DEVELOPMENT OF 2D TURBULENT NAVIER-STOKES SOLVER FOR CARTESIAN GRIDS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ONUR ATA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

SEPTEMBER 2022

Approval of the thesis:

DEVELOPMENT OF 2D TURBULENT NAVIER-STOKES SOLVER FOR CARTESIAN GRIDS

submitted by **ONUR ATA** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Mehmet Ali Sahir Arıkan
Head of Department, **Mechanical Engineering** _____

Prof. Dr. Mehmet Haluk Aksel
Supervisor, **Mechanical Engineering, METU** _____

Assist. Prof. Dr. Özgür Uğraş Baran
Co-supervisor, **Mechanical Engineering, METU** _____

Examining Committee Members:

Assist. Prof. Dr. Ali Karakuş
Mechanical Engineering, METU _____

Prof. Dr. Mehmet Haluk Aksel
Mechanical Engineering, METU _____

Assist. Prof. Dr. Özgür Uğraş Baran
Mechanical Engineering, METU _____

Assist. Prof. Dr. Hediye Atik
Aerospace Engineering, Atilim University _____

Assist. Prof. Dr. Onur Baş
Mechanical Engineering, TEDU _____

Date:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Onur Ata

Signature :

ABSTRACT

DEVELOPMENT OF 2D TURBULENT NAVIER-STOKES SOLVER FOR CARTESIAN GRIDS

Ata, Onur
M.S., Department of Mechanical Engineering
Supervisor: Prof. Dr. Mehmet Haluk Aksel
Co-Supervisor: Assist. Prof. Dr. Özgür Uğraş Baran

September 2022, 81 pages

A computer code is developed for solving two-dimensional compressible Reynolds-Averaged Navier-Stokes (RANS) equations. The compressible RANS equations are closed with the negative version of the Spalart-Allmaras (SA) turbulence model. Quad-tree-based Cartesian/Quad grids are used to discretize the solution domain. Then, a cell-centered, finite-volume approach is applied to solve turbulent flows. Solution-based mesh adaptivity is used to obtain mesh-free solutions. Since a quad-tree-based data storage is used, mesh refinement and coarsening are done efficiently. Flow variables are reconstructed by using the weighted and unweighted least squares approach. Convective fluxes are formulated with the approximate solver of Roe and limited with Venkatakrishnan's limiter. Formulation of convective terms of the turbulence model is achieved by using first-order upwinding. The gradients used in viscous calculations are obtained using a modified average of the reconstructed variables.

Keywords: Hybrid Grids, Body conforming grids, Turbulence models

ÖZ

KARTESYEN IZGARALAR İÇİN 2D TURBULENT NAVIER-STOKES ÇÖZÜCÜNÜN GELİŞTİRİLMESİ

Ata, Onur
Yüksek Lisans, Makina Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Mehmet Haluk Aksel
Ortak Tez Yöneticisi: Dr. Öğr. Üyesi. Özgür Uğraş Baran

Eylül 2022, 81 sayfa

İki boyutlu sıkıştırılabilir Reynolds-Averaged Navier-Stokes (RANS) denklemlerini çözmek için bir bilgisayar kodu geliştirilmiştir. Sıkıştırılabilir RANS denklemleri, Spalart-Allmaras (SA) türbülans modelinin negatif versiyonu ile kapatılmıştır. Çözüm alanını ayrıklaştırmak için dörtlü ağaç tabanlı Kartezyen/Dörtlü ızgaralar kullanılmıştır. Daha sonra türbülanslı akışları çözmek için hücre merkezli, sonlu hacim yaklaşımı uygulanmıştır. Ağsız çözümler elde etmek için çözüm tabanlı ağ uyarlaması kullanılmıştır. Dörtlü ağaç tabanlı bir veri depolama kullanıldığından, ağ iyileştirme ve kabalaştırma verimli bir şekilde yapılmıştır. Akış değişkenleri, ağırlıklı ve ağırlıksız en küçük kareler yaklaşımı kullanılarak yeniden oluşturulmuştur. Konvektif akılar, yaklaşık Roe çözücüsü ile formüle edilir ve Venkatakrishnan'ın sınırlayıcısı ile sınırlandırılmıştır. Türbülans modelinin konvektif terimlerinin formülasyonu, birinci dereceden yukarı sarma kullanılarak elde edilmiştir. Viskoz hesaplamalarda kullanılan gradyanlar, yeniden yapılandırılmış değişkenlerin değiştirilmiş bir ortalaması kullanılarak elde edilmiştir.

