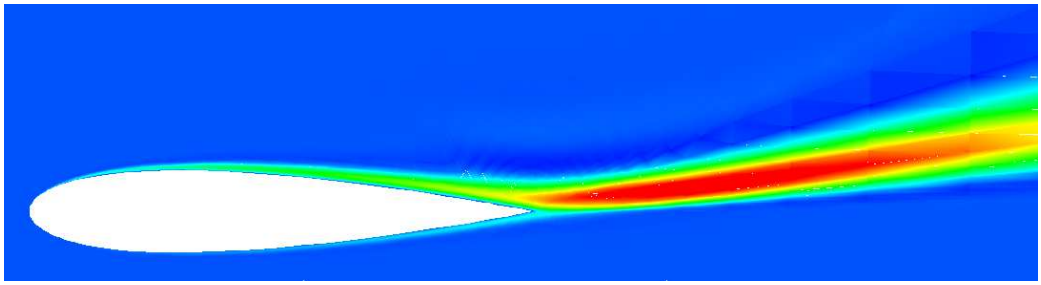
Table 4.3: Convergence of drag and lift coefficients with mesh refinement for subsonic flow around NACA 0012

Order	n.DoF	No. of iterations		CPU Time (Sec.)	Work Unit	Average Time (Sec.)		
		Linear	Non-linear			Res. Eval.	Jac. Form.	LS Solver
2nd	$6,272$	837	36	52.72	2140	0.025	0.61	0.82
	$25,088$	2521	43	301.72	3040	0.10	2.37	4.50
	$100,352$	7281	46	2126.52	5260	0.40	9.88	35.64
3rd	$6,272$	662	50	88.10	2260	0.039	0.92	0.77
	$25,088$	2210	56	486.05	3270	0.15	3.83	4.39
	$100,352$	6638	47	2432.71	4010	0.60	14.70	36.04
4th	$6,272$	389	55	203.19	4000	0.05	1.31	2.72
	$25,088$	1269	46	777.48	3810	0.20	5.29	11.12
	$100,352$	11673	76	8306.67	10220	0.81	21.25	85.30

Table 4.4: Convergence properties of turbulent RANS solver for subsonic flow over a NACA 0012 airfoil



(a) Second-order solution



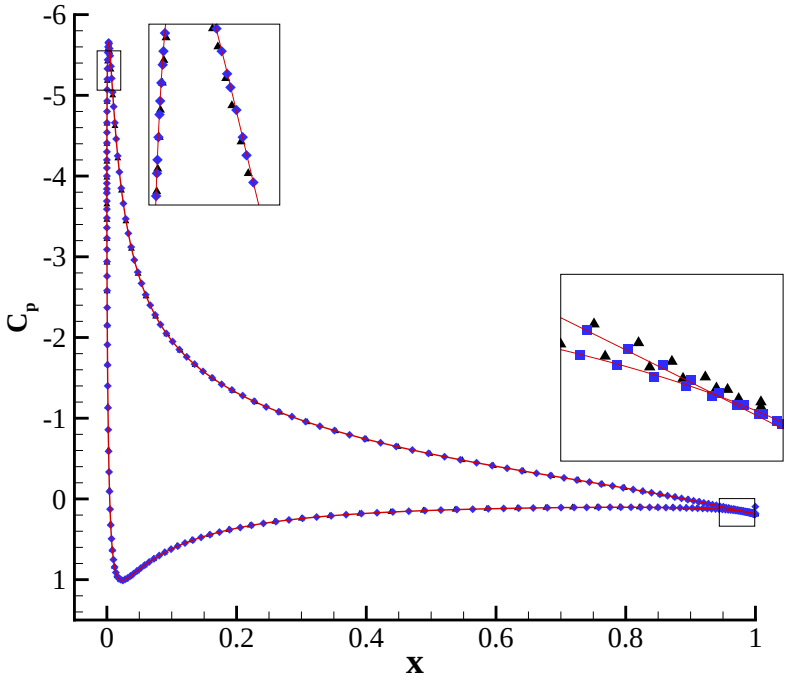
(b) Fourth-order solution

Figure 4.7: Distribution of scaled turbulence working variable for subsonic flow over NACA 0012 on a mixed-element mesh (n.DoF = 25,088)

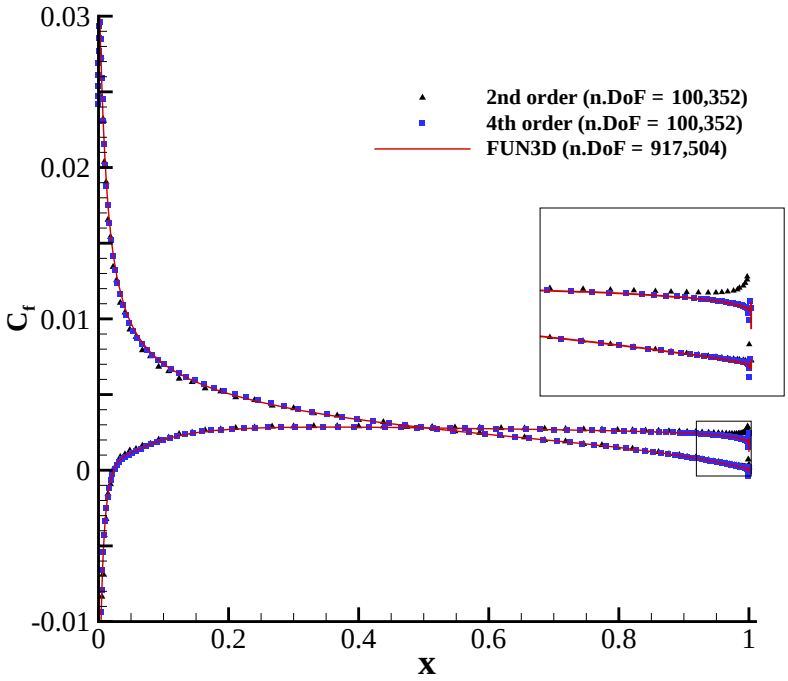
4.4.3 Transonic Flow Over a RAE 2822 Airfoil

In this test case, we consider a transonic, turbulent flow over a RAE 2822 airfoil. The free stream Mach number is $Ma_\infty = 0.73$, the Reynolds number based on the chord is $Re_c = 6.5 \times 10^6$ and the angle of attack is $\alpha = 2.79°$. For this problem, which is the same as case 9 in the experimental investigation of Cook *et al.* [103], a shock wave appears on the upper surface of the airfoil and interacts with the turbulent boundary layer, forming a small recirculation behind the shock. As a result, the convergence to the steady-state solution is more challenging, particularly for a higher-order discretization. This test case can examine the robustness of our solver for non-trivial flow situations with the presence of shock and singularities.

For this test case, we use a family of C-meshes consisting of triangles and quads with three levels of refinement (four levels for second-order). The far-field boundary is placed $20c$ away from the airfoil surface and adiabatic no-slip conditions are applied on the wall. The flow is assumed to be fully turbulent with free stream turbulent viscosity of $\tilde{\nu}/\nu_\infty = 3.0$.

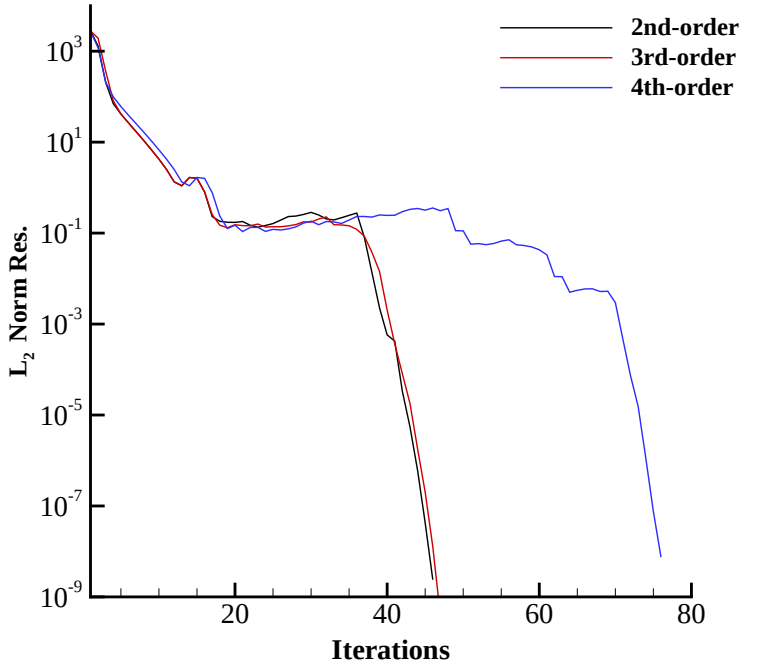


(a) Pressure coefficient

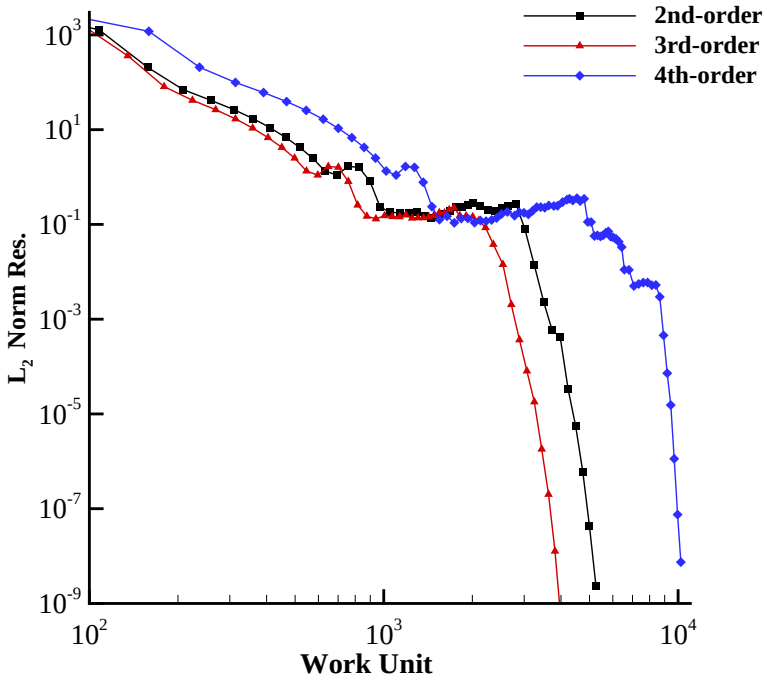


(b) friction coefficient

Figure 4.8: Distribution of surface pressure and friction coefficients for second- and fourth-order and comparison with FUN3D



(a) Non-linear iteration



(b) Work units

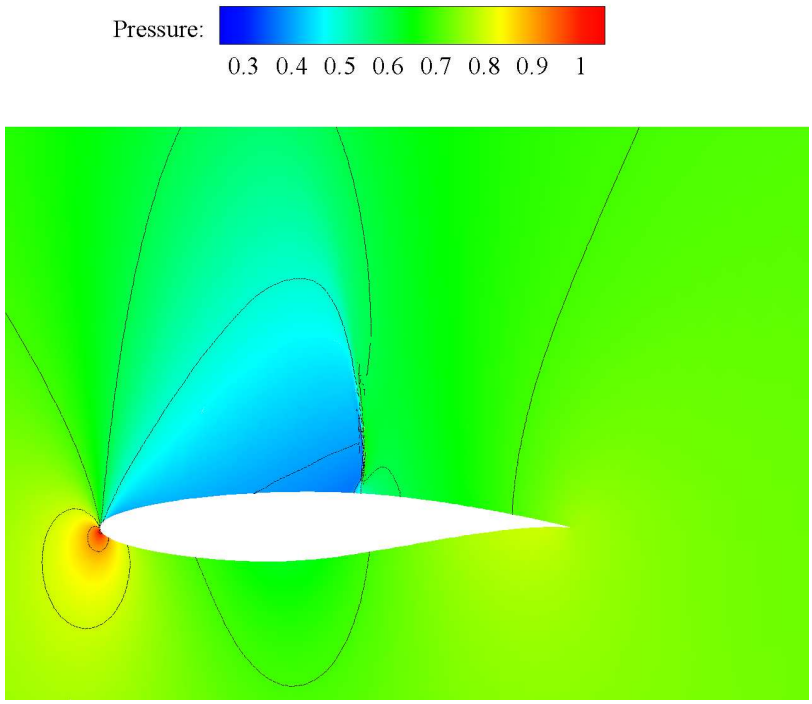
Figure 4.9: Convergence history of finite volume solver for subsonic flow over NACA 0012 for n.DoF = 100,352

Figure 4.10 shows the contours of pressure and scaled turbulence working variable for the third-order solution on a mesh with $35,840$ control volumes. Note that the shock wave and small recirculation bubble behind the shock have been captured automatically without any change in the flux functions or using slope limiters. In addition, the turbulence working variable predicted by the SA-neg model is largest in the wake region and negative at the edge of boundary layer.

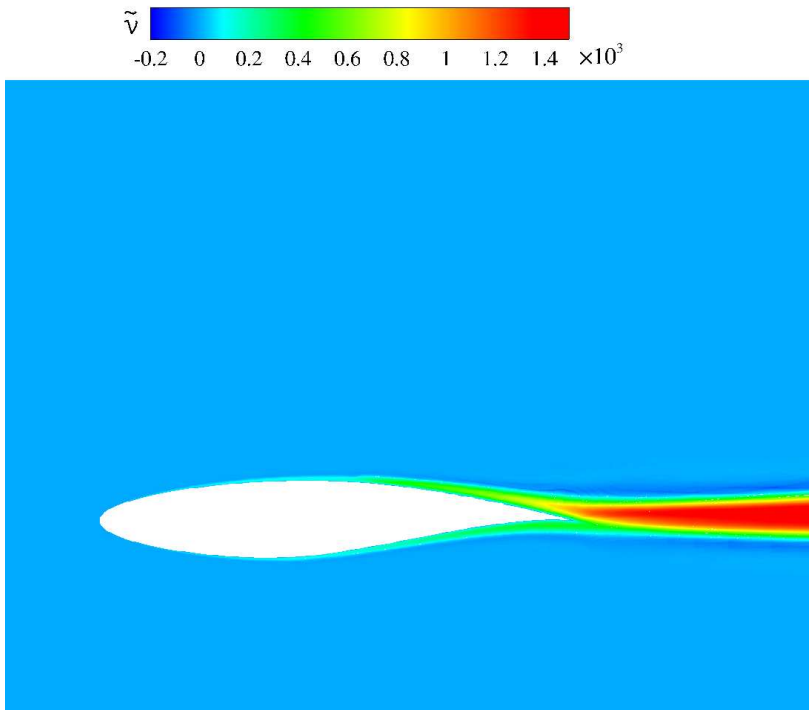
The distributions of pressure and friction coefficients obtained by different orders of discretizations are shown in Figure 4.11. There is a good match, including the location of the shock on the upper surface, between our computed pressure coefficients and experimental data. However, the difference between second- and higher-order values is less clear.

Figure 4.12 shows the convergence of drag and lift coefficients for this test case with mesh refinement. Similar to the other cases, the advantage of higher-order methods is observed as they converge faster to grid converged values. In addition, the error associated with fourth-order discretization is smaller on coarser meshes compared to second- and third-order discretizations.

As described earlier, the initial condition for the previous test cases were the uniform free stream states. However, the presence of shock wave in this case makes the convergence of our iterative solver slower as a considerable number of iterations are required to locate the shock properly. On the other hand, starting from a good initial guess, which gives the location of shock approximately, can help the convergence and even robustness in some cases. As is common in the higher-order community, a lower-order solution can be used as the initial state for the higher-order discretizations. Figure 4.13 shows the convergence history of our solver for different conditions on a mesh with $35,840$ degrees of freedom. Note that starting directly from free stream values leads to about the same number of non-linear iterations for second- and third-order and non-convergence for fourth-order. Alternatively, starting from the converged second-order solution results in fast convergence in a small number of iterations and work units. The iterative convergence properties tabulated for higher-order discretizations in Table 4.5 are for the conditions started from the second-order solution on the same mesh. Note that for the second-order cases, more iterations and work units are necessary compared to the previous cases, which illustrates the challenge in computing the solution of this case. To obtain a higher-order solution on the same mesh, only a few more work units are required which are perfectly affordable

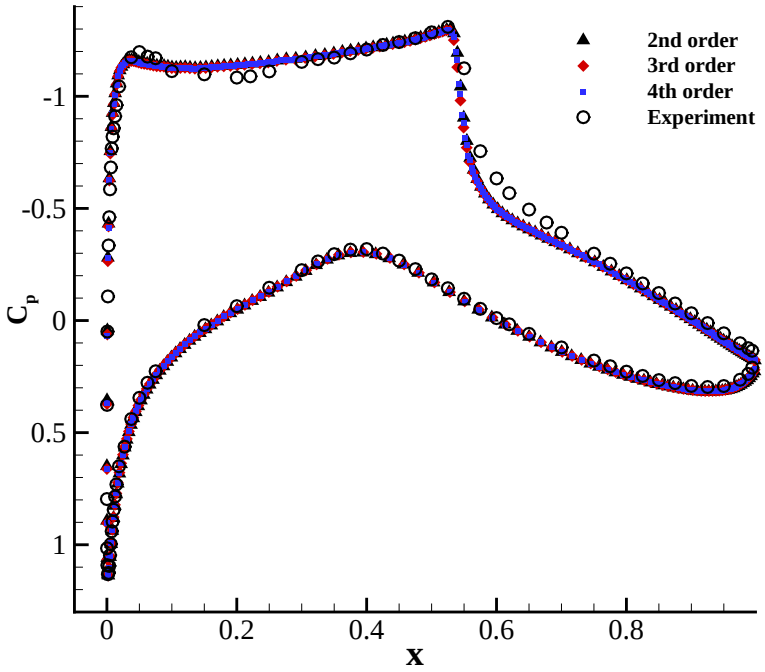


(a) Pressure

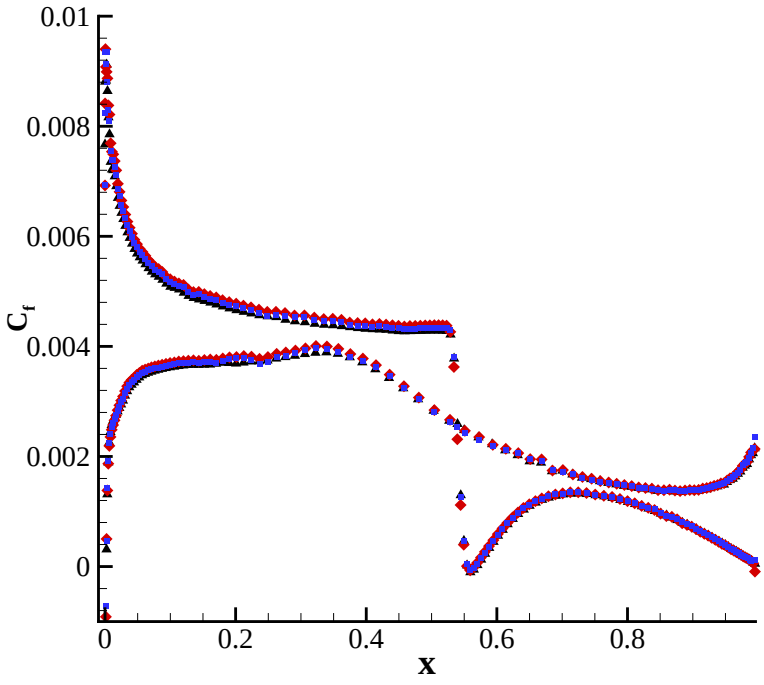


(b) Turbulence working variable

Figure 4.10: Third-order solution of transonic flow over RAE 2822 on a mixed-element mesh (n.DoF $= 35,840$)

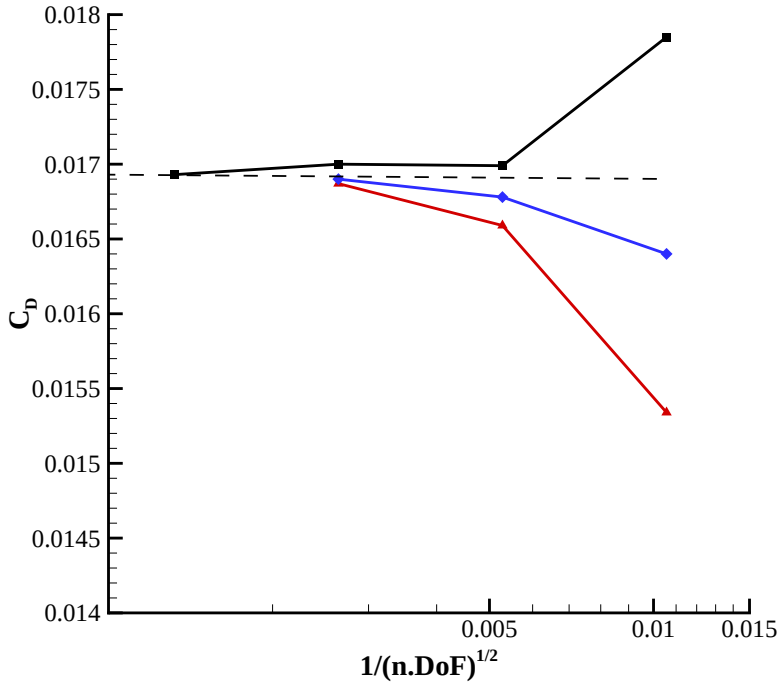


(a) Pressure coefficient

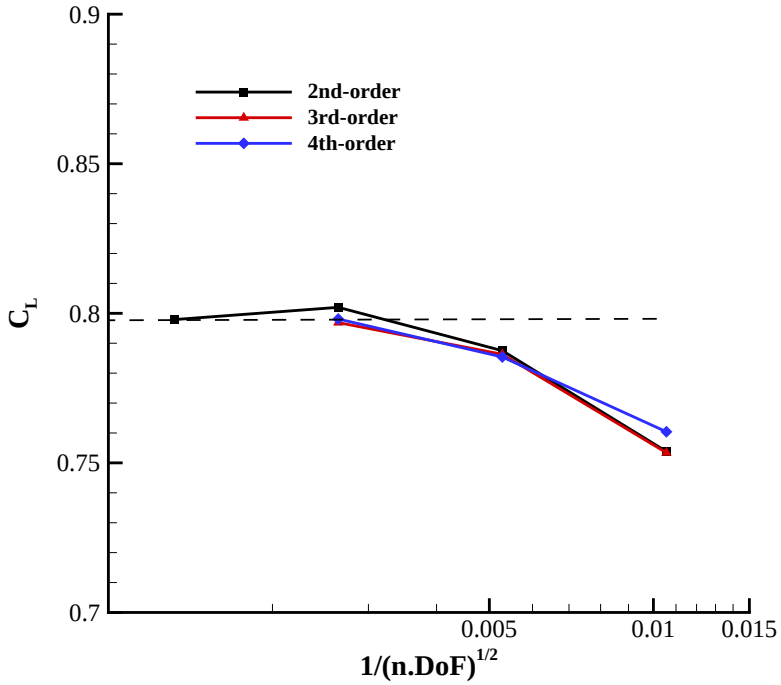


(b) Friction coefficient

Figure 4.11: Distribution of surface pressure and friction coefficients for different orders and comparison with experiment



(a) Drag coefficient



(b) Lift Coefficient

Figure 4.12: Convergence of drag and lift coefficients with mesh refinement for transonic flow around RAE 2822

for grid converged mesh resolutions.

Order	n.DoF	No. of iterations		CPU Time (Sec.)	Work Unit	Average Time (Sec.)		
		Linear	Non-linear			Res. Eval.	Jac. Form.	LS Solver
2nd	8,960	3000	60	123.24	3460	0.035	0.87	1.07
	35,840	13853	125	1429.55	9460	0.15	3.41	7.32
	143,360	27105	146	9126.80	14340	0.63	13.58	45.98
3rd	8,960	1164	44	104.77	1700	0.06	1.34	0.88
	35,840	2864	35	415.20	1810	0.23	5.25	6.29
	143,360	12167	59	4706.12	4990	0.94	21.24	55.65
4th	8,960	628	32	130.68	1560	0.084	1.91	2.06
	35,840	1831	36	696.62	2220	0.31	7.55	11.35
	143,360	6022	55	5840.08	4560	1.28	30.76	71.38

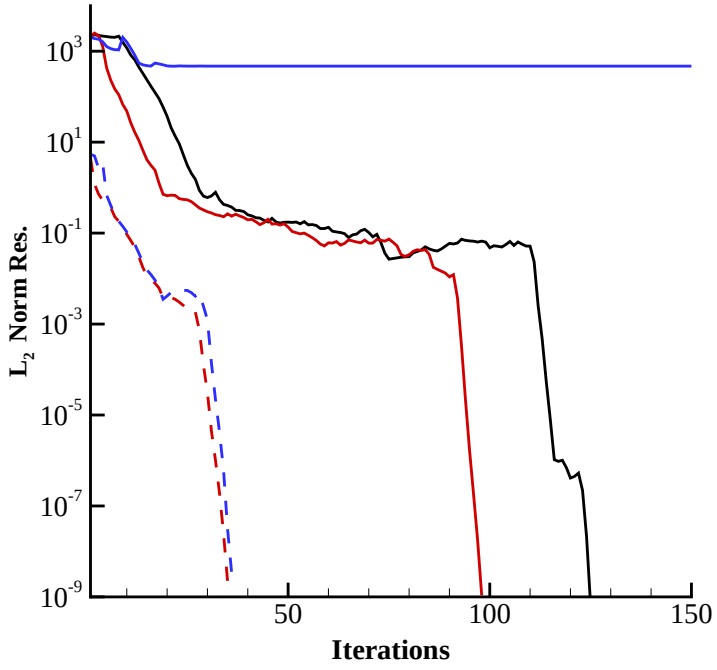
Table 4.5: Convergence properties of turbulent RANS solver for transonic flow over a RAE 2822 airfoil

4.4.4 High-lift Multi-element Airfoil

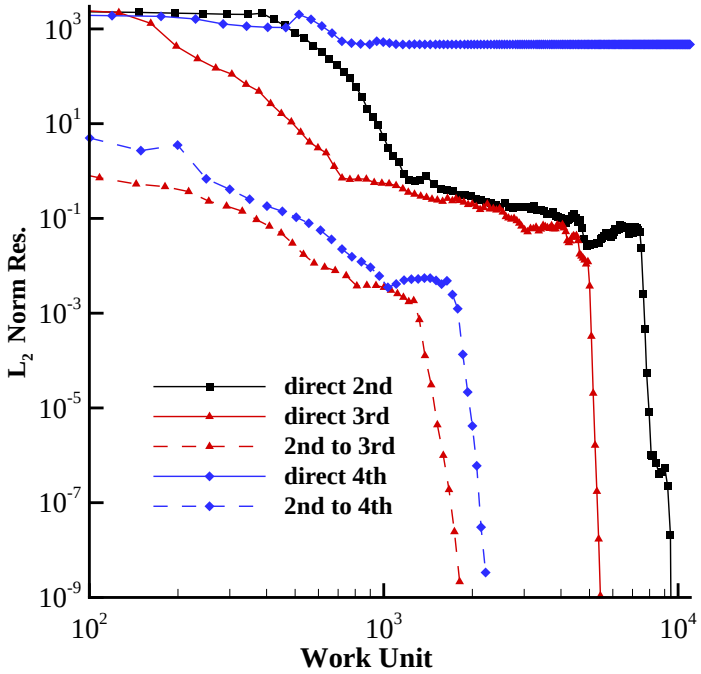
As our final test case, we consider turbulent flow over the three-element airfoil MDA 30P30N configuration. The free stream Mach number is $Ma_\infty = 0.2$, the Reynolds number is $Re = 9 \times 10^6$ and the angle of attack is $\alpha = 16^\circ$. The geometry of the multi-element airfoil, which consists of a leading edge slat, a main center element and a trailing edge flap, has a number of sharp corners. This is considered a hard test case in 2D because of the complex geometry, various flow structures and high angle of attack [104, 105].

For this test case, we only demonstrate the capability of our flow solver for one single mesh with 45,802 mixed cells, Figure 4.14. It is known that higher-order solution reconstruction on anisotropic meshes near singularities leads to solver failure even for simple geometries [106]. As a result, the mesh consists of isotropic cells near the sharp corners. Other cells that are sufficiently far from the sharp corners and still close to the walls have high aspect ratio. The new curvilinear coordinate is only constructed for these cells with high aspect ratio; isotropic cells still use the Cartesian coordinate system. Figure 4.14c shows the choice of reconstruction coordinate for control volumes near the slat.

Figure 4.15 shows the contours of the turbulence working variable for a fourth-



(a) Non-linear iteration



(b) Work units

Figure 4.13: Convergence history of finite volume solver for transonic flow over RAE 2822 for n.DoF = 35,840

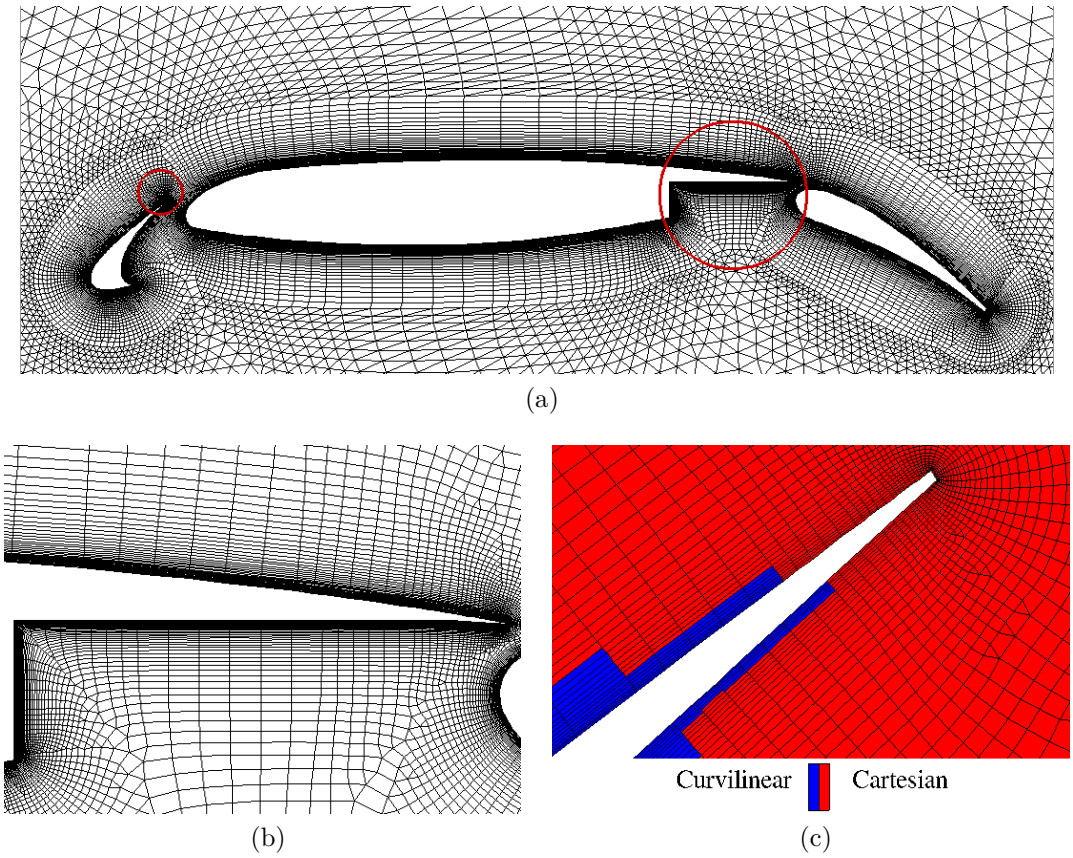


Figure 4.14: Mixed element mesh illustration for the high-lift three-element configuration

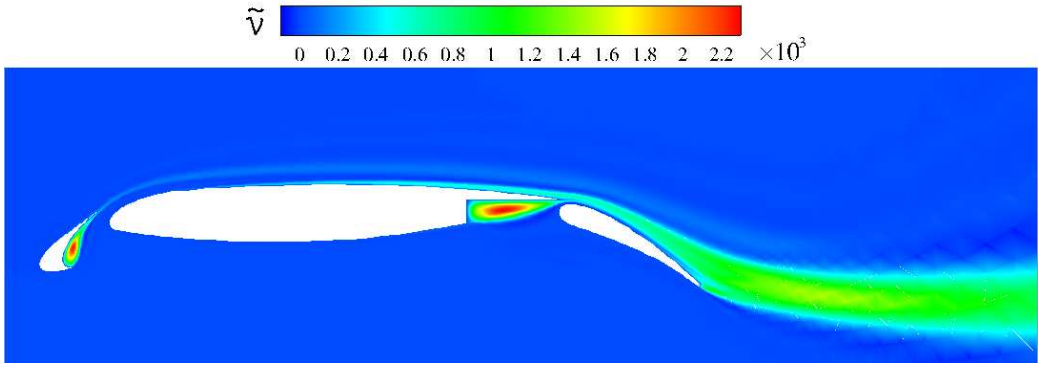


Figure 4.15: Fourth-order solution of turbulence working variable over multi-element airfoil on a mixed-element mesh (n.DoF $= 45,802$)

order discretization. Note that the slat wake continues all the way over the main element and flap and joins the wakes of the two other elements. Also, Figure 4.16 compares the distribution of pressure coefficient obtained by different orders of discretizations with experimental data available in Ref. [92]. As seen, there is a good match between the computed pressure coefficients and experimental data.

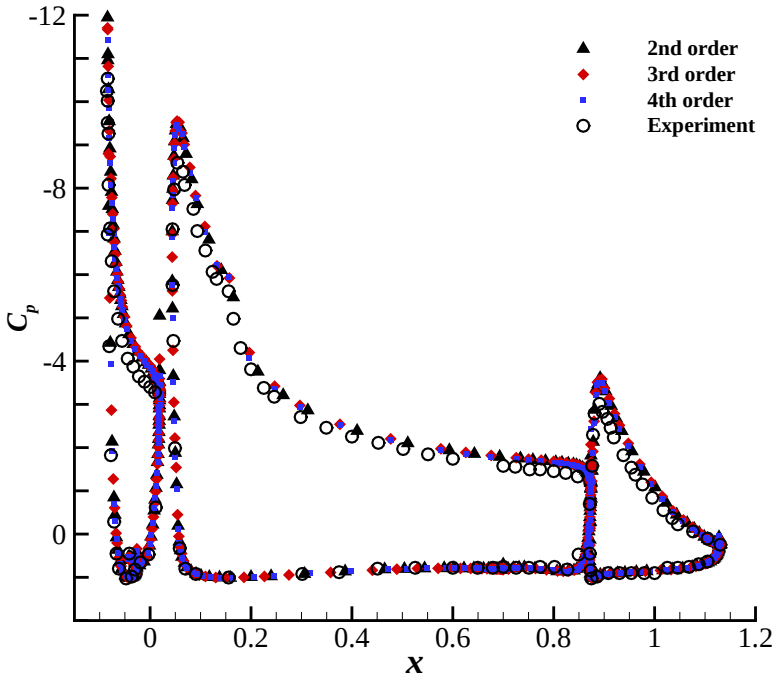


Figure 4.16: Distribution of surface pressure coefficient over multi-element airfoil

Finally, the convergence history of this test case is shown in Figure 4.17. Note that order sequencing has been used to accelerate the convergence to the steady-state solution. For this test case, we solve the linear system of Equation 4.20 with the tighter tolerance of $\eta_l = 10^{-6}$ to obtain a more accurate solution update at each non-linear iteration as suggested by Ceze and Fidkowski [90]. The solver exhibits remarkable robustness for this complex test case as the solutions of all orders of accuracy converge in a small number of non-linear iterations.

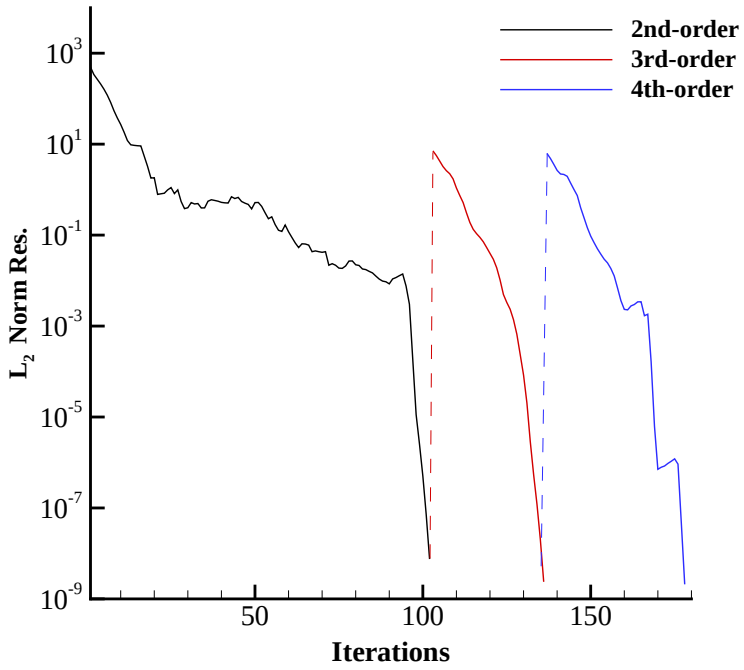


Figure 4.17: Convergence history of finite volume solver for flow over multi-element airfoil

Chapter 5

Adaptive Mesh Refinement and Order Enrichment

As noted, higher-order discretizations of fluid dynamic equations have received a great deal of attention due to their potential advantages in obtaining more accurate solutions with less cost. However, higher-order accuracy is only obtained in the smooth regions of the solution where there is no discontinuity in the solution or gradient. In aerodynamic applications, several sources of discontinuities such as shocks, contact discontinuities, and the turbulence working variable at the edge of boundary layer (e.g., in SA model), can deteriorate the order of accuracy in non-smooth regions and also cause solver failure in some cases. This motivates the idea of simultaneous mesh refinement (h-refinement) and order enrichment (p-enrichment) in higher-order compressible flows solvers. In other words, the order of solution approximation can be increased in those parts of the domain where the solution is smooth whereas the mesh resolution is enhanced in non-smooth regions in which a lower-order discretization is employed.

For compressible flow simulations, the combination of such a strategy with output-based adaptation (known as hp-adaptation) has been used by several Discontinuous Galerkin (DG) solvers where the number of degrees of freedom increases rapidly with the order of discretization. In this way, it is possible to optimally place degrees of freedom within a problem and achieve required accuracy with minimal costs [107, 108, 44]. In addition, it has the advantage of solver robustness by employing a first-order discretization of flow field variables near a shock wave without using slope limiters or artificial dissipation [109, 110]. hp-adaptive methods have been successfully applied to the other variants of Galerkin-based schemes such as hybridized-DG [111] and Petrov-Galerkin [112, 113] and also the correction procedure via reconstruction (CPR) method [114].

In contrast to compact schemes such as DG and its variants, higher-order finite

volume discretizations extend the stencil to obtain a more accurate estimate of the solution. In such schemes, there is no coupling between the number of degrees of freedom and order of discretization as the number of control volumes remains constant on a mesh with a fixed number of elements regardless of the order of accuracy. Nevertheless, a higher-order polynomial for solution approximation requires a larger number of derivatives to reconstruct at each iteration [54]. Also, in the case of using an implicit time advance scheme for the convergence to the steady-state solution, higher-order methods lead to a larger number of non-zero entries in the Jacobian matrix. This increases the memory usage and computational cost due to a larger storage requirement and also solving a denser linear system [49, 7] which must be preconditioned by a factorization method (e.g., Incomplete LU). As a result, *hp*-adaptation methodology, which has originally been designed for compact discretization schemes, has certain advantages for being used in higher-order finite volume flow solvers. In this way, the extra number of derivatives are only reconstructed in those regions where needed and the Jacobian matrix becomes sparser with a smaller bandwidth. Furthermore, this strategy is capable of automatic limiting for flows with inherent discontinuities as we can start from a lower-order solution everywhere and increase the order of polynomial in smooth regions while the mesh is refined near discontinuities. This can be viewed as an alternative to typical limiting approaches where the slope limiter designed to be active only near discontinuities may deteriorate solution accuracy even in smooth regions [55] and/or hamper the convergence to the steady-state due to non-differentiability [115].

This chapter discusses the development of an *hp*-adaptive unstructured finite volume solver and its application to two-dimensional compressible flow problems ranging from inviscid flows governed by the Euler equations to viscous turbulent flows governed by the RANS equations and the Spalart-Allmaras turbulence model. In particular, we adapt the mesh resolution and discretization order based on an estimate of error in the computed solution at each level of refinement.

The quality of the adaptive schemes relies on the accuracy of error estimates. The conventional method of adaptation is based on feature-based criteria that highlight the distinctive features such as shock waves and boundary layers in the flow field [116, 117, 118]. This approach, which is simple and effective in some CFD applications, requires trial and error to determine the appropriate flow features and thus fails to provide a general and robust error estimate [119, 120]. Alternatively, the adaptation

procedure can be carried out using residual-based approaches in which the truncation error of the fluid flow quantities constructs the adaptation criteria [121, 122, 123]. Moreover, output-based adaptation criteria based on solutions to the so-called adjoint (dual) problem, which is derived for an output of interest, has become very mature in recent years [124]. In this type of error estimates, the solution of the adjoint problem is multiplied by the local contribution of the truncation error estimate to provide information about the interaction of the error in different components of the solution [125] and subsequently those locations that require more resolution (typically h-refinement) for a more accurate estimate of the output quantity [126, 127]. It has been shown that the inclusion of the adjoint solution improves the effectiveness of the adaptation procedure over the traditional residual-based approaches [120]. These techniques have been used to perform adaptive mesh refinement (only h-refinement) in second-order unstructured finite volume methods for inviscid [128] and viscous laminar flows [120]. In addition, they have been used in the context of DG methods for inviscid [41], viscous laminar [129] and turbulent RANS [130, 89, 88] simulations.

In our work, we use residual-based and adjoint-based error estimation methods for hp-adaptation in our unstructured finite volume solver. In the former approach, we employ a higher-order residual operator to estimate the truncation error of a lower-order discretization scheme. The magnitude of the estimated truncation error is used as a local error indicator for h- or p-refinement. For the adjoint-based hp-adaptation, we compute the discrete adjoint solution obtained by a single linear system solve at each refinement level and multiply the adjoint solution by the higher-order estimate of the truncation error to find a different error indicator. The adjoint problem is formed by evaluating one order higher operators based on the lower-order solution. In either approach, a certain fraction of control volumes contribute most to the total error are flagged for refinement. The decision for h-refinement versus p-enrichment is based on local smoothness of the primal problem. It is worth mentioning that we allow for non-conforming interfaces (i.e., hanging nodes) in the mesh once we do the h-refinement to be able to handle triangles and quadrilaterals in the same way, including meshes containing both.

5.1 Error Estimation

As mentioned, a reliable error indicator is required for any adaptation procedure. For residual-based adaptation, a local estimate of the error in flow field quantities is used whereas such an estimate is weighted by the solution of an adjoint problem for output-based adaptations. In our work, we use a higher-order operator to obtain an estimate of the truncation error.