Anahtar Kelimeler: Hibrit Izgaralar, Gövde uyumlu ızgaralar, Türbülans modelleri

Dedicated to my family

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my thesis supervisor, Prof. Dr. M. Haluk Aksel, and co-supervisor Dr. Özgür Uğraş Baran for their guidance during this period. Additionally, I want to thank all CFD team and my friends from graduate school for their guidance and for motivating me. Thank you, Ceren, for your accompany and for being thoughtful during this period. I want to thank my sister and brother-in-law Begüm Ata and Yiğithan Tufan. I want to thank my Grandparents, Bünyamin and Nurten Yılmaz, for their understanding of my business. Last but certainly not least, I thank my mother and father, Fatma and Mustafa Ata, for their endless support from the beginning of my thesis studies. This work is partially supported by TUBITAK 1001 grant 217M455.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

Greeks

α Angle of attack

$\tilde{\alpha}$ Wave strength

γ Specific Heat Ratio

θ Face normal angle

κ Karman constant

$\tilde{\lambda}$ Eigenvalue

μ Dynamic viscosity

μ_{eff} Effective viscosity

μ_t Turbulent viscosity

ν Kinematic viscosity

$\tilde{\nu}$ Turbulence working variable

ρ Density

τ Shear stress

Φ Limiter

χ Intermediate variable

Roman

a Sound Speed

A Area

$c_{b1}, etc.$ Empirical constants in the turbulence model

C_f Skin friction coefficient

C_L Lift coefficient

C_p Pressure coefficient

c_p Specific heat at constant pressure

c_v Specific heat at constant volume

D Destruction

d Wall distance

e Internal energy

e_{k} Kinetic energy

e_{tot} Total energy

$f_{v2}, etc.$ Empirical functions in the turbulence model

h Enthalpy

h_{tot} Total enthalpy

$\tilde{K}$ Right Eigenvector

k Conductivity, Tangential unit vector

l Normal unit vector

m Mass

M_∞	Mach Number
n	Unit normal vector
P	Pressure, Production
Pr	Prandtl Number
Pr_t	Turbulent Prandtl Number
q	Heat flux, Primitive Variable
R	Residual, Gas constant
S	Vorticity strength
$\overrightarrow{S}$	Source Vector
T	Temperature
t	Time
u	x-Velocity
V	Velocity vector, Volume
V_∞	Magnitude of the free-stream velocity
v	y-Velocity
w	Reconstruction weight function
$\overrightarrow{F}$	Inviscid Flux Vector
$\overrightarrow{G}$	Viscous Flux Vector
$\overrightarrow{Q}$	Vector of Conservative Variabes

Operators

∂X	Partial Derivative of X
$\nabla \cdot X$	Divergence of X
$\nabla \times X$	Curl of X
$\int_S X dS$	Facial Integral of X
$\int_V X dV$	Volumetric Integral of X

Abbrevations

FANS	Favre-Averaged Navier-Stokes
NS	Navier-Stokes
PDE	Partial Differential Equation
RANS	Reynolds-Averaged Navier-Stokes
SA	Spalart-Allmaras

CHAPTER 1

INTRODUCTION

This thesis study presents a quad-tree-based Cartesian grid solver to solve steady, compressible, high Reynolds number flows in a two-dimensional domain. The discretization method that the solver uses is a recursive algorithm that generates a combination of different grid types. Quad-tree data structure is used to store generated grids and handle the connectivity. In this method, after Cartesian grid generation is done and cells are cut at the intersection of predetermined boundaries, the remaining sections in the domain are filled with body-conforming quadrilaterals. Each quadrilateral at the outermost boundary of the boundary region has a matched edge with a cut or split cell. The connectivity at the interface is provided by using this relationship. The main aim of this thesis study is to solve fully turbulent flows by using this approach and observe its accuracy and efficiency. After the solution domain is filled with mixed grids, compressible Reynolds-Averaged Navier-Stokes (RANS) equations are solved with a cell-based finite volume approach using a one-equation turbulence model Spalart-Allmaras (SA).

In this section, a general explanation of different grid-generating strategies is given. Then, turbulence models are discussed from a large perspective. Lastly, a brief review is presented of Cartesian cell-based solution approaches.

1.1 Structured and Unstructured Grids

1.1.1 Structured Grids

Structured grids are generated with an ordered index notation according to boundaries presented in the solution domain. It provides implicit connectivity information and no extra effort is required, with simplicity in the solution phase. Mainly there are two different ways to generate structured grids. In the first case, generation is done using a function to compute the coordinates of grids in the solution domain. It is called algebraic grid generation. The second way is solving partial differential equations (PDEs), generally elliptic or hyperbolic, to generate grids. When hyperbolic PDEs are used, mesh generation is accomplished by starting from a given surface mesh and solving PDEs explicitly for each layer. If there is no strict requirement for controlling the shape of the outer layers, this type of PDE can be used for structured mesh generation.

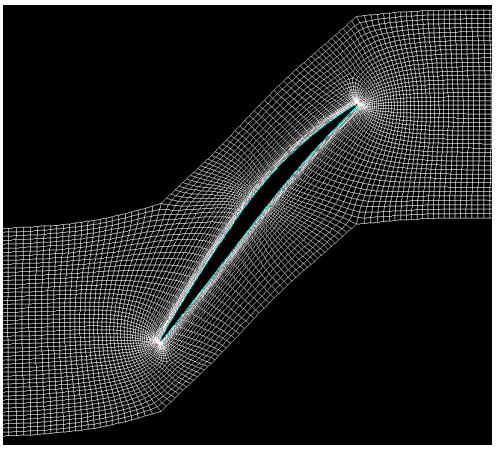


Figure 1.1: Structured grids generated around a rotor blade

1.1.2 Unstructured Grids

Unstructured grids are generally constructed by triangles and tetrahedra in 2 and 3-dimensional domains, respectively. Unlike structured grids, they are not connected by predetermined index definitions. Thus, it is required to make an additional identification procedure to determine the order of the unstructured grid. While it leads to

an extra effort compared to structured ones, it also gives unstructured grid generation flexibility. Domains with complex geometries can be discretized with much less effort by using unstructured grids. Although, traditional unstructured methods require the discretization of body surfaces. To obtain a good match between cells and the surface, users sometimes need to apply grid generation several times, especially for complex geometries. It limits the true automation of the grid generation process.

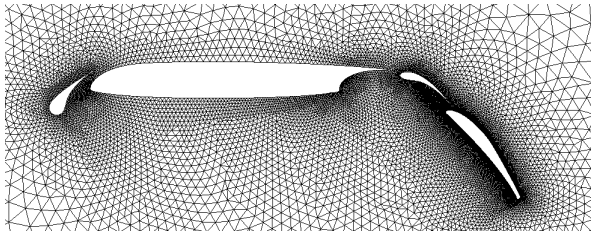


Figure 1.2: Unstructured grids generated around a multi-body airfoil

1.1.3 Cartesian Grids

Cartesian grids are a type of unstructured grid that contains axis-aligned squares in two-dimensional and cubes in a three-dimensional domain. The Cartesian grid approach is applied by recursively dividing the domain according to defined boundaries as a function or discrete points. All the information desired to provide grid communication can be stored by using a data tree. The grid generation process can be automated by cutting the cells at the boundary intersections. Also, with the data structure's help, solution adaptivity and multi-grid methods become favorable for Cartesian grids. Solution adaptions provide mesh-free solutions and automate the mesh refinement process. All in all, Cartesian grids provide robustness and minimize user intervention for the grid generation process, making them a dependable alternative.

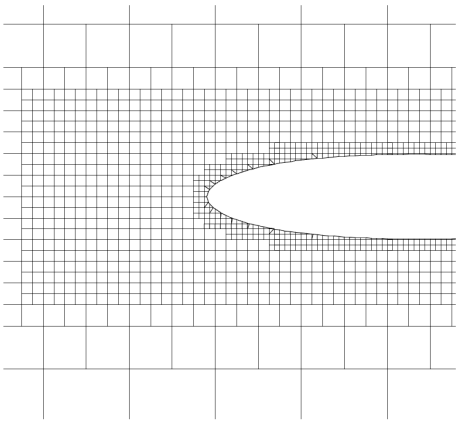


Figure 1.3: Cartesian grids

1.2 Turbulence Modeling

Turbulent flows are one of the most common flow cases in many engineering applications since laminar flows transform into turbulent ones if parameters like the Reynolds Number exceed the critical limit. Therefore, there had been excessive effort to understand and estimate turbulence flows experimentally and numerically.

Experiments are an essential part of understanding the behavior of turbulent flows. Crucial experimental techniques have been developed thanks to researchers in this field, like advanced flow visualization, hot wires, laser-Doppler systems, and many others.