Consider the following continuous non-linear problem for which U is the exact solution:

$$R\left(U\right) = 0 \tag{5.1}$$

A lower-order discrete approximation of the exact solution, U_{p-1}, satisfies the lower-order discrete non-linear residual as:

$$R_{p-1}\left(U_{p-1}\right) = 0 \tag{5.2}$$

The exact truncation error for the lower-order discrete problem is defined as the amount by which the discrete lower-order solution does not satisfy the continuous PDE. Such an error property can be estimated by applying a higher-order discrete operator, R_p, on the lower-order solution [125, 131, 132]:

$$R\left(U_{p-1}\right) \approx R_p\left(U_{p-1}\right) \tag{5.3}$$

Therefore, the local contribution of control volume i to the total error can be obtained by the norm of the estimated truncation error in the corresponding control volume:

$$\epsilon_i = \left|R_p\left(U_{p-1}\right)\right|_i \tag{5.4}$$

For adjoint-based adaptation, the error involved in the computation of an output functional based on a lower-order solution, $\mathcal{J}_{p-1}\left(U_{p-1}\right)$, is estimated. For this purpose, a higher-order estimate of the output functional based on a higher-order solution can be obtained by expanding a Taylor series expansion about the lower order solution projected to the higher-order space, $U_{p-1\rightarrow p}$. Note that in the context of finite volume methods, the lower-order solution containing the control volume averages can be identically mapped into the higher-order space ($U_{p-1\rightarrow p} = U_{p-1}$). Therefore,

the higher-order functional can be expanded as:

$$\mathcal{J}_p\left(U_p\right) = \mathcal{J}_p\left(U_{p-1}\right) + \left(\frac{\partial \mathcal{J}_p}{\partial U_p}\right)_{U_{p-1}} \left(U_p - U_{p-1}\right) + ... \tag{5.5}$$

where $\mathcal{J}_p\left(U_{p-1}\right)$ and $\left(\frac{\partial \mathcal{J}_p}{\partial U_p}\right)_{U_{p-1}}$ are the higher-order functional and its sensitivity with respect to the higher-order solution both evaluated at the lower-order solution state. Considering that a higher-order solution is not in hand, a Taylor series expansion can be used for the higher-order residual to eliminate $\left(U_p - U_{p-1}\right)$ in Equation 5.5:

$$R_p\left(U_p\right) = R_p\left(U_{p-1}\right) + \left(\frac{\partial R_p}{\partial U_p}\right)_{U_{p-1}} \left(U_p - U_{p-1}\right) + ... \tag{5.6}$$

The higher-order solution satisfies its corresponding discrete operator, $R_p\left(U_p\right) = 0$. So Equation 5.6 can be re-arranged to solve for the higher-order unknown solution as:

$$\left(U_p - U_{p-1}\right) \approx -\left(\frac{\partial R_p}{\partial U_p}\right)^{-1}_{U_{p-1}} R_p\left(U_{p-1}\right) \tag{5.7}$$

Substituting Equation 5.7 into Equation 5.5 yields the following expression for the estimate of the error in the functional:

$$\mathcal{J}_p\left(U_p\right) - \mathcal{J}_p\left(U_{p-1}\right) \approx -\left(\frac{\partial \mathcal{J}_p}{\partial U_p}\right)_{U_{p-1}} \left(\frac{\partial R_p}{\partial U_p}\right)^{-1}_{U_{p-1}} R_p\left(U_{p-1}\right) \tag{5.8}$$

Next, the higher-order discrete adjoint solution is defined as the variable Z_p such that:

$$\left(\frac{\partial R_p}{\partial U_p}\right)^{T}_{U_{p-1}} Z_P = \left(\frac{\partial \mathcal{J}_p}{\partial U_p}\right)^{T}_{U_{p-1}} \tag{5.9}$$

Note that both sides of Equation 5.9, which show the sensitivity of the higher-order residual and functional to the higher-order solution, are evaluated at the available lower-order solution. The functional error can be re-written in terms of the discrete adjoint solution as:

$$\mathcal{J}_p\left(U_p\right) - \mathcal{J}_p\left(U_{p-1}\right) \approx -Z_p^T\, R_p\left(U_{p-1}\right) \tag{5.10}$$

For adaptation purposes, we require a local error indicator. Therefore, the magnitude of the contribution to the functional error from a particular control volume i can be

approximated as:

$$\epsilon_i = \left| Z_p^T \, R_p \left(U_{p-1} \right) \right|_i \tag{5.11}$$

5.2 Adaptation Methods

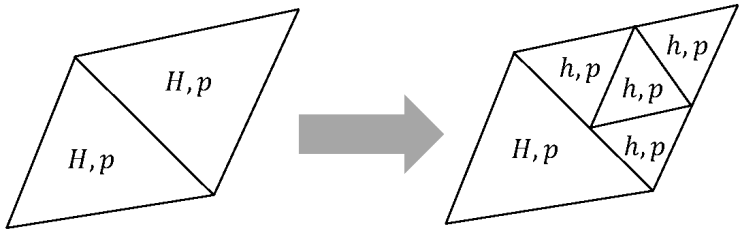
In our *hp*-adaptive process, we use both error indicators of Equation 5.4 and 5.11 to compare the effectiveness of the two estimates. Following Ceze and Fidkowski [108], we flag a certain percentage of control volumes, f^{adapt}, with the largest error indicators for refinement at each step of the adaptive procedure. For this purpose, a sorted list of the control volumes based on the value of their corresponding error indicator from the highest to the lowest is created. A loop over the list is executed and the control volumes are flagged for p-enrichment or h-refinement procedure where appropriate until the pre-specified target ($N_{CV} \times f^{adapt}$) is reached. This ensures that only the control volumes with the highest error magnitude are refined for highly non-uniform error distributions and thus those parts of the computational domain which have negligible contribution to the error in a functional output of interest are excluded from refinement.

As described, accuracy enhancement in each control volume can be attained by decreasing the mesh size (h) or increasing the order of accuracy (p). In what follows, we discuss the details of each method in isolation and then how they are combined to give the hybrid *hp*-adaptive scheme for our unstructured finite volume solver.

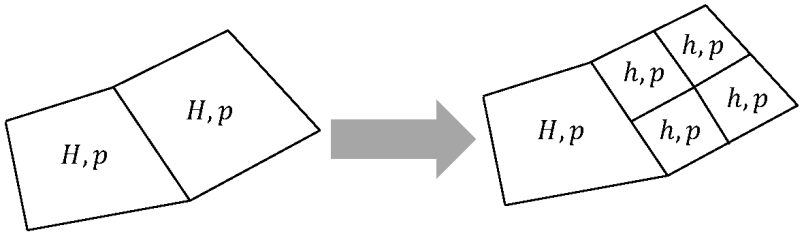
5.2.1 *h*-refinement

To be able to handle triangular and quadrilateral cells in the same way, we use a non-conforming mesh refinement framework where hanging nodes are allowed regardless of the type of elements. In such a refinement, a triangle or quadrilateral is divided into four sub-cells. A triangle is refined using mid-point subdivision so that a node is placed at the mid-point of the faces whereas an additional node is required at the center of the cell for a quadrilateral. Figure 5.1 shows the refinement pattern for both types of the cells in a four-to-one basis recognizing that the sub-cells created through refinement inherit the approximation order from the original cell.

The existence of hanging nodes complicates the computation of flux integrals and also selection of reconstruction stencils. Any face connecting two vertices is identified



(a) Triangular cells



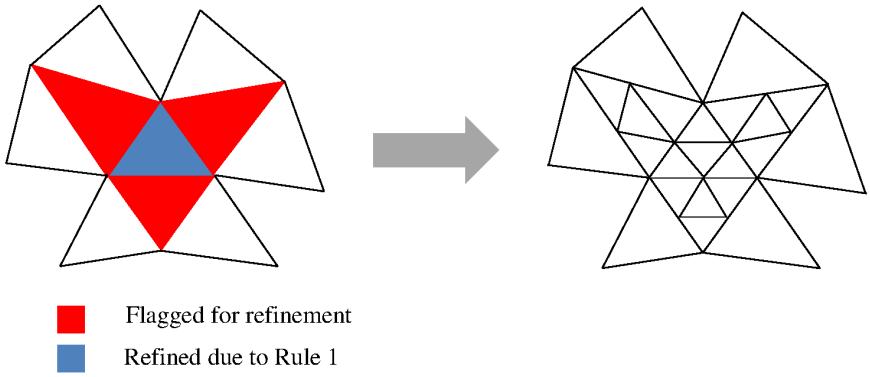
(b) Quadrilateral cells

Figure 5.1: Schematic illustration of h-refinement pattern for 2D cells

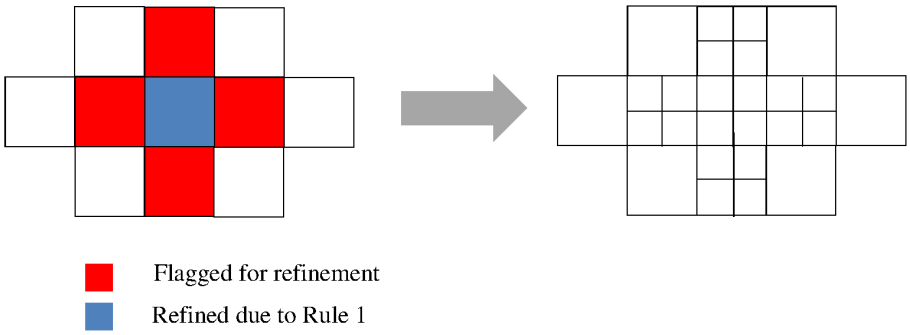
as a unique face on which integration quadrature points are placed. In addition, the indices of the cells at the two sides of such a face is stored to retrieve the indices of the first layer of neighbors in the process of stencil selection. In this approach, a triangle with one hanging node is represented by four faces (or similarly a quadrilateral with one hanging node is represented by five faces).

To provide a smooth distribution of cell size throughout the mesh, we set up two standard rules for the refinement. First, a cell is automatically flagged for refinement if all of its faces are split due to the refinement of the neighbors. This rule has the advantage of limiting the number of hanging nodes to two and three for triangles and quadrilaterals, respectively in addition to removing potential un-refined holes from the mesh. Figure 5.2 shows the implementation of this rule for triangular and quadrilateral cells. Note that the red cells are refined due to large error indicator or non-smooth primal while the blue cell is refined just because all of its neighbors have been refined.

The second rule enforces that all non-conforming faces have a 2:1 face length ratio by controlling the subdivision of half length non-conforming faces. If a fine cell with



(a) Triangular cells

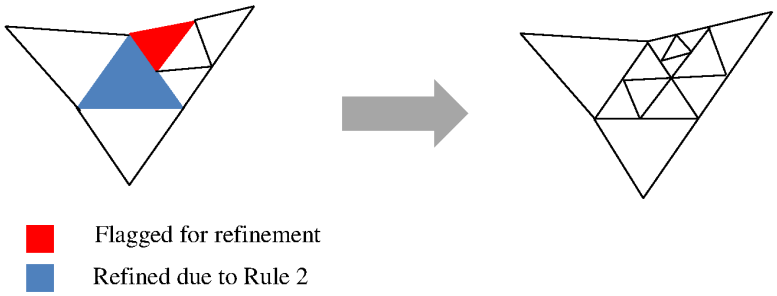


(b) Quadrilateral cells

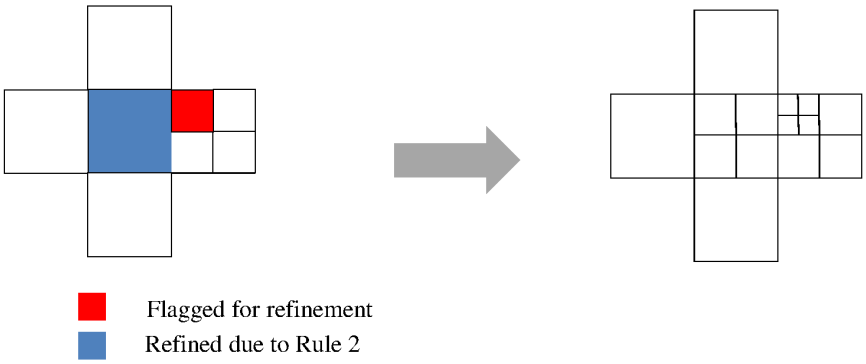
Figure 5.2: Schematic illustration of the first rule for the refinement of 2D cells

a half length face needs to be refined, the coarse cell on the other side containing the half length face must be divided as well to balance the face length ratio between adjacent cells. Figure 5.3 depicts the implementation of the second rule for triangular and quadrilateral cells. In this case, the red cells are those need to be refined due to a large error indicator or non-smooth primal while the blue cells are refined to keep the 2:1 face length ratio.

As described earlier, higher-order simulation on highly anisotropic meshes requires curving the interior faces of the mesh to prevent intersection with higher-order boundary representations. When considering non-conforming meshes, special attention must be paid to how to curve the cells that have hanging nodes. One tedious way is taking the linear mesh, applying the necessary refinement and re-curving the mesh



(a) Triangular cells



(b) Quadrilateral cells

Figure 5.3: Schematic illustration of the second rule for the refinement of 2D cells

by solving the linear elasticity equation at each level of refinement. Considering that we use the continuous Galerkin finite element method to solve the elasticity equation, this approach necessitates the enforcement of a constraint over non-conforming interfaces to ensure that the faces defined from both sides conform to the same mapping. This comes from the fact that the refined faces on a non-conforming interface contain different number of degrees of freedom from each side (Figure 5.4). Alternatively, we can assume that the initial mesh represents the geometry sufficiently accurately and no further geometry information is added throughout the refinement. In this approach, we only curve the initial conforming mesh and use the mapping information to create higher-order non-conforming meshes at the next levels. The mapping from each linear cell of the initial mesh to its corresponding cubic cell is computed and stored. At each level of refinement, the top parent for each sub-cell is found. The cubic mapping corresponding to the top parent is used to create the cubic sub-cells

within the cubic top parent cell without the loss of geometry due to non-conforming mapping.

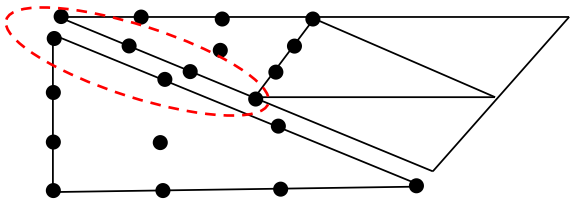


Figure 5.4: Different number of degrees of freedom for linear elasticity problem on half-length non-conforming face

5.2.2 *p*-enrichment

The implementation of p-enrichment is easier as it does not lead to any geometrical complexity and remains the same for triangular and quadrilateral cells. Recall that the primitive variables of the flow field are approximated throughout a control volume by a Taylor series expansion about the reference point as:

$$\begin{aligned}\phi_i^R(x,y) &= \phi|_i + \left.\frac{\partial\phi}{\partial x}\right|_i (x-x_i) + \left.\frac{\partial\phi}{\partial y}\right|_i (y-y_i) + \left.\frac{\partial^2\phi}{\partial x^2}\right|_i \frac{(x-x_i)^2}{2} \\ &\quad + \left.\frac{\partial^2\phi}{\partial x\partial y}\right|_i (x-x_i)(y-y_i) + \left.\frac{\partial^2\phi}{\partial y^2}\right|_i \frac{(y-y_i)^2}{2} + \ldots \end{aligned} \tag{5.12}$$

To enhance the accuracy by p-enrichment, the order of this polynomial is incremented by one and a larger least-squares system with more control volumes in the stencil is solved to find the derivatives of the primitive variables at the reference point. Figure 5.5 shows the p-enrichment of a cell from $p = 1$ (second-order) to $p = 2$ (third-order) schematically.

We also apply some rules over the order of accuracy which mimic the rules used in h-refinement for uniform distribution of cell size and lead to uniform distribution of order of accuracy throughout the mesh. First, if the order of accuracy in all neighboring cells is larger, we increment the order within the cell by one. Second, we modify the order of polynomial in neighboring control volumes not to end up with a jump of more than one order between adjacent cells.

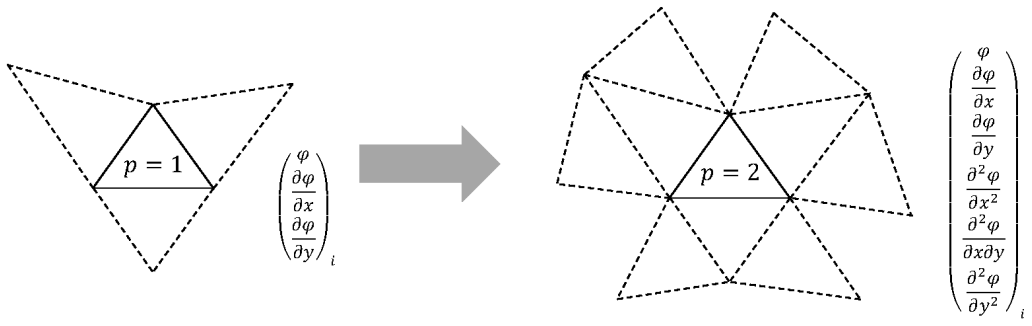


Figure 5.5: Schematic illustration of p-enrichment for an arbitrary cell from second-order to third-order

The number of integration quadrature points is determined based on the maximum order ($\max(p^+, p^-)$) of the two cells on each side of a face to ensure that the higher-order reconstructed polynomial is integrated sufficiently accurately along the faces.

5.2.3 *hp*-refinement

The final step in the adaptation procedure is the decision between h-refinement and p-enrichment. As mentioned, this is typically decided based on the smoothness of the primal solution [111, 110, 109] so that the mesh size is reduced near discontinuities and order is increased in the smooth regions of the primal solution. In our work, the smoothness is determined by an inter-element jump indicator designed for shock detection in a DG solver [133]. This indicator has also been successfully used for the decision between mesh refinement and order enrichment in other *hp*-adaptive DG solvers [109, 110] and can be easily fit in our unstructured finite volume solver. The value of the jump for an arbitrary quantity of the flow field, ϕ, in a cell (cell i) is given as:

$$S_i = \frac{1}{|\partial\Omega_i|} \oint_{\partial\Omega_i} \left| \frac{\phi^+ - \phi^-}{\frac{1}{2}(\phi^+ + \phi^-)} \right| ds \tag{5.13}$$

where ϕ^+ and ϕ^- are the reconstructed solutions at the two sides of a quadrature point, $|\partial\Omega_i|$ is the perimeter of the cell and ds is the infinitesimal length along the faces of the cell. For smooth solutions, the jump in the reconstructed solution and thus S_i are expected to be small whereas the indicator should return a value of $\mathcal{O}(1)$

near a discontinuity such as a shock wave since the jump and average flow properties are of the same order of magnitude. Therefore, the choice between whether to apply h-refinement or p-enrichment is made by:

$$\begin{cases} S_i \geqslant \frac{1}{\mathcal{K}_\phi}, & h\text{-refinement} \\ S_i < \frac{1}{\mathcal{K}_\phi}, & p\text{-enrichment} \end{cases} \tag{5.14}$$

where $\mathcal{K}_\phi$ is a constant required to be large enough to capture discontinuities properly. In this work, we apply the jump indicator only on pressure for non-turbulent flows and on pressure *and* turbulence working variable of SA model for turbulent flows. The proper values of the jump indicator constants are found by several numerical experiments. In our solver, the values of of $200 \leq \mathcal{K}_P \leq 400$ for pressure and $10 \leq \mathcal{K}_{\tilde{\nu}} \leq 20$ for turbulence working variable seem to be effective choices for our problems. However, the best choices of constants in these ranges can be obtained by considering the flow regimes governed by important non-dimensional groups (Mach and Reynolds number).

5.3 Numerical Results

The *hp*-adaptation method proposed is evaluated for four different flow conditions over the NACA 0012 geometry: inviscid subsonic and transonic flows governed by the Euler equations, laminar subsonic flow governed by the Navier-Stokes equations and turbulent subsonic flow governed by the RANS equations and SA-negative turbulence model. In each case, the efficiency of the residual-based and adjoint-based *hp*-adaptive schemes are compared with second- and higher-order uniform refinements in terms of the number of degrees of freedom per equation and also CPU time required to obtain a grid-converged value of an output of interest. The presented number of degrees of freedom is the number of control volumes in each level and the wall clock time for the various test cases is based on simulations on a single core of an i7-4790 (3.60 GHz) CPU. In the adjoint-based adaptation method, the time includes the solution of both the primal and adjoint whereas only the primal solve time is included for the uniform refinement and residual-based adaptations.

Considering that our solver supports up to fifth-order discretizations ($p = 4$), once a control volume with the polynomial degree of $p_{\max} = 3$ is flagged for extra

refinement in smooth regions of the solution, we perform h-refinement instead of p-enrichment since we need one order higher discrete operator for error estimation (Equations 5.4 and 5.11). However, this strategy fails for highly anisotropic meshes with curvature and leads to solver failure. For such problems, the highly anisotropic cells close to curved walls (those using the curvilinear coordinate system as described in Chapter 3) are taken out of the error indicator list once they reach $p = 3$. Instead, some other control volumes with lower discretization order and/or other regions of the mesh are flagged for p-enrichment or h-refinement as appropriate. In this way, we can still reduce the error without breaking the structure of anisotropic cells near the wall.

5.3.1 Inviscid Subsonic Flow

As our first test case, we consider the inviscid subsonic flow over the NACA 0012 with chord length of $c = 1$ and zero thickness at the trailing edge. The flow is characterized by the free stream Mach number of $Ma_\infty = 0.5$ and angle of attack of $\alpha = 2^\circ$. For residual-based adaptation, we use the L_2-norm of the truncation error obtained by the application of one order higher residual operator on all four components of the solution for each control volume. For adjoint-based adaptation, we consider the target functional of the pressure drag coefficient given by:

$$\mathcal{J}(U) = \int_{\partial\Omega} \psi \cdot (P\hat{n})\, ds \tag{5.15}$$

where $\hat{n}$ is the outward unit normal vector and $\psi = \frac{1}{C_\infty}(\cos\alpha, \sin\alpha)^T$along the wall boundaries and 0 everywhere else. C_∞ is a normalized reference value defined as:

$$C_\infty = \frac{1}{2}\gamma Ma_\infty P_\infty c \tag{5.16}$$

In the hypothetical case of infinitely far outer boundary, the drag force for this isentropic inviscid flow must converge to zero. However, we place the outer boundary $100c$ away from the wall boundaries and thus the pressure drag coefficient converges to a finite small value. The adaptation process starts with the second-order solution on a baseline coarse mesh with $2,776$ triangular cells. At each level of the adaptive procedure, $f^{adapt} = 15\%$ of control volumes with the largest error indicators are flagged for p-enrichment or h-refinement. Considering that the solution of this prob-

lem is smooth almost everywhere in the computational domain, we choose the upper bound of the pressure smoothness indicator, $\mathcal{K}_P = 200$, for the decision between p-enrichment and h-refinement.