Navier-Stokes (NS) equations can describe turbulent flow by themselves. However, it requires excessive computational power to solve all turbulence flow scales using only NS equations, even for low Reynolds Numbers. Direct Numerical Simulation (DNS) describes the solutions of turbulent flows only using the NS equations. A detailed review of DNS can be found in Moin's [1] study. Although even today's computer power is still a limiting factor for DNS applications, many applications which were thought to be impossible to solve once can be solved by DNS today. For instance, incompressible turbulent pipe flow is solved by Wu and Moin [2] by using 640 million grid points for $Re = 44000$.

Other than DNS, many methodologies are developed to model the turbulent flows by

considering what type of properties need to be estimated. Principally there are five main branches of modeling turbulence which are

• algebraic,

• one-equation,

• multiple-equation,

• second-order closures (Reynolds-stress models),

• large-eddy simulation (LES).

The first three approaches are mostly based on the eddy-viscosity hypothesis offered by Boussinesq. These main approaches have advantages and disadvantages according to relevant problems where turbulence occurs. Thus, there are no distinct criteria that make one better than another.

Algebraic models, also called zero-equation models, calculate the turbulent eddy viscosity using empirical relations. They do not require the solution of any additional equations. The convection and diffusion of eddy viscosity are not taken into account. Thus, they cannot predict complex flows reliably like separated flows. Baldwin and Lomax developed one of the most popular algebraic models, which is still used for simpler flow cases [3].

Diffusive and convective effects are considered in one and two-equation models. In one-equations models, an additional transport equation is used, which is generally for the turbulent kinetic energy. Prandtl's one-equation model is the first example of one-equation models [4].

Two-equation models have two additional transport equations. These models consider not just the turbulent kinetic energy but also the turbulent length scale. K-epsilon and K-omega models are most preferred once among two-equation models.

LES is an approach in which large eddies are solved, and the smallest subgrid-scale (SGS) eddies are modeled. The idea is that the main contribution to Reynolds Stresses comes from the large eddies, while smaller ones have a small contribution. In addition, small eddies have more universal characteristics, which is more appropriate for

modeling. While DNS aims to solve whole scales of turbulence, LES solves larger scales. It reduces the required computational power in comparison with DNS.

In this study, a well-known one-equation model is selected to predict features of turbulent problems. The model was first suggested by Spalart and Allmaras [5], named after their surnames. It is based on the Boussinesq approach. The model is selected because of its proven robustness for aerodynamic applications and lower computational requirement with solving only one more equation.

1.3 A Review of Cartesian Cut-Cell and Mixed Grid Approaches

De Zeeuw [6] used a quadtree-based adaptively-refined Cartesian-grid algorithm to solve Euler equations. His application showed the flexibility of the cut-cell approach with the help of an achievable solution-based refinement approach. He showed that using Cartesian grids to solve Euler equations makes it easier to generate grids, formulate fluxes, and simplify the data structure. Besides its advantages, some disadvantages of using Cartesian cut-cells are mentioned in this study also. For instance, trailing and leading edge cut-cell representations may have poor resolution.

Coirier [7] has applied a similar approach and developed a Navier-Stokes solver. One of the focuses of his study was to investigate several viscous gradient schemes by inspecting their accuracy and positivity.

Delanaye et al.[8] pointed out that the Cartesian grids may cause problems for the simulations of viscous flows. Isotropic refinements may lead to an excessive number of cells near the wall boundaries. Thus, he alleviated this problem by using two different grid types; a body-fitted grid close to the body and a Cartesian grid for the rest of the domain. The study aimed to solve High-Reynold number flows in a two-dimensional domain. They presented a robust viscous term discretization approach for the presence of mesh irregularities. One of their conclusions is that solution adaptive grids may be used to solve wake regions better. A similar method has been applied by Wang [9]. He included adaptive refinement in his approach and used a simple process to calculate viscous fluxes to avoid computational expense. In his later study [10], he implemented a nested multi-grid strategy to increase the convergence rate. Wang

and Chen [11] tried to solve turbulent flows by directly using cartesian grids with anisotropic refinement. They showed the efficiency of their approach in terms of the total number of cells used against isotropic approaches.

Karman [12] proposed an approach that resolves geometry with Cartesian-aligned hexahedrons. Then body-conforming meshes are generated to resolve the viscous boundary layer. He also developed a mesh smoothing strategy to cure skewed cells produced at corner regions.