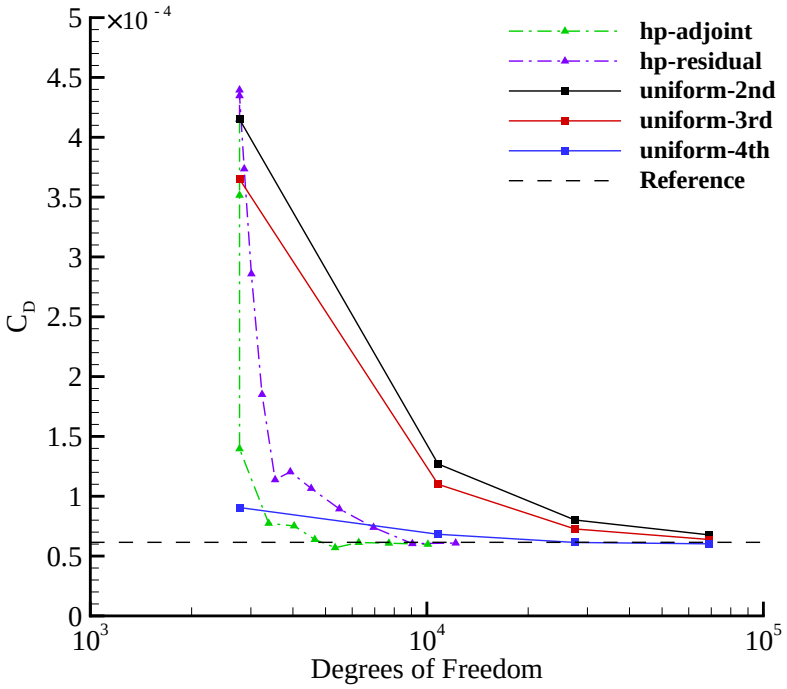
The efficiency of the hp-adaptive refinement schemes is shown in Figure 5.6 in comparison with uniform refinement of different discretization order. A reference value for this problem is obtained by a fourth-order solution on a mesh with $197,891$ cells. The adjoint-based adaptation outperforms the other refinement strategies as it converges to the grid converged drag coefficient after a few cycles with a smaller number of degree of freedom. The change in the values of the functional becomes small after 6 cycles of adaptations and remains almost constant in the last three consecutive cycles. The residual-based adaptation converges to the same value but after more refinement cycles and with a larger number of control volumes. In terms of the CPU time, the adjoint-based method takes slightly longer due to the extra adjoint solver. For this problem with a smooth solution, the fourth-order uniform refinement produces reasonably accurate output values with sufficient number of degrees of freedom which takes longer in term of CPU time. However, the second- and third-order uniform refinement procedures are far from the grid converged value except for the finest mesh with about $80,000$ control volumes.

Figure 5.7 illustrates the baseline mesh and also the order of accuracy and mesh resolution after 6 cycles of the adjoint-based hp-adaptation for this test case. Note that the region near the leading edge in which the flow experiences high gradients towards the stagnation point is refined the most considering that the resolution of the initial mesh is not sufficient there. It is also worth-mentioning that the order of accuracy far from the wall remains equal to 2 which leads to less numerical complexity compared to uniformly higher-order discretizations .

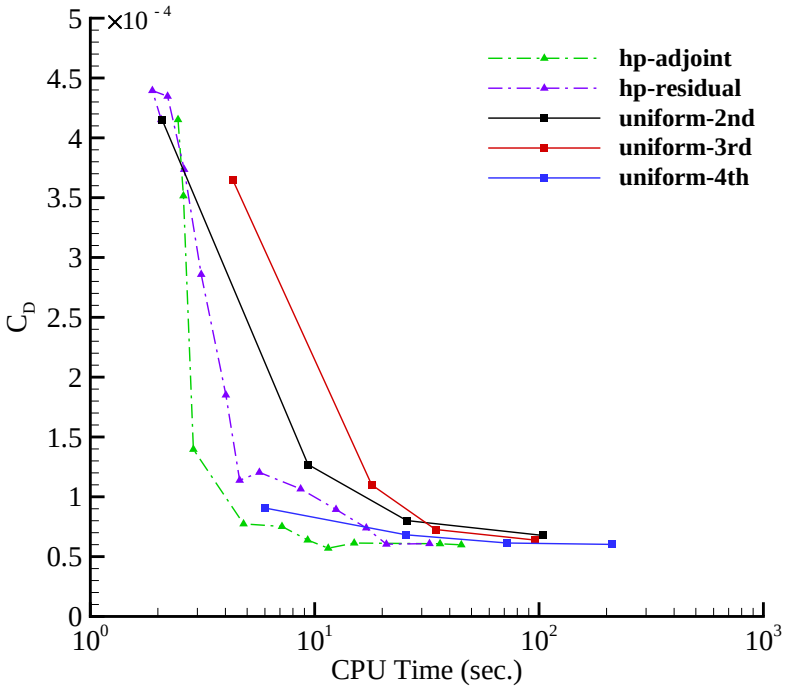
In addition, the contours of the Mach number on the two meshes with the prescribed order of accuracy for each control volume is shown in Figure 5.8. As expected, the hp-adapted mesh produces a smoother distribution of the Mach number near the airfoil.

5.3.2 Inviscid Transonic Flow

As our second test case, we consider the transonic inviscid flow around the NACA 0012 with free stream Mach number of $Ma_\infty = 0.8$ and angle of attack of $\alpha = 1.25°$.

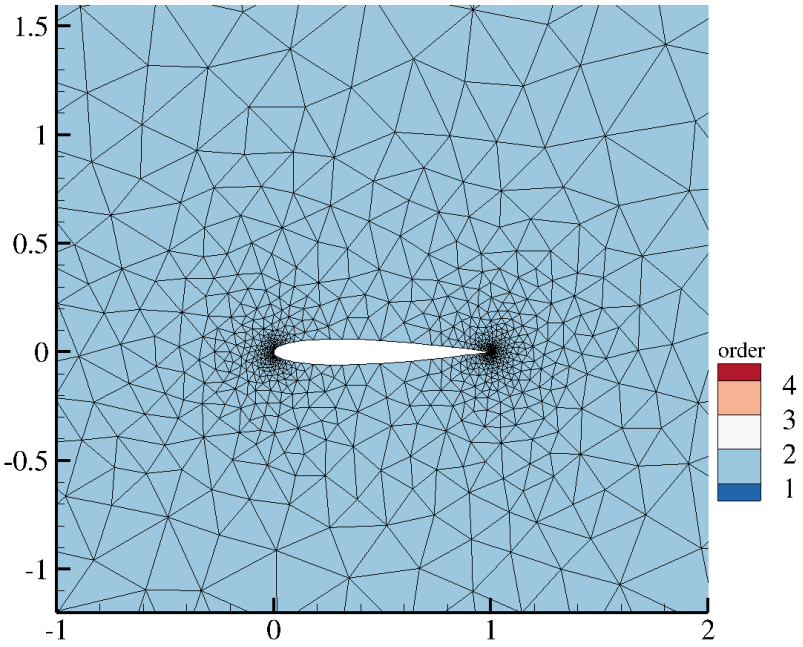


(a) Drag coefficient evolution with respect to degrees of freedom

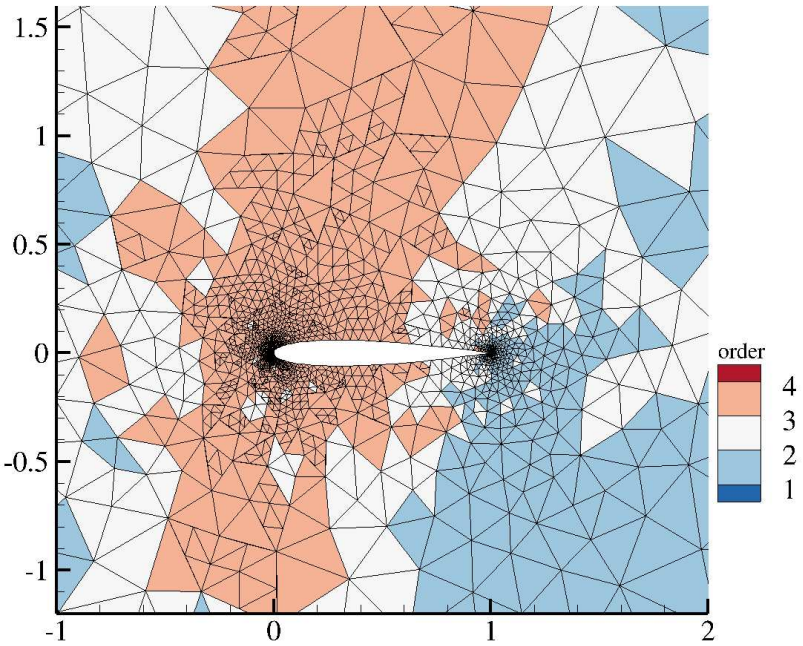


(b) Drag coefficient evolution with respect to CPU time

Figure 5.6: Convergence of drag coefficient for inviscid subsonic test case ($Ma_\infty = 0.5$ and $\alpha = 2°$)

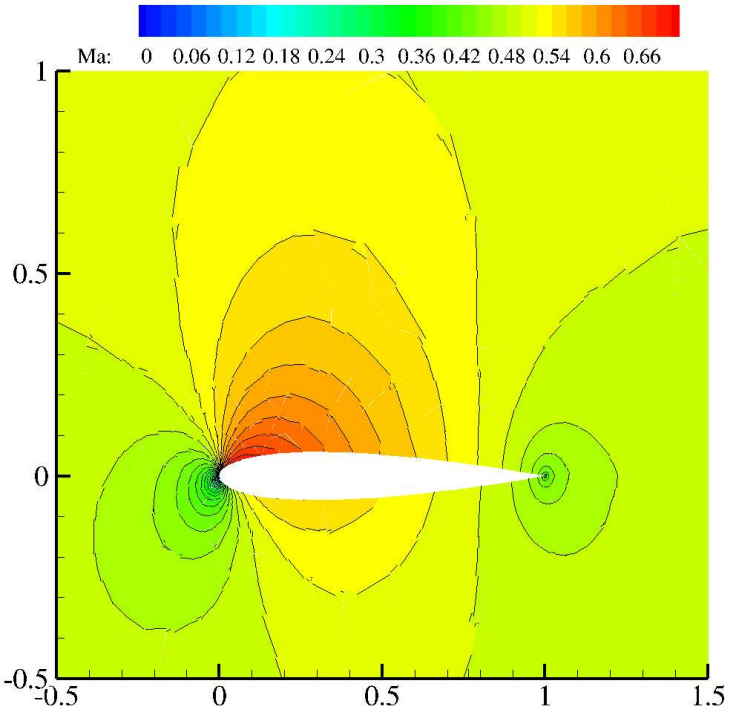


(a) Baseline mesh with 2,776 triangular cell (purely second-order)

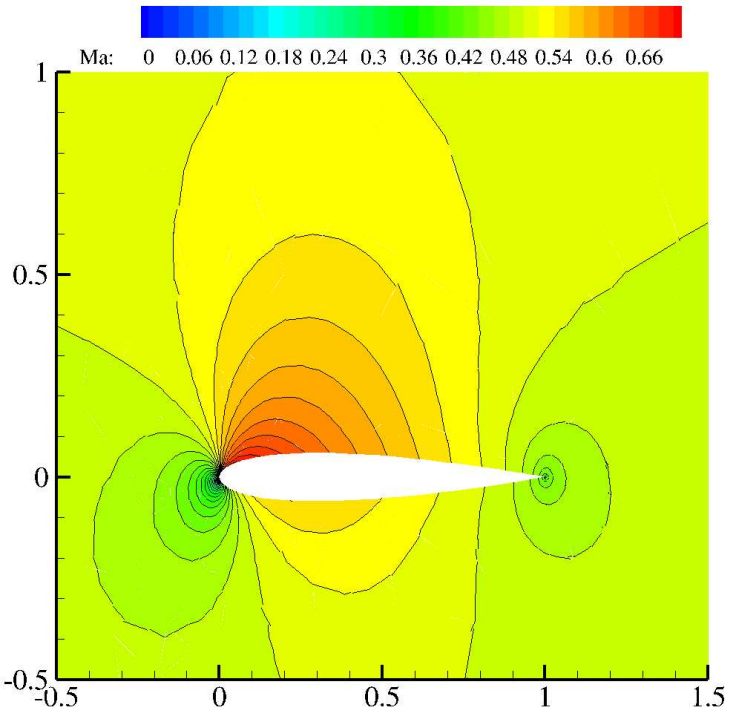


(b) adjoint-based *hp*-adapted after 6 adaptation cycles (second- to fourth-order)

Figure 5.7: Mesh resolution and discretization order for inviscid subsonic test case ($Ma_\infty = 0.5$ and $\alpha = 2°$)



(a) Initial mesh with purely second-order discretization



(b) adjoint-based *hp*-adapted after 6 levels of adaptation

Figure 5.8: Contours of Mach number for inviscid subsonic test case ($Ma_\infty = 0.5$ and $\alpha = 2°$)

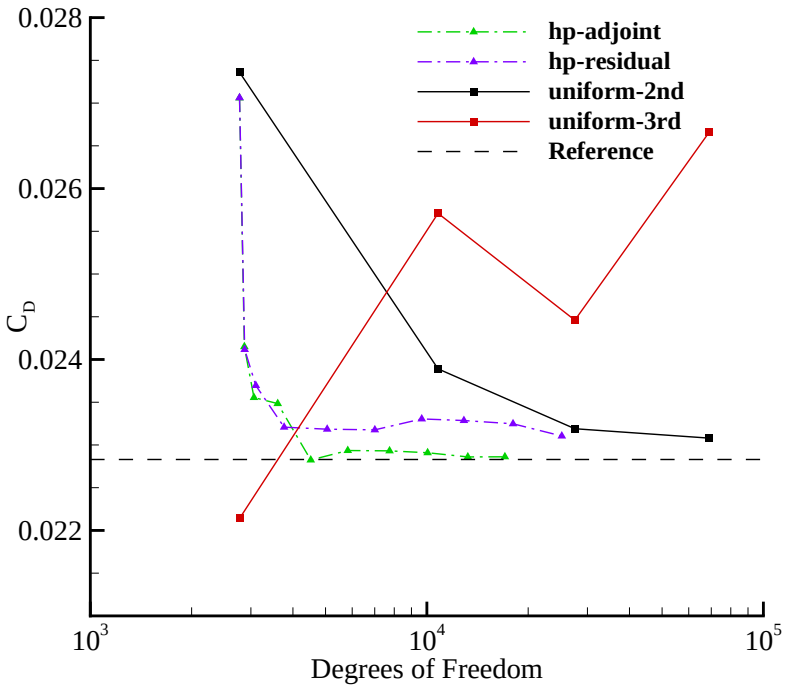
For this test case, a strong shock and a weak shock appear on the upper and lower surfaces of the airfoil, respectively. Therefore, this is a good example to demonstrate the capability of the proposed *hp*-adaptation method for shock capturing. Similar to the inviscid subsonic case, we use the L_2-norm of the truncation error for residual-based adaptation and pressure drag coefficient as the functional of interest for the adjoint-based method. The other adaptation parameters are chosen as $f^{adapt} = 15\%$ and $\mathcal{K}_P = 300$. As described, the *hp*-adaptation framework can be an alternative to slope limiting procedure for compressible problems with shocks and so can be used to enhance the robustness of an unstructured finite volume solver. This suggests we allow for first-order approximations near the shock as traditional slope limiters reduce the order of accuracy to one close to singularities. Although it is possible to start from a first-order solution everywhere and increasing the order in smooth regions of the flow field, this approach leads to the over-refinement of the initial mesh due to the the non-smooth behavior of the first-order solution. This is also repeated in the next cycles since the order is not increased in all of the *h*-refined cells (with order = 1). Instead, we start from a purely second-order solution on the same baseline mesh as the subsonic test case but we reduce the order of discretization by one (provided that the order is larger than 1) whenever a cell is flagged for *h*-refinement due to non-smooth primal solution. In this way, we can attain the first-order approximation near the singularities after the first cycle.

For uniform refinement studies, we only employ the second- and third-order discretizations with the slope limiter of Venkatakrishnan [134] and avoid the fourth-order discretization since it suffers from stability issues for such a problem with a strong shock. Figure 5.9 compares the efficiency of different refinement scenarios for this case. A reference drag coefficient is obtained by a second-order solution of this problem on an unstructured mesh with $308,470$ cells. It is evident that the third-order discretization is not converging to a fixed value as the slope limiter degrades the accuracy of the solution. Furthermore, the tunable parameter in the slope limiter is adjusted for each mesh to get the solution to converge and this exacerbates the asymptotic convergence to a single value. Therefore, increasing the order of accuracy for problems with locally non-smooth solutions does not lead to more accurate answers as expected. On the other hand, the adjoint-based *hp*-adaptive method reaches the grid converged value after a small number of adaptation cycles so that the change in the drag coefficient becomes less than 0.2% in the last three consecutive cycles.

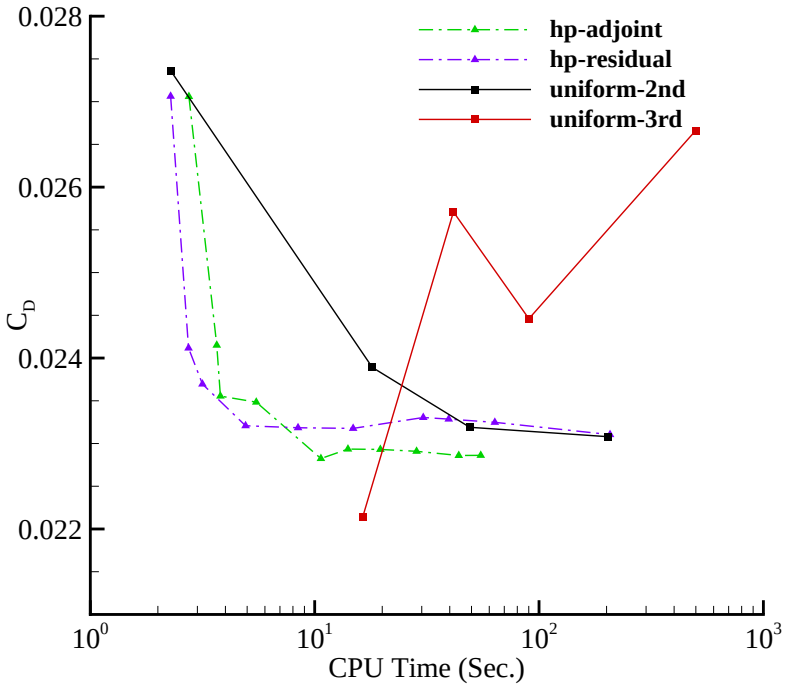
Note that such a good estimate of the drag coefficient is found on an *hp*-adapted mesh with about $20,000$ control volumes. In term of the CPU time, the adjoint-based adaptation obviously outperforms uniform refinement considering the fact that the drag predicted by the second-order scheme on the finest mesh is not still sufficiently close to the reference value and takes about 5 times longer. The residual-based adaptation is also approaching the reference value but with a larger number of refinement cycles, degrees of freedom and CPU time compared the adjoint-based.

The difference between the drag coefficients predicted by the two adaptation error indicators can be explained by the refinement pattern obtained from each one. Figure 5.10 compares the mesh resolution and local order of accuracy in the last cycle of the adjoint-based and residual-based adaptations. The residual-based indicator mainly focuses on the strong shock of the upper surface with minimal refinement of the leading and trailing edges. This behavior was previously reported by Woopen *et al.* [135] in residual-based mesh adaptation for their hybridized discontinuous Galerkin (HDG) method. In their study, the drag coefficients obtained on the residual-based adapted meshes differ by a constant amount from the values of the adjoint-based adapted meshes after a few refinement cycles. Conversely, the adjoint-based error indicator exhibits a more accurate refinement pattern as the the area close to the weak shock on the lower surface is refined properly in addition to the sufficient refinement of the leading and trailing edges. This implies that weighting the local residuals with the discrete adjoint solution provides a more accurate error indicator that captures all important features of the flow field during the refinement process. It should be noted that the first-order approximation is maintained close to the upper surface strong shock as magnified in Figure 5.10a which is ideal for regions with singular solution. Such a strategy is advantageous as it results in the efficiency of the solver in obtaining a grid converged output value with small number of degrees of freedom and also its robustness by employing a low-order solution in the singular regions of the flow field.

Figure 5.11 shows the contours of Mach number close to the airfoil for the initial and final adjoint-based adapted meshes of this test case. As expected, the two shocks sharpen as we proceed and increase the mesh resolution in their vicinity. Also, the use of first-order approximation in these regions prevents any overshoots in the final solution. In the smooth regions of the flow, the solution becomes smoother as we provide extra resolution by *p*-enrichment.

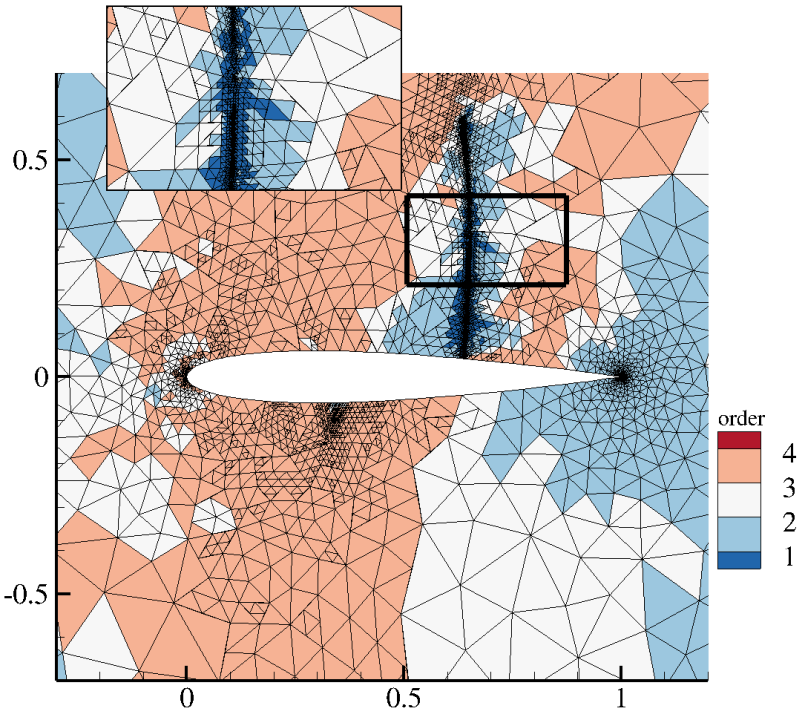


(a) Drag coefficient evolution with respect to degrees of freedom

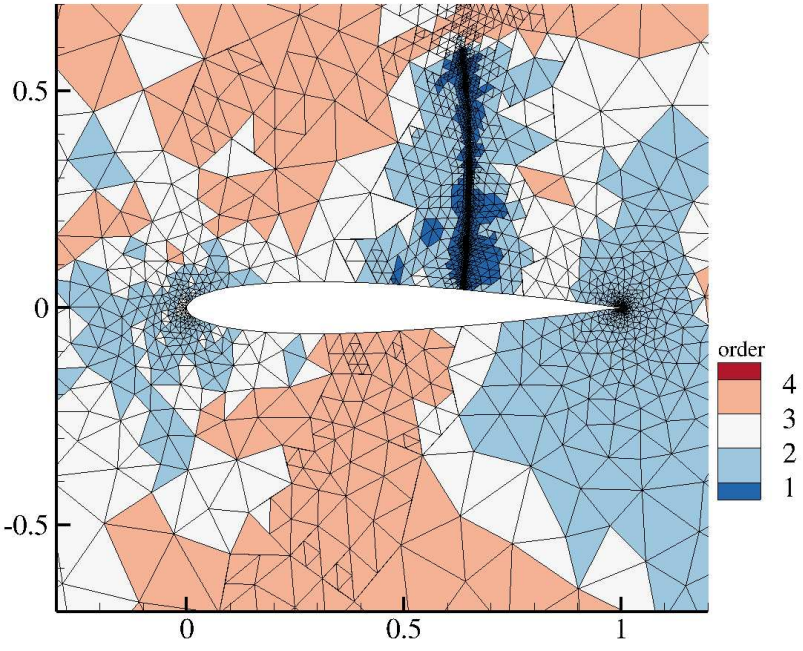


(b) Drag coefficient evolution with respect to CPU time

Figure 5.9: Convergence of drag coefficient for inviscid transonic test case ($Ma_\infty = 0.8$ and $\alpha = 1.25°$)

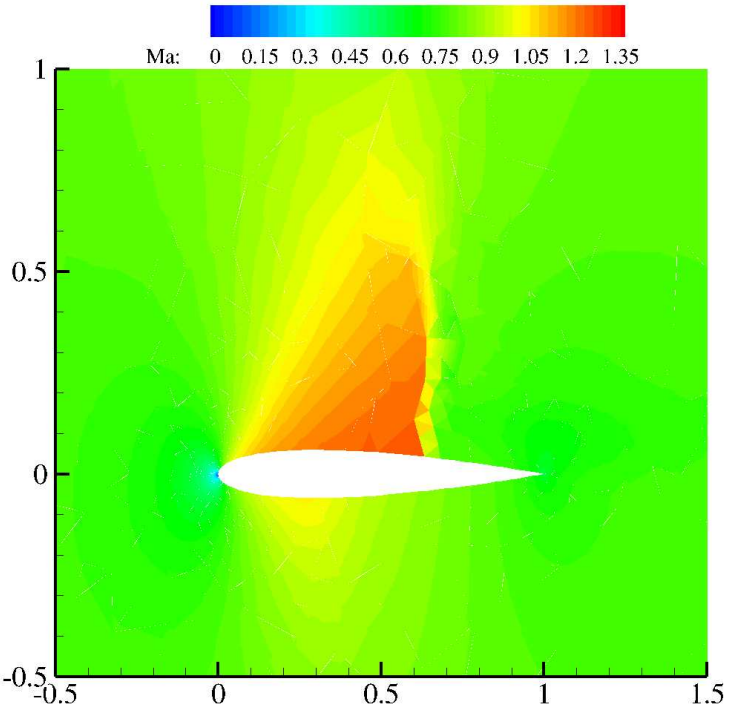


(a) Adjoint-based adaptation

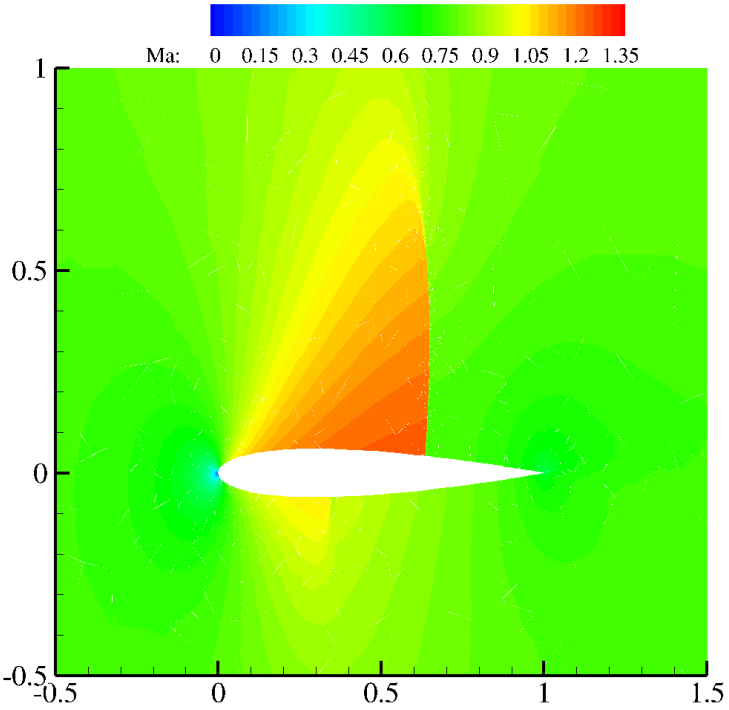


(b) Residual-based adaptation

Figure 5.10: Final *hp*-adapted mesh and order for inviscid transonic test case ($Ma_\infty = 0.8$ and $\alpha = 1.25°$)



(a) adjoint-based *hp*-adapted after 1 adaptation cycle (first adapted mesh)



(b) adjoint-based *hp*-adapted after 9 adaptation cycles (final adapted mesh)

Figure 5.11: Contours of Mach number for inviscid transonic test case ($Ma_\infty = 0.8$ and $\alpha = 1.25°$)

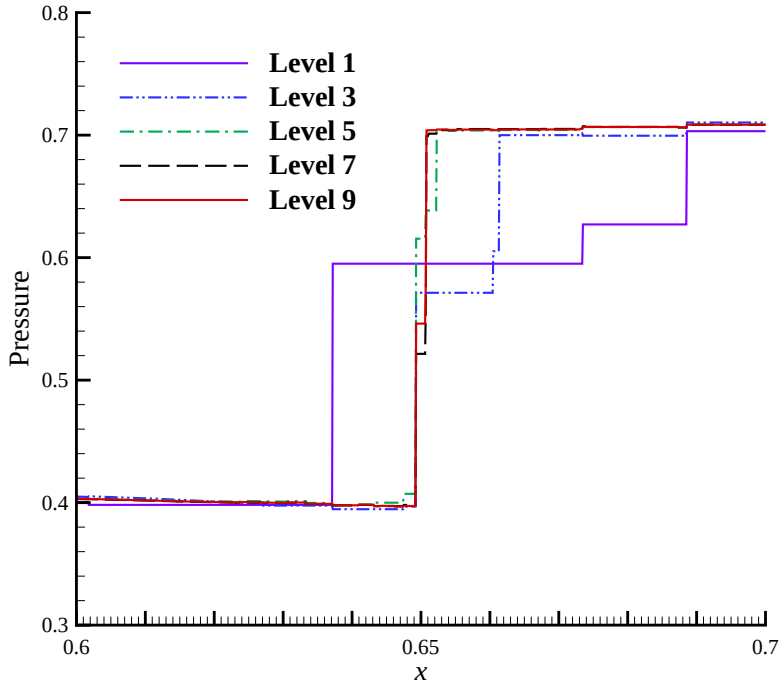


Figure 5.12: Comparison of pressure profiles near the upper surface strong shock between adjoint-based *hp*-adapted meshes for inviscid transonic test case at $y = 0.3$

Figure 5.12 compares the pressure profiles near the strong shock at $y = 0.3$ between different levels of adjoint-based *hp*-adaptation. On the initial mesh, the shock spans a wide distance with several jumps in the magnitude of pressure. However, it gradually becomes thinner and sharper such that the shock computed on the last level is around 50 times thinner than the first one. This plot reveals the advantage of *hp*-adaptation in shock capturing in flows with singularities.

Also, Figure 5.13 compares the pressure profile at $y = 0.3$ between adjoint-based and residual-based methods after the final cycle of adaptation. Note that the residual-based adaptation refines the region near the strong shock more than the adjoint-based as expected. This can be understood by the larger number of piecewise constant pressure values (first-order solutions) in the vicinity of the shock.

5.3.3 Laminar Subsonic Flow

Now we turn our attention to viscous laminar subsonic flow around the NACA 0012 airfoil with a free stream Mach number of $Ma_\infty = 0.5$, Reynolds number of $Re = 5000$

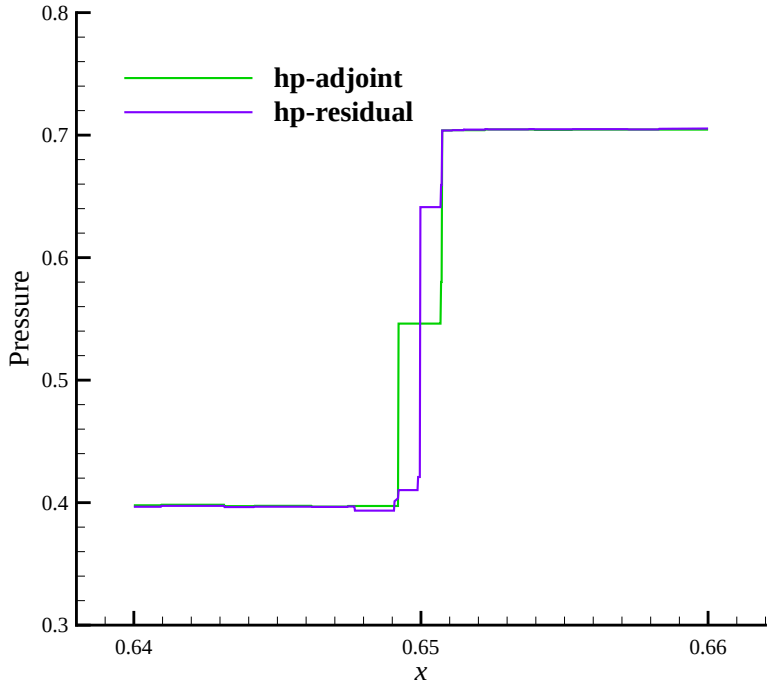


Figure 5.13: Comparison of pressure profiles near the upper surface strong shock between adjoint-based and residual-based *hp*-adaptation at $y = 0.3$

with constant dynamic viscosity and angle of attack of $\alpha = 1°$. In this flow, a thin boundary layer appears on the surface of the airfoil and is followed by a wake downstream of the trailing edge. Similarly, the L_2-norm of the truncation error is used for residual-based adaptation, whereas the drag coefficient obtained as the total of the pressure and viscous drag coefficients is employed for the adjoint-based method. The total drag functional is defined as:

$$\mathcal{J}\left(U, \nabla U\right) = \int_{\partial\Omega} \psi \cdot \left(P\hat{n} - \tau\hat{n}\right) \, ds \tag{5.17}$$

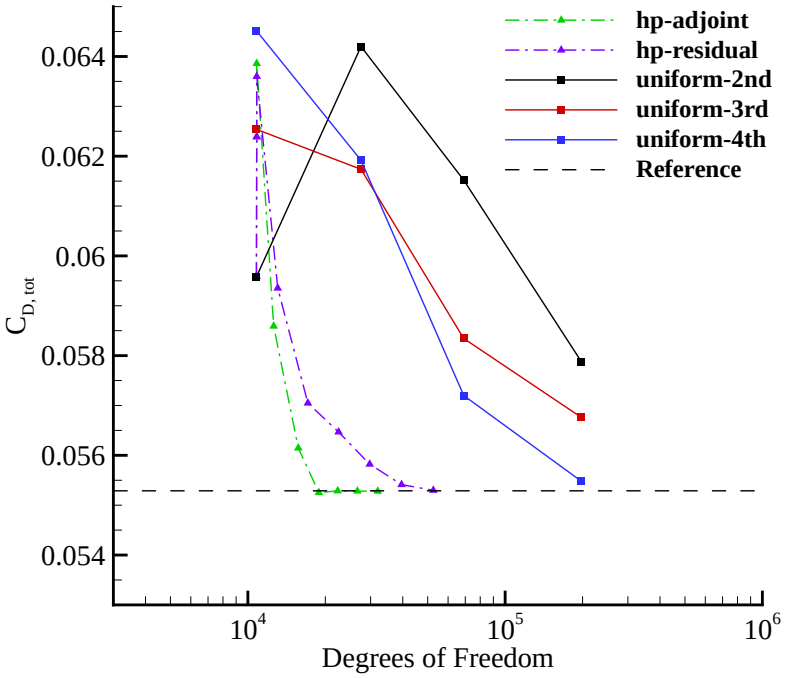
in which $\hat{n}$ is the outward unit normal vector, τ is the viscous stress tensor and $\psi = \frac{1}{C_\infty}\left(\cos\,\alpha, \sin\,\alpha\right)^T$along the wall boundaries and 0 everywhere else. We start the adaptations with a second-order solution on an unstructured mesh with $10,797$ triangular cells. The adaptation parameters for this problem with smooth solution are set to $f^{adapt} = 10\%$ and $\mathcal{K}_P = 200$.

Figure 5.14 compares the efficiency of the *hp*-adaptation methods with uniform refinement of second-, third- and fourth-order discretizations. For this purpose, the

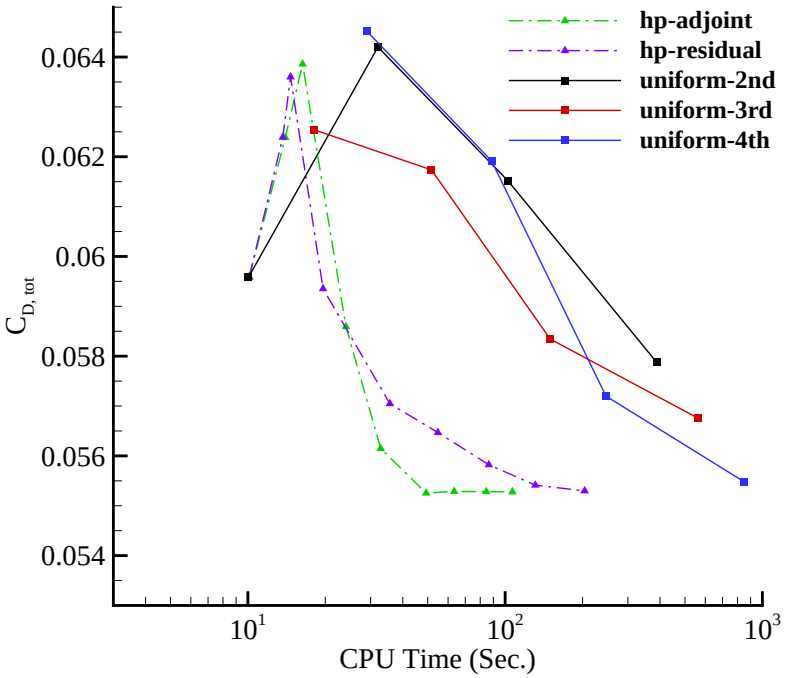
convergence of the total drag coefficient versus the number of degrees of freedom and CPU time is plotted. For this problem, a reference drag coefficient is found by a fourth-order discretization on a mesh with $308,470$ cells. The advantage of *hp*-adaptation over uniform refinement is very clear for this test case with thin shear layers: the *hp*-adapted meshes reach grid convergence with about as many degrees of freedom as the first uniform refinement. At this resolution, the second-order discretization on the uniformly refined mesh has not yet begun to improve the results. The higher than second-order schemes (particularly fourth-order) converge faster as expected for this problem with smooth solution; however, the CPU time required to get close to the reference drag coefficient is more than 6 times larger compared with *hp*-adaptation. Similar to the previous test cases, the adjoint-based adaptation is more successful than the residual-based. Even though the residual-based converges to the same output value, it adds a number of unnecessary degrees of freedom in each cycle that do not improve functional accuracy. Note that the increase in the number of degrees of freedom for this problem typically originates from those fourth-order control volumes which need more resolution and thus are refined via *h*-refinement. On the other hand, the adjoint-based adaptation indicator optimally increases the resolution to be able to deliver a very accurate functional for which the adaptation is performed. It is worth mentioning that the time required for solving the discrete adjoint problem is a small portion of the total CPU time. Interestingly, the output value becomes almost constant in the last three cycles of the adjoint-based adaptation which highlights its efficiency.

Figure 5.15 depicts the mesh resolution and local order for the control volumes of the initial mesh and also adjoint-based *hp*-adapted mesh after 7 cycles. As required, the order of accuracy increases in the boundary layer and wake regions via *p*-enrichment. Since some of the control volumes in these regions still contribute to the error the most even after the *p*-enrichment, we apply *h*-refinement so that we can reduce the error in the next cycles effectively and converge to the reference output value as fast as possible. To provide such a resolution with uniform refinement, we require considerably more cells in the mesh, most of which are far from the airfoil and do not contribute to the total error. Note that the mesh size and order do not change sufficiently far from the viscous dominated regions in the *hp*-adapted mesh.

Finally, the contours of Mach number on the initial and final *h*p-adapted meshes are shown in Figure 5.16. The longer wake region and smoother velocity distribution

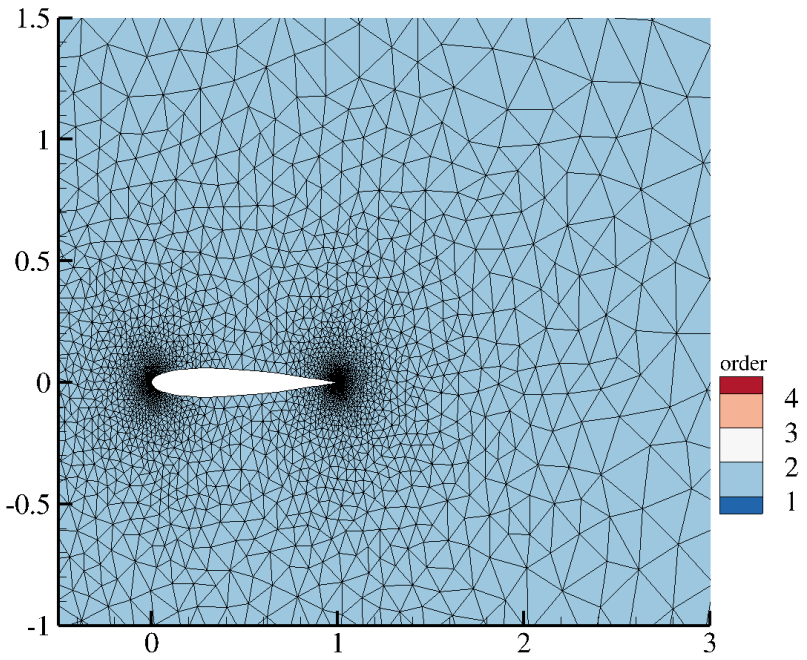


(a) Drag coefficient evolution with respect to degrees of freedom

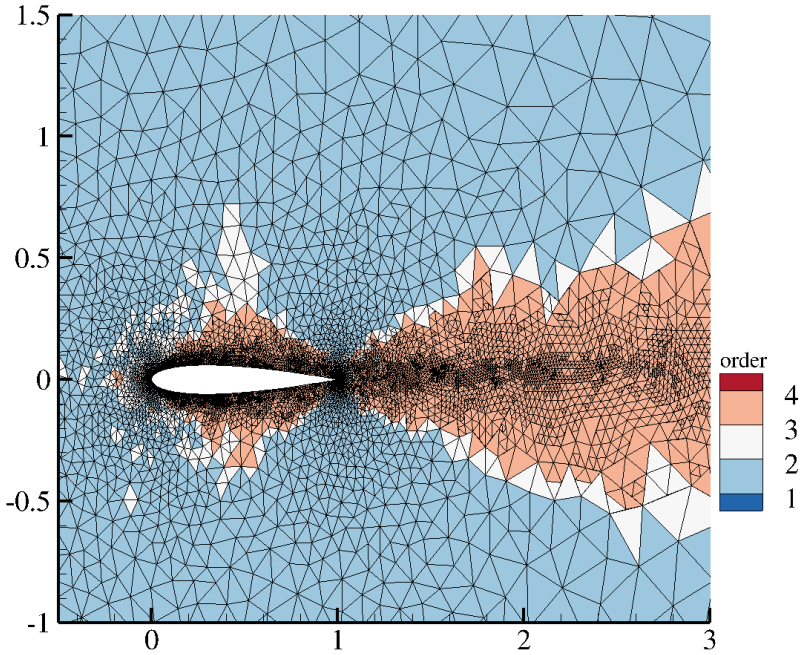


(b) Drag coefficient evolution with respect to CPU time

Figure 5.14: Convergence of drag coefficient for laminar subsonic test case ($Ma_\infty = 0.5$, $Re = 5000$ and $\alpha = 1°$)

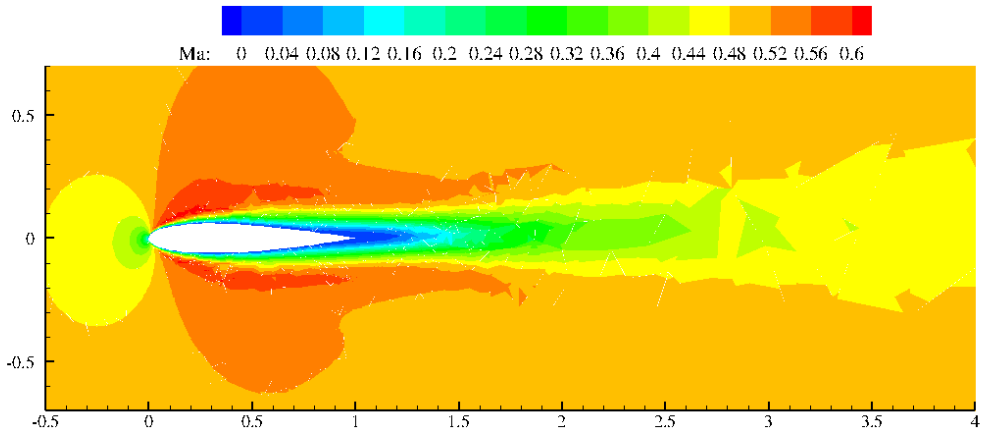


(a) Baseline mesh with 10,797 triangular cells

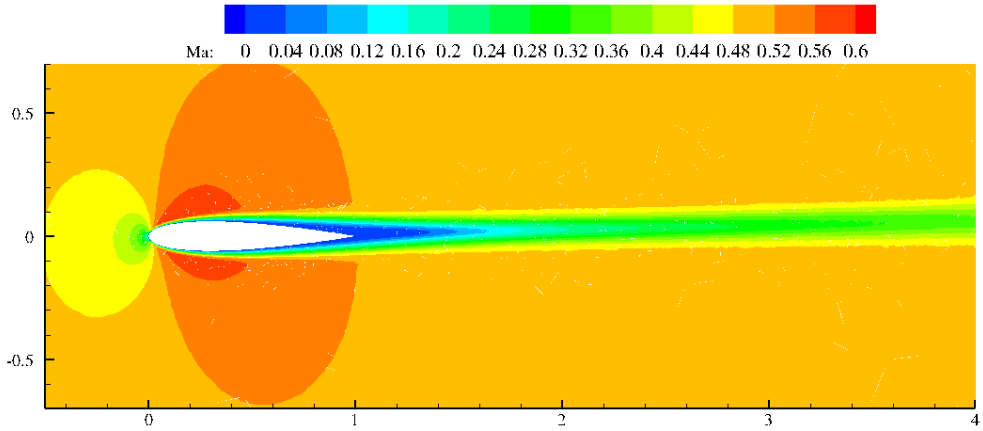


(b) adjoint-based *hp*-adapted after 7 adaptation cycles (second- to fourth-order)

Figure 5.15: Mesh resolution and discretization order for laminar subsonic test case ($Ma_\infty = 0.5$, $Re = 5000$ and $\alpha = 1°$)



(a) Initial mesh with purely second-order discretization



(b) adjoint-based *hp*-adapted final mesh

Figure 5.16: Contours of Mach number for laminar subsonic test case ($Ma_\infty = 0.5$, $Re = 5000$ and $\alpha = 1°$)

near the airfoil on the final mesh are clearly visible.

5.3.4 Turbulent Subsonic Flow

As our final test case, we compute the turbulent subsonic flow around the NACA 0012 airfoil described in Chapter 4. This flow is characterized by a free stream Mach number of $Ma_\infty = 0.15$, Reynolds number of $Re = 6 \times 10^6$ and angle of attack of $\alpha = 10°$. As demonstrated by the previous test cases, the adjoint-based method is superior to the residual-based adaptation. Therefore, we only consider the adjoint-based error indicator for *hp*-adaptation. The lift coefficient is used as the output

functional of interest and is defined as:

$$\mathcal{J}\left(U, \nabla U\right) = \int_{\partial\Omega} \psi \cdot \left(P\hat{n} - \tau\hat{n}\right) \, ds \tag{5.18}$$

with $\psi = \frac{1}{C_\infty} \left(-\sin\,\alpha, \cos\,\alpha\right)^T$along the wall boundaries and 0 everywhere else. The computations are performed on the second family of structured quadrilateral meshes available on the NASA turbulence modeling resource [85]. For simplicity, we assume that the dynamic viscosity is constant.

For uniform refinement, we consider second- to fourth-order discretization on three nested meshes with $14,336$, $57,344$ and $229,376$ control volumes. The adjoint-based *hp*-adaptation starts with the second-order solution on the coarsest mesh (Figure 5.17). As described earlier, those highly anisotropic cells near the wall for which the curvilinear coordinate is employed come out of consideration for further refinement once they reach $p = 3$. The other adaptation parameters are set to $\mathcal{K}_P = 200$, $\mathcal{K}_{\tilde{\nu}} = 20$ and $f^{adapt} = 10\%$.

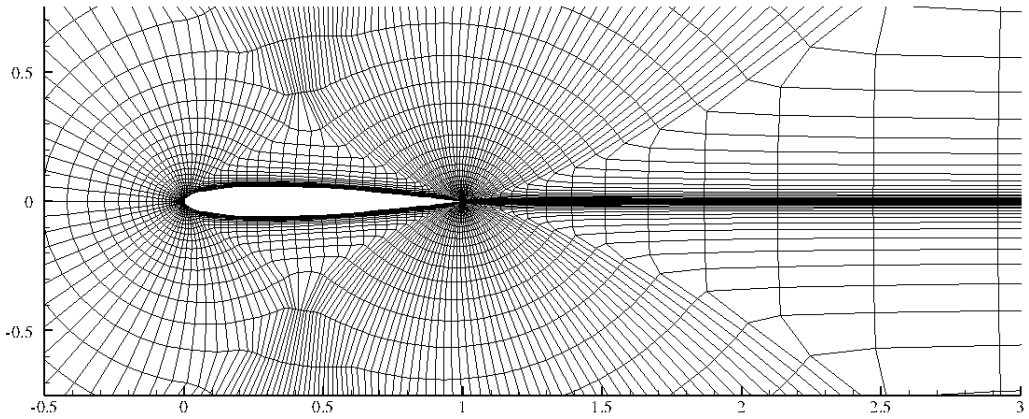


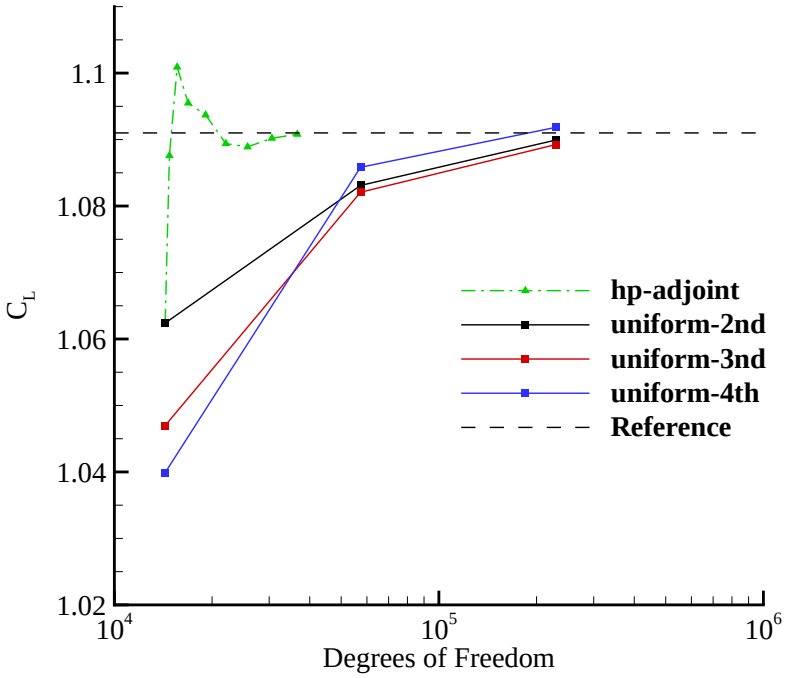
Figure 5.17: The coarsest quadrilateral mesh for turbulent subsonic flow

Figure 5.18 compares the efficiency of the adjoint-based *hp*-adaptation with uniform refinement of different discretization orders. The reference lift coefficient is the value reported by the NASA turbulence modeling resource for these meshes. Again, adaptive refinement is considerably more successful than uniform refinement as it requires 5 times smaller number of control volumes to obtain the grid converged output. This is expected since adaptive refinement places sufficient resolution in the boundary layer and wake regions without unnecessary refinement of other regions.

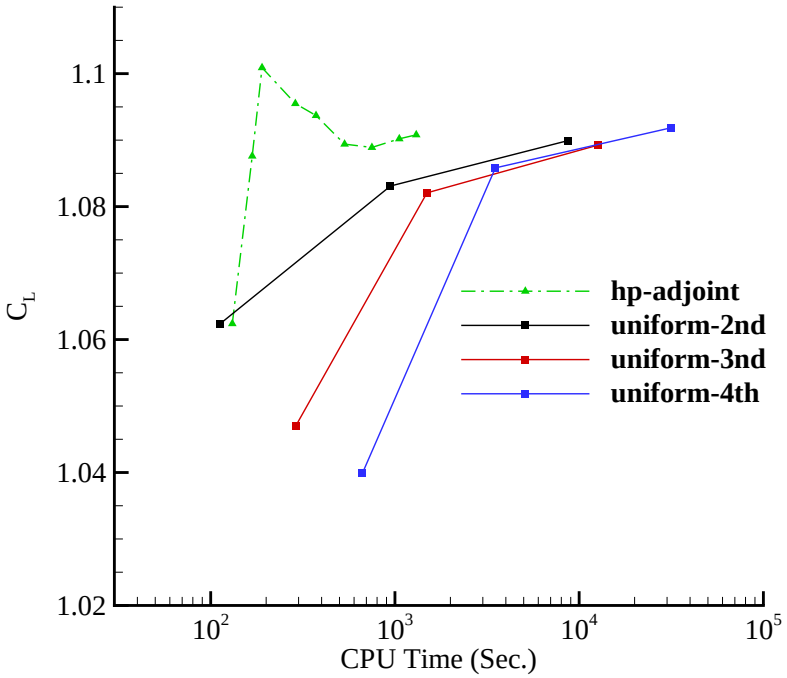
Such a resolution can only be obtained on super fine meshes in the case of uniform refinement. More importantly, the CPU time on the final adapted mesh is 6, 9 and 24 times smaller than that of second-, third- and fourth-order discretizations on the finest mesh, respectively. This is also expected since the extra derivatives required for higher-order approximations are only computed in a small fraction of control volumes and also the Jacobian matrix has fewer entries.

Figure 5.19 shows the final adapted mesh and also order of accuracy for the control volumes close to the airfoil. Note that the adaptive refinement focuses on the boundary layer, the wake behind the trailing edge and the stagnation line at the leading edge. It is also worth mentioning that the order of accuracy at the edge of the boundary layer and wake regions where the derivative of the turbulence working variable is discontinuous does not increase even though the mesh is refined.

Figure 5.20 compares the turbulence working variable between the initial and final *hp*-adapted solutions. The viscous dominated parts of the flow, including the wall boundary layer and wake are captured properly. Moreover, the region at the edge of viscous regions where $\tilde{\nu}$ is negative becomes thinner as we refine. The smooth contours of the turbulence working variable and the peak value in the wake region show the improvement in the solution of the *hp*-adapted mesh compared to the initial mesh.

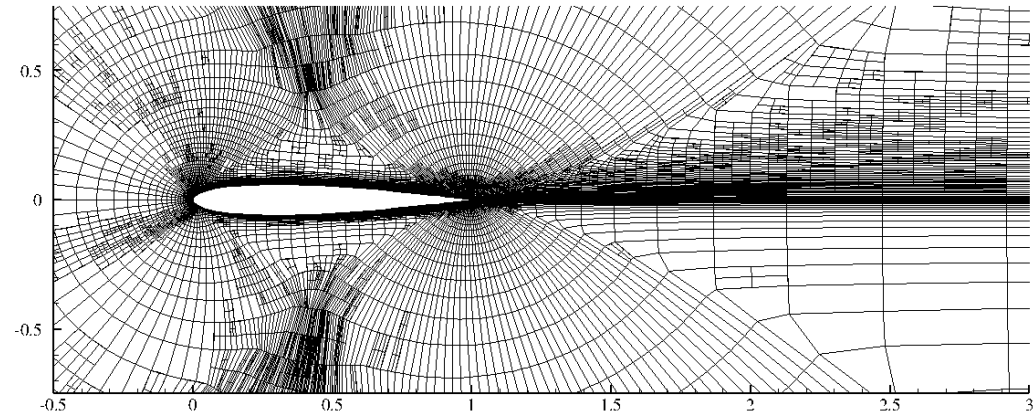


(a) Lift coefficient evolution with respect to degrees of freedom

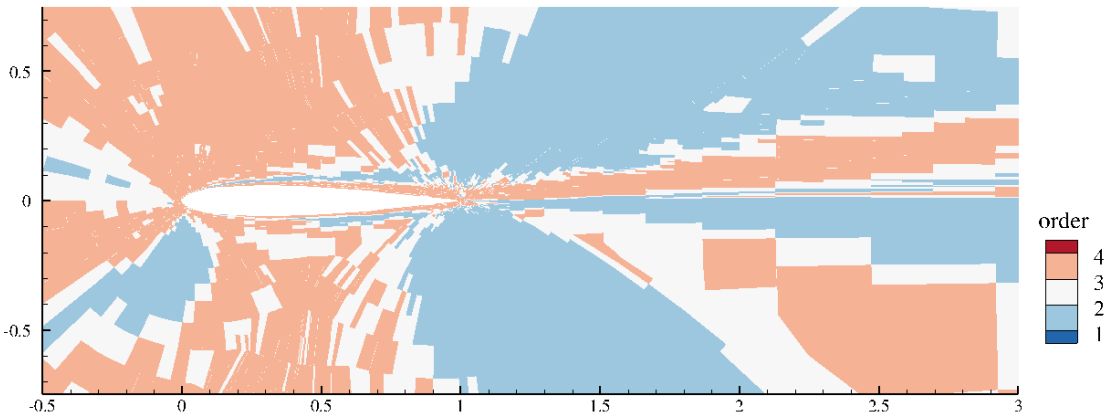


(b) Lift coefficient evolution with respect to CPU time

Figure 5.18: Convergence of lift coefficient for turbulent subsonic test case ($Ma_\infty = 0.15$, $Re = 6 \times 10^6$ and $\alpha = 10°$)

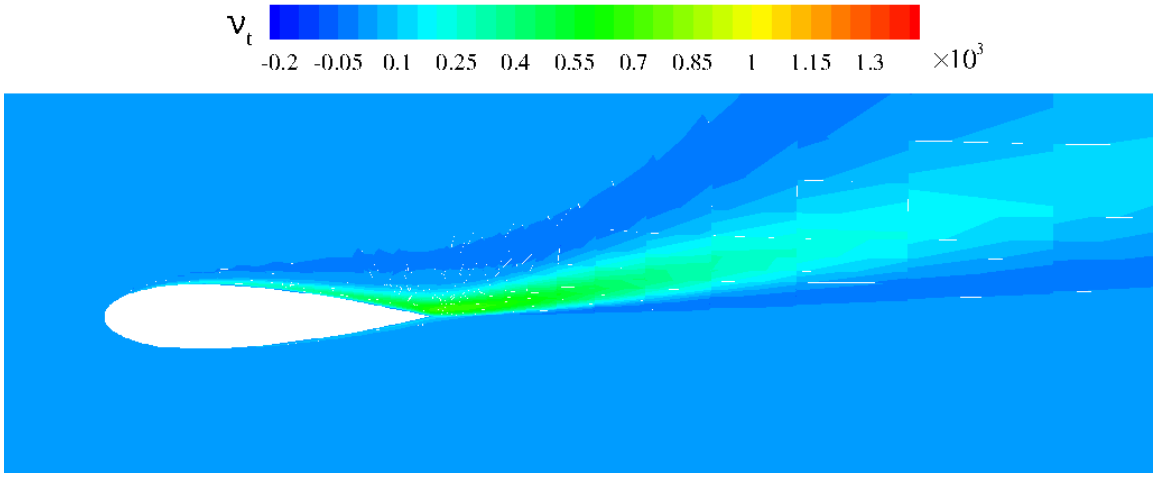


(a) Mesh

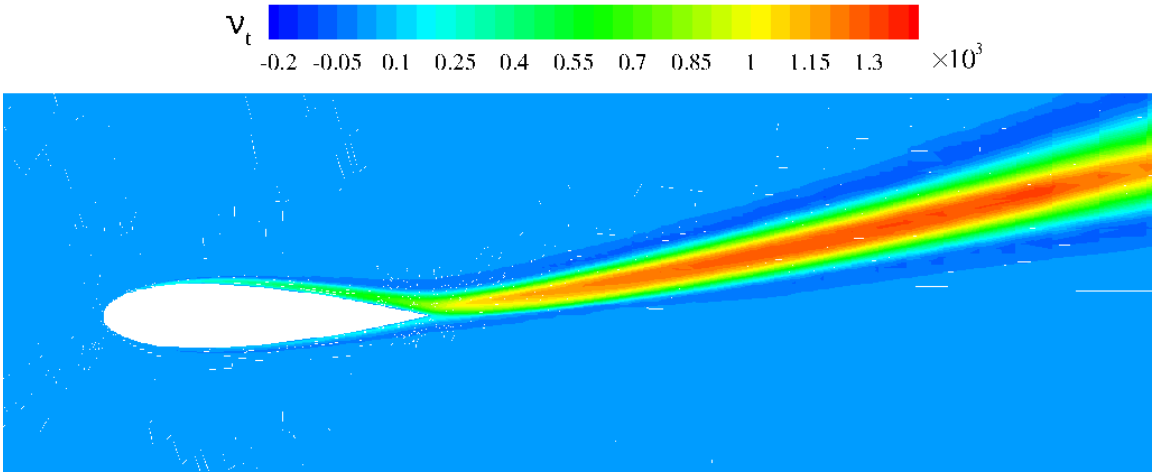


(b) Local order of accuracy

Figure 5.19: Illustration of final *hp*-adapted mesh for turbulent subsonic flow



(a) Initial mesh with purely second-order discretization



(b) adjoint-based *hp*-adapted final mesh

Figure 5.20: Contours of turbulence working variable for turbulent subsonic test case ($Ma_\infty = 0.15$, $Re = 6 \times 10^6$ and $\alpha = 10°$)

Chapter 6

Concluding Remarks

6.1 Summary

This thesis described RANS simulation of turbulent compressible aerodynamic flows using a higher-order unstructured finite volume discretization of governing equations. In addition an *hp*-adaptation framework was developed to improve the efficiency and robustness of the flow solver for the two dimensional problems that are of interest in computational aerodynamics.

The solution of turbulent flows necessitates the use of highly anisotropic meshes since the gradients of flow properties normal to the shear layers are considerably larger than the gradients along them. Considering that all higher-order accurate solvers must account for boundary curvature, we curved the interior faces for highly anisotropic meshes to prevent faces from intersecting near curved boundaries. For this purpose, we used an elasticity analogy to project the boundary curvature into the interior assuming that the domain to be meshed behaves as an elastic solid. The geometrical features of the higher-order mesh were calculated using the coordinate transformation from the curved cells to the original straight cells.

Moreover, the higher-order cell-centered finite volume solution reconstruction procedure was revisited on highly anisotropic meshes over curved surfaces to address two known issues: poor conditioning of the least-squares system and poor accuracy of reconstructed values on anisotropic meshes over curved surfaces. For highly anisotropic meshes, the least-squares system suffers from ill-conditioning and this issue is exacerbated for higher-order computations as the presence of the higher moments of area results in a considerable difference in the order of magnitude of matrix columns. We also investigated the accuracy and conditioning of the k-exact reconstruction on a wide range of unstructured anisotropic meshes from straight triangles to more general cases used for turbulent simulations over aerodynamic configurations. For this purpose, we manufactured anisotropic functions that mimic the characteristics of real

anisotropic solutions in aerodynamic problems.

For turbulent flow simulations, the negative variant of the Spalart-Allmaras turbulence model, developed for higher-order discretizations, was fully coupled to the system of RANS equations. The convective fluxes of the coupled system were obtained by the Roe-Pike flux function and the gradients used in the viscous fluxes were calculated by the average of the two reconstructed gradients plus a jump term having a stabilization property. An implicit time stepping method combined with a line search algorithm was employed to solve the discrete system of equations efficiently. At each non-linear iteration, the GMRES method preconditioned with ILU(k) was used to solve the linear system. We also presented our numerical results for simple and complicated turbulent flow problems in 2D.

Finally, an *hp*-adaptation method was proposed to enhance the capabilities of the solver and improve its efficiency. Considering that the higher-order approximations are only advantageous in smooth regions of the solution, we employed either h- or p-refinement based on the smoothness of the solution. This procedure was carried out for those cells that contribute to the total error the most using residual-based and adjoint-based error indicators. For mesh refinement, we allowed for non-conforming interfaces in the mesh to be able to handle triangles and quadrilaterals in the same. We presented the result of our adaptive solver for different compressible flow problems including inviscid, laminar and turbulent cases.

6.2 Conclusions

To conclude this thesis, a short review of the highlights observed in each step is presented.

For solution reconstruction on anisotropic meshes, our results showed that column scaling using the maximum entry of each column and performing the reconstruction procedure along principal axes alleviates the poor conditioning of the least-squares systems significantly. As a result, the condition number of the scaled matrix in the principal coordinate system can be used as the measure of conditioning.

Furthermore, our reconstruction results demonstrated that the unweighted system produces more accurate reconstruction coefficients in the case of straight meshes where the orientation of principal axes of control volumes change only slightly. Moreover, the condition number of the scaled LS system is small and independent of mesh

size and aspect ratio for the unweighted system. For curved meshes, the reconstructed values using the unweighted LS method in its traditional form do not give the expected order of accuracy. Although adding a weight function to the LS system resolves this issue, it leads to large relative errors for the reconstructed solution and poor estimate of normal derivatives. Also, the condition number of the scaled LS system grows with aspect ratio and mesh refinement in both cases particularly for fourth-order reconstruction. Instead, we proposed a new baseline where a curvilinear coordinate system aligned with the wall is constructed and used for reconstruction. The accuracy results in the proposed coordinates have small values of error and match the expected order of accuracy with or without geometric weights. Nevertheless, the unweighted LS errors are noticeably smaller and thus outperform the weighted LS method. Likewise, the unweighted LS problem is well-conditioned as the condition numbers are small and do not change with mesh size and aspect ratio. Finally, we extended our analysis to general meshes such as those used for turbulent simulations over NACA 0012 airfoils. These meshes are comprised of anisotropic regions near the curved walls and in wake region and also isotropic triangles sufficiently far from these regions. We presented a method for separating the mesh into different regions where different coordinate systems are required for accurate reconstruction. Moreover, we demonstrated the advantage of the new curvilinear coordinate system in approximating anisotropic solutions in the boundary layer region of these meshes.

In terms of turbulent flow simulations, numerical results were presented for several test cases: the turbulent flow over the flat plate, the subsonic flow over the NACA 0012 airfoil, the transonic flow over the RAE 2822 airfoil and the turbulent flow over the three-element airfoil. The solutions of these case were verified by comparison against the solutions of well-established codes or experimental data. For the flat plate and single airfoil test cases, the accuracy advantage of higher-order discretizations was shown by obtaining a more accurate solution on a coarser mesh. It was also shown that solution reconstruction in the newly developed curvilinear coordinates results in more accurate values of drag and lift coefficients for the curved geometries. In addition, the convergence to the steady-state solution from a free stream uniform solution was shown to be fast and efficient for the flat plate and subsonic airfoil test cases for all order of accuracy (second- to fourth-order). However, for the more challenging test case of transonic airfoil in which a shock wave interacts with turbulent boundary layer, starting from a lower-order solution helps the convergence signifi-

cantly. We also used order sequencing to accelerate the convergence for the difficult case of multi-element airfoil. The results of this test case were presented to show the robustness of our solver for complex geometries with several flow structures.

In the previous chapter, the results of the *hp*-adaptation algorithm developed for an unstructured finite volume solver in this thesis were presented for the subsonic inviscid, transonic inviscid, subsonic laminar and subsonic turbulent flows around the NACA 0012 airfoil. In each case, uniform refinement of second- and higher-order methods was compared against residual-based and adjoint-based adaptations in terms of the number of degrees of freedom and CPU time required to reach a reference value. In all of the cases, it was shown that the adjoint-based *hp*-adaptive method outperforms the other refinement procedures and is more successful in capturing the flow features that affect the accuracy of an output functional. Moreover, it was shown that the *hp*-adaptive method is capable of shock capturing and automatic limiting of flows with discontinuities (such as the transonic inviscid case) by mesh refinement and lowering the order of accuracy to one in those regions, simultaneously. For such flows where increasing the order of accuracy in non-smooth regions of the solution does not lead to more accurate solutions or sometimes causes solver failure, such a strategy is superior. Also, the advantage of the adaptive methods were sufficiently clear in the case of viscous laminar and turbulent cases where thin shear layers are present in the solution. For these cases, the adaptive methods place enough resolution in the viscous dominated regions and thus provide a good estimate of functional values as opposed to the uniform refinements where many control volumes are needed to resolve those regions accurately.

6.3 Recommended Future Work

This thesis demonstrated an *hp*-adaptive unstructured finite volume solver for RANS simulations over 2D configurations with aerodynamic applications. There are a number of areas to which this research could be extended.

1. The linear elasticity approach used to curve the interior faces of meshes with highly anisotropic cells was successful for the cases discussed in the thesis. However, for complex configurations with high curvature in several regions, a more powerful curving strategy is required. There are some other analogy approaches

such as those that use non-linear elasticity equations [75] or incorporate thermal stress terms into the state equation [136] to enhance the quality of the higher-order mesh.

2. Although the non-linear convergence of our solver is quite good in most cases, the CPU time spent by the linear system solver does not scale linearly with the number of degrees of freedom. Such behavior could be obtained by using a more powerful preconditioner such as *hp*-multigrid with agglomeration and directional coarsening in the boundary layer region where highly anisotropic meshes exist [137].

3. Higher-order simulation of 2D turbulent flows is challenging as described and requires several modifications. Considering that the ultimate goal in the numerical simulation of flows with engineering applications is the accurate and efficient computation of 3D turbulent flows, this work could be extended to 3D problems later by generalizing the ideas explained in this thesis for 2D problems. However, this requires several considerations:

   - The curvilinear coordinate system developed in this thesis must be extended to 3D. This can be achieved by introducing the unit normal direction, $\hat{n}_i$, in the same way as 2D by finding the unit vector that connects the closest point on the wall to the reference location of the target control volume. For the two other axes (we call them $\hat{t}_i$ and $\hat{b}_i$), we can use the two principal directions of the polygon obtained by the intersection of the cross plane (normal to $\hat{n}_i$) and the target control volume. The first axis, $\hat{t}_i$, is defined as the one that forms the minimum angle with the projection of the free stream velocity vector on the cross plane and the second axis is defined so that $\hat{b}_i = \hat{t}_i \times \hat{n}_i$.
   - For turbulence modeling in 3D, the negative variant of SA model could still be used. However, the current status of computation power and resources allows one uses LES or hybrid RANS/LES for those flows encountered in computational aerodynamics.
   - Also, a very powerful solution strategy particularly for the solution of the linear system is a need. For 3D problems, the size of the linear system is very large and thus better preconditioning techniques are required. Note

that the solution of 3D turbulent problems requires parallel processing with many nodes and thus a scalable linear solver cannot be avoided.

4. In the current setting of our *hp*-adaptive algorithm, second-order discretization is used in those regions that do not affect the functional accuracy. It might be advantageous to add a de-refinement strategy to the adaptive methods to decrease the number of cells in such regions and make the solver even less expensive.

5. Finally, the *hp*-adaptive technique proposed here must be extended to 3D problems since obtaining grid-converged output values is not affordable by uniform refinement in 3D.

Bibliography

[1] C.R. Illingworth. Some solutions of the equations of flow of a viscous compressible fluid. *Proceedings of the Cambridge Philosophical Society*, 46:469–478, 1950.

[2] Doug Pagnutti. Anisotropic adaptation: Metrics and meshes. Master's thesis, The University of British Columbia, Department of Mechanical Engineering, 2008.

[3] F.R. Bailey. High-end computing challenges in aerospace design and engineering. In *Third International Conference on Computational Fluid Dynamics (Invited Lecture)*, 2004.

[4] P.R. Spalart. Topics in detached-eddy simulation. In *Third International Conference on Computational Fluid Dynamics (Invited Lecture)*, 2004.

[5] V. Venkatakrishnan. A perspective on unstructured grid flow solvers. Technical report, ICASE 95-3 NASA, 1995.

[6] S. De Rango and D. W. Zingg. Higher-order spatial discretization for turbulent aerodynamic computations. *American Institute of Aeronautics and Astronautics Journal*, 39(7):1296–1304, July 2001.

[7] Amir Nejat and Carl Ollivier-Gooch. A high-order accurate unstructured finite volume Newton-Krylov algorithm for inviscid compressible flows. *Journal of Computational Physics*, 227(4):2592–2609, 2008.

[8] Stephen B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[9] Parviz Moin and John Kim. Tackling turbulence with supercomputers. *Scientific American*, 276(1):46–52, 1997.

[10] Sergio Pirozzoli and Francesco Grasso. Direct numerical simulation of impinging shock wave/turbulent boundary layer interaction at M = 2.25. *Physics of Fluids*, 18(6):065113, 2006.

[11] Philipp Schlatter and Ramis Örlü. Assessment of direct numerical simulation data of turbulent boundary layers. *Journal of Fluid Mechanics*, 659:116, 2010.

[12] Xuliang Liu, Shuhai Zhang, Hanxin Zhang, and Chi-Wang Shu. A new class of central compact schemes with spectral-like resolution I: Linear schemes. *Journal of Computational Physics*, 248:235–256, 2013.

[13] J.-B. Chapelier, M. de la Llave Plata, F. Renac, and E. Lamballais. Evaluation of a high-order discontinuous Galerkin method for the DNS of turbulent flows. *Computers & Fluids*, 95:210–226, 2014.

[14] Nicholas J. Georgiadis, Donald P. Rizzetta, and Christer Fureby. Large-Eddy Simulation: Current capabilities, recommended practices, and future research. *AIAA Journal*, 48(8):1772–1784, 2010.

[15] Roland Bouffanais. Advances and challenges of applied large-eddy simulation. *Computers & Fluids*, 39:735–738, 2009.

[16] Barrett S. Baldwin and Harvard Lomax. Thin-layer approximation and algebraic model for separated turbulent flows. In *Proceedings of Sixteenth AIAA Aerospace Science Meeting*, 1978. AIAA Paper 78-257.

[17] T. Cebeci. *Analysis of turbulent boundary layers*. Elsevier, 2012.

[18] Barrett S. Baldwin and Timothy J. Barth. A one-equation turbulence transport model for high Reynolds number wall-bounded flows. NASA TM-102847, 1990.

[19] P.R. Spalart and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. In *Proceedings of the Thirtieth AIAA Aerospace Sciences Meeting and Exhibit*, January 1992. AIAA paper 92-0439.

[20] W. P. Jones and B. E. Launder. The prediction of laminarization with a two-equation model of turbulence. *International Journal of Heat and Mass Transfer*, 15(2):301–314, 1972.

[21] B. E. Launder, G. Jr. Reece, and W. Rodi. Progress in the development of a Reynolds-stress turbulence closure. *Journal of Fluid Mechanics*, 68(3):537–566, 1975.

[22] B. Andersson, R. Andersson, L. Håkansson, M. Mortensen, R. Sudiyo, and B. van Wachem. *Computational Fluid Dynamics for Engineers*. Cambridge University Press, 2011.

[23] Pierre Sagaut and Sébastien Deck. Large eddy simulation for aerodynamics: status and perspectives. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 367(1899):2849–2860, 2009.

[24] Z.J. Wang, Krzysztof Fidkowski, Rémi Abgrall, Francesco Bassi, Doru Caraeni, Andrew Cary, Herman Deconinck, Ralf Hartmann, Koen Hillewaert, H.T. Huynh, Norbert Kroll, Georg May, Per-Olof Persson, Bram van Leer, and Miguel Visbal. High-order CFD methods: Current status and perspective. *International Journal for Numerical Methods in Fluids*, 72(8):811–845, 2013.

[25] M. Terracol and S. Deck. Numerical investigation of the flow around a three-element high-lift airfoil using two zonal hybrid RANS/LES methods: ZDES and NLDE. In *Progress in Hybrid RANS-LES Modelling*, pages 345–355. Springer, 2012.

[26] Shia-Hui Peng, Bastian Nebenführ, and Lars Davidson. Lessons learned from hybrid RANS-LES computations of a three-element airfoil flow. In *Proceedings of Twenty-First AIAA Computational Fluid Dynamics Conference*, 2013. AIAA Paper 2013-2841.

[27] Hui Zhu, Song Fu, Lei Shi, and Z. J. Wang. Implicit large-eddy simulation for the high-order flux reconstruction method. *AIAA Journal*, pages 2721–2733, 2016.

[28] S. Lele. Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, 103:16–42, 1992.

[29] M. Visbal and D. Gaitonde. On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1):155–185, 2002.

[30] Xiaogang Deng, Meiliang Mao, Guohua Tu, Hanxin Zhang, and Yifeng Zhang. High-order and high accurate CFD methods and their applications for complex grid problems. *Communications in Computational Physics*, 11(4):1081–1102, 2012.

[31] D. A. Brown and D. W. Zingg. Performance of a Newton-Krylov-Schur algorithm for solving steady turbulent flows. *AIAA Journal*, pages 1–14, 2016.

[32] M. M. Rai. A relaxation approach to patched grid calculations with the Euler equations. *Journal of Computational Physics*, 66:99–131, 1986.

[33] A. Lerat and Z. N. Wu. Stable conservative multidomain treatments for implicit Euler equations. *Journal of Computational Physics*, 123:45–64, 1996.

[34] J. E. Hicken and D. W. Zingg. Parallel Newton-Krylov solver for the Euler equations discretized using simultaneous-approximation terms. *AIA Journal*, 46(11):2773–2786, 2008.

[35] Ami Harten, Stanley Osher, Björn Engquist, and Sukumar R. Chakravarthy. Some results on uniformly high-order accurate essentially non-oscillatory schemes. *Applied Numerical Mathematics*, 2:347–377, 1986.

[36] Ami Harten, Björn Enquist, Stanley Osher, and Sukumar R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71(2):231–303, August 1987.

[37] D. Zingg, S. De Rango, M. Nemec, and T. Pulliam. Comparison of several spatial discretizations for the Navier-Stokes equations. *Journal of Computational Physics*, 160:683–704, 2000.

[38] B. Cockburn, S. Hou, and Shu C. W. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case. *Mathematics of Computation*, 54(190):545–581, 1998.

[39] B. Cockburn and C. W. Shu. The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 35(6):2440–2463, 1998.

[40] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2D Euler equations. *Journal of computational physics*, 138(2):251–285, 1997.

[41] Ralf Hartmann and Paul Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2):508–532, 2002.

[42] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 131:267–279, 1997.

[43] Krzysztof J. Fidkowski, Todd A. Oliver, James Lu, and David L. Darmofal. p-multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations. *Journal of Computational Physics*, 207(1):92 – 113, 2005.

[44] Nicholas K. Burgess and Dimitri J. Mavriplis. An hp-adaptive discontinuous Galerkin solver for aerodynamic flows on mixed-element meshes. In *Proceedings of the Forty-Ninth AIAA Aerospace Sciences Meeting*, 2011.

[45] N. K. Burgess and D. J. Mavriplis. High-order discontinuous galerkin methods for turbulent high-lift flows. In *Proceeding of Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, 2012.

[46] Timothy J. Barth and Paul O. Frederickson. Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction. AIAA paper 90-0013, January 1990.

[47] Amir Nejat. *A Higher-Order Accurate Unstructured Finite Volume Newton-Krylov Algorithm for Inviscid Compressible Flows*. PhD thesis, The University of British Columbia, Department of Mechanical Engineering, 2007.

[48] Christopher Michalak. *Efficient high-order accurate unstructured finite-volume algorithms for viscous and inviscid compressible flows*. PhD thesis, The University of British Columbia, Department of Mechanical Engineering, 2009.

[49] Christopher Michalak and Carl Ollivier-Gooch. Globalized matrix-explicit Newton-GMRES for the high-order accurate solution of the Euler equations. *Computers and Fluids*, 39:1156–1167, 2010.

[50] Timothy J. Barth and Mats G. Larson. A posteriori error estimates for higher order Godunov finite volume methods on unstructured meshes. *Finite Volumes for Complex Applications III, London*, 2002.

[51] Rémi Abgrall. On essentially non-oscillatory schemes on unstructured meshes: Analysis and implementation. *Journal of Computational Physics*, 114(1):45–58, 1994.

[52] Thomas Sonar. On the construction of essentially non-oscillatory finite volume approximations to hyperbolic conservation laws on general triangulations: polynomial recovery, accuracy and stencil selection. *Computer Methods in Applied Mechanics and Engineering*, 140:157–181, 1997.

[53] Timothy J. Barth. Recent developments in high order k-exact reconstruction on unstructured meshes. AIAA paper 93-0668, January 1993.

[54] Carl F. Ollivier-Gooch and Michael Van Altena. A high-order accurate unstructured mesh finite-volume scheme for the advection-diffusion equation. *Journal of Computational Physics*, 181(2):729–752, sep 2002.

[55] Christopher Michalak and Carl Ollivier-Gooch. Accuracy preserving limiter for the high-order accurate solution of the Euler equations. *Journal of Computational Physics*, 228(23):8693–8711, 2009.

[56] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, 1983.

[57] J.L. Steger and R.F. Warming. Flux vector splitting of the inviscid gasdynamics equations with applications to finite difference methods. *Journal of Computational Physics*, 40(2):263–293, 1981.

[58] Bram van Leer. Flux vector splitting for the Euler equations. ICASE Report No. 82-30, NASA Langley Research Center, 1982.

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

[60] Meng-Sing Liou. A sequel to AUSM: AUSM$^+$. *Journal of Computational Physics*, 129(2):364–382, December 1996.

[61] Meng-Sing Liou. A further development of the AUSM+ scheme towards robust and accurate solutions for all speeds. In *Proceedings of the Sixteenth AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, June 2003. AIAA paper 2003-4116.

[62] P. L. Roe. Characteristic-based schemes for the Euler equations. In *Annual Review of Fluid Mechanics*, volume 18, pages 337–365. Annuals Reviews, Inc., 1986.

[63] Charles Hirsch. *Numerical computation of internal and external flows: Computational methods for inviscid and viscous flows*. Wiley, 1990.

[64] Andrew W. Cary, Andrew J. Dorgan, and Mori Mani. Towards accurate flow predictions using unstructured meshes. In *Proceedings of the Nineteenth AIAA Computational Fluid Dynamics Conference*, 2009. AIAA paper 2009-3650.

[65] A. H. Stroud and Don Secrest. *Gaussian Quadrature Formulas*. Prentice-Hall, Englewood Cliffs, N.J., 1966.

[66] Carl Ollivier-Gooch, Amir Nejat, and Christopher Michalak. On obtaining and verifying high-order finite-volume solutions to the Euler equations on unstructured meshes. *American Institute of Aeronautics and Astronautics Journal*, 47(9):2105–2120, 2009.

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

[68] D. A. Knoll, P. R. McHugh, and D. E. Keyes. Newton-Krylov methods for low-Mach-number compressible combustion. *AIAA Journal*, 34(5):961–967, 1996.

[69] D. A. Knoll and D. E. Keyes. Jacobian-free Newton-Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, 2004.

[70] Amir Nejat and Carl Ollivier-Gooch. Effect of discretization order on preconditioning and convergence of a high-order unstructured Newton-GMRES solver for the Euler equations. *Journal of Computational Physics*, 227(4):2366–2386, 2008.

[71] David C. Wilcox. Comparison of two-equation turbulence models for boundary layers with pressure gradient. *AIA Journal*, 31(8):1414–1421, 1993.

[72] David C. Wilcox. Simulation of transition with a two-equation turbulence model. *AIAA Journal*, 32(2):247–255, 1994.

[73] Xiaojuan Luo. *An Automatic Adaptive Directional Variable p-version Method in 3D Curved Domains*. PhD thesis, Rensselaer Polytechnic Institute, 2005.

[74] Spencer J. Sherwin and Joaquim Peiro. Mesh generation in curvilinear domains using high-order elements. *International Journal for Numerical Methods in Engineering*, 52(2-3):207–223, 2002.

[75] Per Olaf Persson and Jaime Peraire. Curved mesh generation and mesh refinement using Lagrangian solid mechanics. In *Proceedings of the Forty-Seventh AIAA Aerospace Sciences Meeting*, 2009. AIAA 2009-949.

[76] Li Wang, W. Kyle Anderson, J. Taylor Erwin, and Sagar Kapadia. Solutions of high-order methods for three-dimensional compressible viscous flows. In *Proceedings of the Twentieth AIAA Computational Fluid Dynamics Conference*, 2012.

[77] Dimitri J. Mavriplis. Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes. In *Proceedings of the Sixteenth AIAA Computational Fluid Dynamics Conference*, 2003.

[78] Boris Diskin, James L. Thomas, Eric J. Nielsen, Hiroaki Nishikawa, and Jeffrey A. White. Comparison of node-centered and cell-centered unstructured finite-volume discretizations: Viscous fluxes. *American Institute of Aeronautics and Astronautics Journal*, 48(7):1326–1338, July 2010.

[79] Natalia B. Petrovskaya. Discontinuous weighted least-squares approximation on irregular grids. *Computer Modeling in Engineering and Sciences*, 32:69–84, 2008.

[80] Natalia B. Petrovskaya. A quadratic least-squares solution reconstruction in a boundary layer region. *International Journal for Numerical Methods in Biomedical Engineering*, 26:1721–1735, 2010.

[81] Natalia B. Petrovskaya. Data dependent weights in discontinuous weighted least-squares approximation with anisotropic support. *Calcolo*, 48:127–143, 2011.

[82] Alireza Jalali and Carl F. Ollivier-Gooch. Higher-order finite volume solution reconstruction on highly anisotropic meshes. In *Proceedings of the Twenty-First AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics, 2013.

[83] Lucian Ivan and Clinton Groth. High-order solution-adaptive central essentially non-oscillatory (CENO) method for viscous flows. *Journal of Computational Physics*, 257:830–862, 2014.

[84] H. Reichardt. Vollständige darstellung der turbulenten geschwindigkeitsverteilung in glatten leitungen. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 31:208–219, 1951.

[85] Christopher Rumsey. Turbulence Modeling Resource, NASA Langley Research Center, http://turbmodels.larc.nasa.gov/, 2014.

[86] Francesco Bassi, Andrea Crivellini, Stefano Rebay, and Marco Savini. Discontinuous Galerkin solution of the Reynolds-averaged Navier-Stokes and k-ω turbulence model equations. *Computers and Fluids*, 34(4-5):507 – 540, 2005. Residual Distribution Schemes, Discontinuous Galerkin Schemes and Adaptation.

[87] Ngoc Nguyen, Per-Olof Persson, and Jaime Peraire. RANS solutions using high order discontinuous Galerkin methods. In *Proceedings of Forty-fifth AIAA Aerospace Sciences Meeting and Exhibit*, 2007. AIAA Paper 2007-914.

[88] Todd A. Oliver. *A High-Order, Adaptive, Discontinuous Galerkin Finite Element Method for the Reynolds-Averaged Navier-Stokes Equations*. PhD thesis, Massachusetts Institute of Technology, 2008.

[89] T. Oliver and D. Darmofal. An unsteady adaptation algorithm for discontinuous Galerkin discretizations of the RANS equations. In *Proceedings of the Eighteenth AIAA CFD Conference*, 2007. AIAA Paper 2007-3940.

[90] M. Ceze and K. Fidkowski. Constrained pseudo-transient continuation. *International Journal for Numerical Methods in Engineering*, 102:1683–1703, 2015.

[91] R. Hartmann. Higher-order and adaptive discontinuous Galerkin methods with shock-capturing applied to transonic turbulent delta wing flow. *International Journal for Numerical Methods in Fluids*, 72:883–894, 2013.

[92] Li Wang, W. Kyle Anderson, J. Taylor Erwin, and Sagar Kapadia. Discontinuous Galerkin and Petrov Galerkin methods for compressible viscous flows. *Computers & Fluids*, 100:13–29, 2014.

[93] D. De Santis. High-order linear and non-linear residual distribution schemes for turbulent compressible flows. *Computer Methods in Applied Mechanics and Engineering*, 285:1–31, 2015.

[94] D. Moro, N. C. Nguyen, and J. Peraire. Navier-Stokes solution using hybridizable discontinuous Galerkin methods. In *Proceedings of the Twentieth AIAA Computational fluid Dynamics Conference*, 2011. AiAA Paper 2011-3407.

[95] S. R. Allmaras, F. T. Johnson, and P. Spalart. Modifications and clarifications for the implementation of the Spalart-Allmaras turbulence model. In *Proceedings of the Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, 2012.

[96] M. Ceze and K. Fidkowski. Pseudo-transient continuation, solution update methods, and CFL strategies for DG discretizations of the RANS-SA equations. In *Proceedings of the Twenty-First AIAA Fluid Dynamics Conference*, 2013. AIAA paper 2013-2686.

[97] F. Eleuterio Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer-Verlag, New York, NY, 1999.

[98] Boris Diskin, James L. Thomas, Eric J. Nielsen, Hiroaki Nishikawa, and Jeffrey A. White. Comparison of node-centered and cell-centered unstructured finite-volume discretizations. Part I: Viscous fluxes. In *Proceedings of the Forty-Seventh AIAA Aerospace Sciences Meeting*, 2009. AIAA paper 2009-597.

[99] Hiroaki Nishikawa. Beyond interface gradient: A general principle for constructing diffusion scheme. In *Proceedings of the Fortieth AIAA Fluid Dynamics Conference and Exhibit*, 2010. AIAA paper 2010-5093.

[100] Alireza Jalali, Mahkame Sharbatdar, and Carl Ollivier-Gooch. Accuracy analysis of unstructured finite volume discretization schemes for diffusive fluxes. *Computers and Fluids*, 101:220–232, 2014.

[101] R.C. Swanson and C.-C. Rossow. An efficient solver for the RANS equations and a one-equation turbulence model. *Computers & Fluids*, 42:13–25, 2011.

[102] Alireza Jalali and Carl Ollivier-Gooch. Higher-order unstructured finite volume RANS solution of turbulent compressible flows. *Computer & Fluids*, 143:32–47, 2017.

[103] P.H. Cook, M.A. McDonald, and M.C.P. Firmin. Aerofoil RAE 2822 - pressure distributions, and boundary layer and wake measurements. In *Experimental Data Base for Computer Program Assessment, AGARD Report AR 138*. AGARD, 1979.

[104] First International Workshop on High-Order CFD Methods. http://people.ku.edu/ z651w035/hiocfd.html, 2012. Sponsored by the AIAA Fluid Dynamics Technical Committee, the US Air Force Office of Scientific Research, and the Deutsches Zentrum fur Luft- und Raumfahrt.

[105] Third International Workshop on High-Order CFD Methods. https://www.grc.nasa.gov/hiocfd/. Sponsored by NASA, AIAA, DLR, and Army Research Office (ARO).

[106] Alireza Jalali and Carl Ollivier-Gooch. Higher-order unstructured finite volume methods for turbulent aerodynamic flows. In *Proceedings of the Twenty-Second AIAA Computational Fluid Dynamics Conference*, number AIAA Paper 2015-2284, 2015.

[107] Stefano Giani and Paul Houston. High–order hp–adaptive discontinuous Galerkin finite element methods for compressible fluid flows. In *ADIGMA-A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*, pages 399–411. Springer, 2010.

[108] Marco Ceze and Krzysztof J. Fidkowski. Anisotropic hp-adaptation framework for functional prediction. *AIAA Journal*, 51(2):492–509, 2013.

[109] Li Wang and Dimitri J Mavriplis. Adjoint-based hp adaptive discontinuous Galerkin methods for the 2d compressible Euler equations. *Journal of Computational Physics*, 228(20):7643–7661, 2009.

[110] Nicholas K. Burgess and Dimitri J. Mavriplis. hp-adaptive discontinuous Galerkin methods for the Navier-Stokes equations. *AIAA Journal*, 50(12):2682–2694, 2012.

[111] Michael Woopen, Arvind Balan, Georg May, and Jochen Schütz. A comparison of hybridized and standard DG methods for target-based hp-adaptive simulation of compressible flow. *Computers & Fluids*, 98:3–16, 2014.

[112] W. Kyle Anderson, Behzad R. Ahrabi, and James C. Newman. Finite element solutions for turbulent flow over the NACA 0012 airfoil. *AIA Journal*, 54(9):2688–2704, 2016.

[113] Behzad R. Ahrabi and W. Kyle Anderson. An adjoint-based hp-adaptive Petrov–Galerkin method for turbulent flows. In *Proceedings of Twenty-Second AIAA Computational Fluid Dynamics Conference*, 2015. AIAA Paper 2015-2603.

[114] Lei Shi and Z. J. Wang. Adjoint-based error estimation and mesh adaptation for the correction procedure via reconstruction method. *Journal of Computational Physics*, 295:261–284, 2015.

[115] Michael J. Aftosmis, Datta Gaitonde, and T. Sean Tavares. On the accuracy, stability, and monotonicity of various reconstruction algorithms for unstructured meshes. AIAA paper 94-0415, January 1994.

[116] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82(1):64–84, 1989.

[117] Mahkame Sharbatdar and Carl Ollivier-Gooch. Anisotropic mesh adaptation: Recovering quasi-structured meshes. In *Proceedings of the Fifty-First AIAA Aerospace Sciences Meeting*, 2013.

[118] Doug Pagnutti and Carl Ollivier-Gooch. A generalized framework for high order anisotropic mesh adaptation. *Computers and Structures*, 87:670–679, 2009.

[119] X. Zhang, M.-G. Vallet, J. Dompierre, P. Labbe, D. Pelletier, J.-Y. Trepanier, R. Camarero, J. Lassaline, L. Manzano, and D. Zingg. Mesh adaptation using different error indicators for the Euler equations. In *Proceedings of the Fifteenth AIAA Computational Fluid Dynamics Conference*, 2001. AIAA Paper 2001-2549.

[120] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional viscous flows. *Journal of Computational Physics*, 187:22–46, 2003.

[121] M. Ainsworth and J. T. Oden. A unified approach to a posteriori error estimation using element residual methods. *Numerische Mathematik*, 65(1):23–50, 1993.

[122] Haiyang Gao and Z.J. Wang. A residual-based procedure for hp-adaptation on 2D hybrid meshes. In *Proceedings of the Forty-ninth AIAA Aerospace Sciences Meeting*, 2011. AIAA Paper 2011-492.

[123] N. Ganesh, N. V. Shende, and N. Balakrishnan. A residual estimator based adaptation strategy for compressible flows. In *Computational Fluid Dynamics 2006*, pages 383–388. Springer, 2009.

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

[125] Niles Pierce and Michael Giles. Adjoint and defect error bounding and correction for functional estimates. *Journal of Computational Physics*, 200(2):769–794, 2004.

[126] Michael Andrew Park. Adjoint-based, three-dimensional error prediction and grid adaptation. *AIAA Journal*, 42(9):1854–1862, 2004.

[127] D. A. Venditti and D. L. Darmofal. Adjoint error estimation and grid adaptation for functional outputs: Application to quasi-one-dimensional flow. *Journal of Computational Physics*, 164(1):204–227, October 2000.

[128] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional inviscid flows. *Journal of Computational Physics*, 175(1):40–69, February 2002.

[129] Ralf Hartmann. Adaptive discontinuous Galerkin methods with shock-capturing for the compressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 51(9–10):1131–1156, 2006.

[130] R. Hartmann., J. Held, and T. Leicht. Adjoint-based error estimation and adaptive mesh refinement for the RANS and k–ω turbulence model equations. *Journal of Computational Physics*, 230(11):4268–4284, 2011.

[131] Mahkame Sharbatdar and Carl Ollivier-Gooch. Comparison of truncation error estimators for defect correction and output error estimation on unstructured mesh finite volume method. In *Proceedings of the Fifty-Third AIAA Aerospace Sciences Meeting*, 2015.

[132] Mahkame Sharbatdar, Alireza Jalali, and Carl Ollivier-Gooch. Smoothed truncation error in functional error estimation and correction using adjoint methods in an unstructured finite volume solver. *Computers & Fluids*, 140:406–421, 2016.

[133] L. Krivodonova, J. Xin, J.-F. Remacle, N. Chevaugeon, and J.E. Flaherty. Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. *Applied Numerical Mathematics*, 48:323–338, 2004.

[134] V. Venkatakrishnan. Convergence to steady-state solutions of the Euler equations on unstructured grids with limiters. *Journal of Computational Physics*, 118:120–130, 1995.

[135] M. Woopen, G. May, and J. Schütz. Adjoint-based error estimation and mesh adaptation for hybridized discontinuous Galerkin methods. *International Journal for Numerical Methods in Fluids*, 76(11):811–834, 2014.

[136] D. Moxey, D. Ekelschot, U. Keskin, S. J. Sherwin, and J. Peiro. A thermo-elastic analogy for high-order curvilinear meshing with control of mesh validity and quality. *Procedia Engineering*, 82:127–135, 2014.

[137] Nicholas K. Burgess. *An Adaptive Discontinuous Galerkin Solver for Aerodynamic Flows*. PhD thesis, University of Wyoming, December 2011.

[138] Roland W. Lewis, Perumal Nithiarasu, and Kankanhalli N. Seetharamu. *Fundamentals of the Finite Element Method for Heat and Fluid Flow*. Wiley, 2004.

[139] Eugenio Oñate. *Structural Analysis with the Finite Element Method. Linear Statics*. Springer, 2013.

Appendix A: Cubic Interpolation Functions

As discussed in Chapter 3, cubic interpolation functions on 1D/2D reference elements are required for flux integration over curved faces, curving the interior faces of an anisotropic mesh and source term integration over cubic cells. Here we present the interpolation functions for each type.

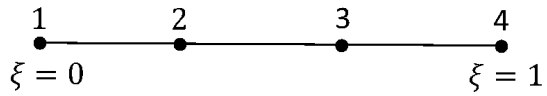


Figure A.1: Reference line segment for face integration

For a reference line segment (Figure A.1), interpolation functions are:

$$\phi_1(\xi) = \frac{1}{2}(1-\xi)(1-3\xi)(2-3\xi)$$

$$\phi_2(\xi) = \frac{9}{2}(1-\xi)\xi(2-3\xi)$$

$$\phi_3(\xi) = \frac{9}{2}(1-\xi)\xi(3\xi-1)$$

$$\phi_4(\xi) = \frac{1}{2}\xi(3\xi-1)(3\xi-2)$$

For a reference triangular element (Figure A.2), we use cubic Lagrangian interpolation functions. To simplify the formulation of these functions, three dependent variables are defined as [138]:

$$L_1 = 1-\xi-\eta\,,\ L_2 = \xi\,,\ L_3 = \eta$$

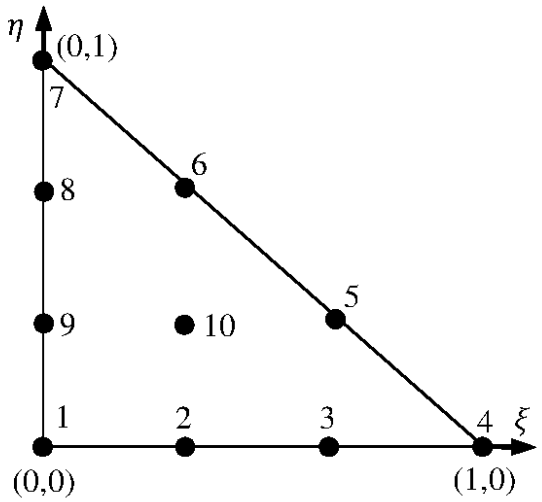


Figure A.2: Reference cubic triangular element

The interpolation functions are given as:

$$\phi_1(\xi,\eta) = \frac{1}{2}L_1(3L_1-1)(3L_1-2)$$

$$\phi_2(\xi,\eta) = \frac{9}{2}L_1L_2(3L_1-1)$$

$$\phi_3(\xi,\eta) = \frac{9}{2}L_1L_2(3L_2-1)$$

$$\phi_4(\xi,\eta) = \frac{1}{2}L_2(3L_2-1)(3L_2-2)$$

$$\phi_5(\xi,\eta) = \frac{9}{2}L_2L_3(3L_2-1)$$

$$\phi_6(\xi,\eta) = \frac{9}{2}L_2L_3(3L_3-1)$$

$$\phi_{7(\xi,\eta)} = \frac{1}{2}L_3(3L_3-1)(3L_3-2)$$

$$\phi_8(\xi,\eta) = \frac{9}{2}L_3L_1(3L_3-1)$$

$$\phi_9(\xi,\eta) = \frac{9}{2}L_3L_1(3L_1-1)$$

$$\phi_{10}(\xi,\eta) = 27L_1L_2L_3$$

For a reference quadrilateral element (Figure A.3), we use cubic serendipity in-

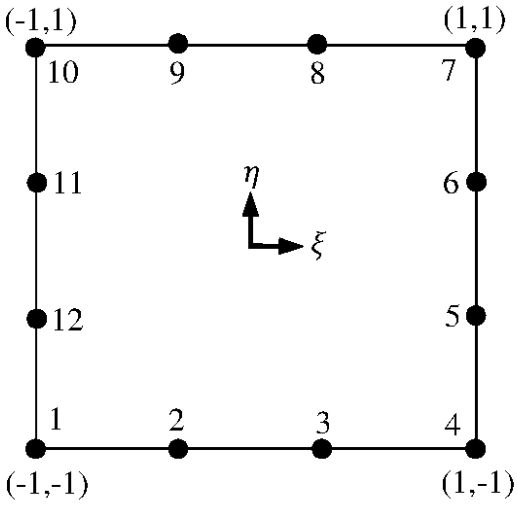


Figure A.3: Reference cubic quadrilateral element

terpolation functions. These functions are given as [139]:

$$\phi_1(\xi,\eta) = \frac{1}{32}(1-\xi)(1-\eta)\left(-10+9\xi^2+9\eta^2\right)$$

$$\phi_2(\xi,\eta) = \frac{9}{32}(1-3\xi)\left(1-\xi^2\right)(1-\eta)$$

$$\phi_3(\xi,\eta) = \frac{9}{32}(1+3\xi)\left(1-\xi^2\right)(1-\eta)$$

$$\phi_4(\xi,\eta) = \frac{1}{32}(1+\xi)(1-\eta)\left(-10+9\xi^2+9\eta^2\right)$$

$$\phi_5(\xi,\eta) = \frac{9}{32}(1+\xi)\left(1-\eta^2\right)(1-3\eta)$$

$$\phi_6(\xi,\eta) = \frac{9}{32}(1+\xi)\left(1-\eta^2\right)(1+3\eta)$$

$$\phi_7(\xi,\eta) = \frac{1}{32}(1+\xi)(1+\eta)\left(-10+9\xi^2+9\eta^2\right)$$

$$\phi_8(\xi,\eta) = \frac{9}{32}(1+3\xi)\left(1-\xi^2\right)(1+\eta)$$

$$\phi_9(\xi,\eta) = \frac{9}{32}(1-3\xi)\left(1-\xi^2\right)(1+\eta)$$

$$\phi_{10}(\xi,\eta) = \frac{1}{32}(1-\xi)(1+\eta)\left(-10+9\xi^2+9\eta^2\right)$$

$$\phi_{11}(\xi,\eta) = \frac{9}{32}(1-\xi)\left(1-\eta^2\right)(1+3\eta)$$

$$\phi_{12}(\xi,\eta) = \frac{9}{32}(1-\xi)\left(1-\eta^2\right)(1-3\eta)$$

Appendix B: Non-dimensional Equations

In most CFD solvers, a non-dimensional set of equations are discretized to keep the magnitude of flow quantities close to each other. In this Appendix, we describe the non-dimensional parameters and groups used in our solver.

To non-dimensionalize the system of RANS equations given in Chapter 4, we need to define reference variables to scale the flow field quantities. For this purpose, we set the speed of sound, density, temperature and viscosity at the free stream as the reference variables:

$$\rho_{ref} = \rho_\infty,\ u_{ref} = a_\infty,\ v_{ref} = a_\infty,\ T_{ref} = T_\infty,\ \mu_{\text{ref}} = \mu_\infty \tag{B.1}$$

The reference variables for the other flow quantities can be obtained using the variables of Equation B.1 and a characteristic length, L, as:

$$P_{ref} = \rho_\infty a_\infty^2,\ \tilde{\nu}_{ref} = \frac{\mu_\infty}{\rho_\infty},\ \mu_{T,ref} = \mu_\infty,\ t_{ref} = \frac{L}{a_\infty},\ x_{ref} = L,\ y_{ref} = L \tag{B.2}$$

We also define our non-dimensional groups as:

$$Ma_\infty = \frac{V_\infty}{a_\infty},\ \ Re = \frac{\rho V_\infty L}{\mu_\infty} \tag{B.3}$$

A non-dimensional variable for a quantity of interest is defined as $X^* = X/X_{ref}$. However, we choose to show the non-dimensional variables without an asterisk to simplify our notation. Using the reference values and non-dimensional groups, we can re-write the conservation of mass, momentum and energy in the non-dimensional form as:

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} = 0 \tag{B.4}$$

$$\frac{\partial(\rho u)}{\partial t}+\frac{\partial(\rho u^2)}{\partial x}+\frac{\partial(\rho uv)}{\partial y}=-\frac{\partial P}{\partial x}+\frac{Ma_\infty}{Re}\left(\frac{\partial \tau_{xx}}{\partial x}+\frac{\partial \tau_{yx}}{\partial y}\right) \tag{B.5}$$

$$\frac{\partial(\rho v)}{\partial t}+\frac{\partial(\rho uv)}{\partial x}+\frac{\partial(\rho v^2)}{\partial y}=-\frac{\partial P}{\partial y}+\frac{Ma_\infty}{Re}\left(\frac{\partial \tau_{xy}}{\partial x}+\frac{\partial \tau_{yy}}{\partial y}\right) \tag{B.6}$$

$$\begin{aligned}\frac{\partial E_t}{\partial t}+\frac{\partial(uE_t)}{\partial x}+\frac{\partial(vE_t)}{\partial y} &= -\left(\frac{\partial(uP)}{\partial x}+\frac{\partial(vP)}{\partial y}\right)\\ &+ \frac{Ma_\infty}{Re}\left(\frac{\partial(u\tau_{xx}+v\tau_{xy})}{\partial x}+\frac{\partial(y\tau_{yx}+v\tau_{yy})}{\partial y}\right.\\ &+ \left.\frac{\gamma}{\gamma-1}\left(\frac{\mu}{Pr}+\frac{\mu_T}{Pr_T}\right)\left(\frac{\partial^2 T}{\partial \bar{x}^2}+\frac{\partial^2 T}{\partial y^2}\right)\right)\end{aligned} \tag{B.7}$$

Similarly, the governing equation of the SA-negative turbulence model must be non-dimensionalized using the same reference variables:

$$\begin{aligned}\frac{\partial(\rho\tilde{\nu})}{\partial t}+\frac{\partial(\rho u\tilde{\nu})}{\partial x}+\frac{\partial(\rho v\tilde{\nu})}{\partial y} &= \frac{Ma_\infty}{Re\cdot\sigma}\left(\frac{\partial}{\partial x}\left(\left(\mu+\mu' f_n\rho\tilde{\nu}\right)\frac{\partial\tilde{\nu}}{\partial x}\right)\right.\\ &+ \frac{\partial}{\partial y}\left(\left(\mu+\mu' f_n\rho\tilde{\nu}\right)\frac{\partial\tilde{\nu}}{\partial y}\right)\\ &+ c_{b2}\,\mu'\,\rho\left(\left(\frac{\partial\tilde{\nu}}{\partial x}\right)^2+\left(\frac{\partial\tilde{\nu}}{\partial y}\right)^2\right)\\ &- \left.\left(\frac{\mu}{\rho}+\mu'\tilde{\nu}\right)\left(\frac{\partial\rho}{\partial x}\frac{\partial\tilde{\nu}}{\partial x}+\frac{\partial\rho}{\partial y}\frac{\partial\tilde{\nu}}{\partial y}\right)\right)\\ &+ \rho\,(P-D)\end{aligned} \tag{B.8}$$

The non-dimensional production and destruction terms are defined as:

$$P=\begin{cases}c_{b1}\,(1-f_{t2})\,\tilde{S}\tilde{\nu} & \tilde{\nu}\geq 0\\ c_{b1}\,(1-c_{t3})\,S\tilde{\nu} & \tilde{\nu}<0\end{cases},\quad D=\begin{cases}\frac{Ma_\infty}{Re}\mu'\left(c_{w1}f_w-\frac{c_{b1}}{\kappa^2}f_{t2}\right)\left(\frac{\tilde{\nu}}{d}\right)^2 & \tilde{\nu}\geq 0\\ -\frac{Ma_\infty}{Re}\mu' c_{w1}\left(\frac{\tilde{\nu}}{d}\right)^2 & \tilde{\nu}<0\end{cases} \tag{B.9}$$

where

$$\tilde{S}=\begin{cases}S+\bar{S} & \bar{S}\geq -c_{v2}S\\ S+\frac{S\left(c_{v2}^2S+c_{v3}\bar{S}\right)}{(c_{v3}-2c_{v2})S-\bar{S}} & \bar{S}\leq -c_{v2}S\end{cases} \tag{B.10}$$

$$\begin{aligned} S &= \left|\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right| \\ \bar{S} &= \frac{Ma_\infty}{Re}\frac{\mu'\tilde{\nu} f_{v2}}{\kappa^2 d^2} \end{aligned} \tag{B.11}$$

and the wall function contributing to the destruction term is obtained as:

$$\begin{aligned} r &= \min\left(\frac{Ma_\infty}{Re}\frac{\mu'\tilde{\nu}}{\tilde{S}\kappa^2 d^2}, 10\right) \\ g &= r + c_{w2}\left(r^6 - r\right) \\ f_w &= g\left[\frac{1+c_{w3}^6}{g^6 + c_{w3}^6}\right] \end{aligned} \tag{B.12}$$

The other constants and functions are the same as the dimensional form of the SA-negative model.