Several kinds of research are done to use Cartesian cut-cells directly with wall functions to solve turbulent boundary layers efficiently. Berger and Aftosmis [13] addressed the inefficiency of Cartesian meshes while solving boundary layers and tried to mitigate that problem by developing a wall model based on the Spalart-Allmaras turbulence model. They indicate that their wall model cannot approximate the outside of the log-layer region and still needs further development. More recently, they presented a new wall model [14], which can approximate the turbulent boundary layer even for the first point $y+$ value corresponding to the wake region.

Dawes et al. [15] used octree Cartesian cut-cells with the viscous layer generated with a Level Set approach. The approach first uses Cartesian cut-cells to obtain the best integer representation of the body shape. Then, the real representation of the body is achieved by using conformal grids down to the body.

Katz et al. [16] applied a meshless grid communication between off-body Cartesian cells and near-body quad cells to simulate viscous flows. In their approach, the interface is filled up with a point cloud. Then, an edge-based meshless scheme is applied in which NS equations are discretized based on a least-squares approach.

Luo et al. [17] illustrated a two-dimensional hybrid grid approach that combines the advantages of three different grid approaches. They used the unstructured triangular cells as a buffer layer between near-body quadrilateral cells and Cartesian cells, which fills the rest of the domain of interest.

Park et al. [18] proposed a similar hybrid grid approach to alleviating the inaccuracies generated by cut-cells near the body. As the first step of the approach, a body-fitted layer of quads is created by extruding frontal nodes with a function depending on

the minimum normal curvature. The level set method is used to find colliding cells. Secondly, the Cartesian grid fills the rest of the domain. Then, the gap between them is filled with triangles. Özkan's thesis [19] is another example. Her study aimed to combine the orthogonality and directionality of a structured grid, the efficiency and simplicity of a Cartesian grid, and the flexibility and ease of an unstructured grid in an attempt to develop an automatic, robust, and fast hybrid mesh generation method.

Hartmann et al. [20] treated small cut-cell problems using a cell-merging procedure. Their research is the first example of a strictly conservative cut-cell method for three-dimensional cases.

Schneiders et al. [21] presented a conservative Cartesian grid method for viscous flows interacting with moving bodies. They developed a new approach to treat small cut cells based on a weighted Taylor series approach.

The overset-grid approach is another solution alternative for using different types of grids together. The basic idea here is, at first, to generate the grids separately around each geometrical entity in the domain. After that, the grids are combined so that they overlap each other where they meet. Oversetting techniques do not require common boundaries between subdomains; instead, a common or overlapped region is required to match the solutions across boundary interfaces. This makes it easier to generate grids, especially for complex geometries, since each grid can be generated independently from the other grids. The usual procedure uses an interpolation of embedded boundaries to provide the necessary communication among the grids. Benek represented one of the earliest studies about overset grids [22]. The interpolation between the different grids is a crucial stage for a higher-order Chimera method. It is shown that the use of linear interpolation schemes in conjunction with high-order methods leads to a decrease in the global accuracy of the solution[23]. More recently, Ramírez [24] published a study in which higher-order accuracy is achieved by using a moving least squares approach to communicate between overlapping regions.

Additionally, several studies have been conducted about Quad-tree-based Cartesian grid solvers in the Department of Mechanical Engineering at METU. Firstly, Siyahhan [25] used the approach to solve two-dimensional compressible Euler equations. Çakmak [26] took forward the research by implementing the solution adaption pro-

cedure and multi-grid method for two and three-dimensional domains. Lastly, Şahin [27] developed a laminar flow solver and implemented the body-conforming grids into the grid generation process.

1.4 The Main Objective and Outline

The approaches that were presented in the previous section have advantages and handicaps. First of all, although the direct usage of cut-cells increases the ability to obtain a solution with a minimum user intervention, the solution of turbulent flows by using this technique is generally quite inefficient and inaccurate, especially for complex geometries. On the other hand, hybrid grid approaches can handle this problem by using body-fitted orthogonal grids. However, it requires a more complex data structure and unique treatments at the interface of the different grid types.

The grid generation approach used in this study is based on Şahin's thesis study [27]. Cartesian cut-cells first resolve user-defined discrete points of geometries. Then, resolved points are translated in the normal directions of defined line segments. The outer area of the boundary defined by translated points is discretized using a Cartesian cut-cell. The near-body area is filled with quadrilaterals to solve the turbulent boundary layer. Quadrilaterals and cut-cells have a perfect match at the intersection. This idea retains the simplicity of data structure and solution adaptive mesh refinement procedure. Thus, it still provides a highly automatic mesh generation and solution procedure. However, it decreases the flexibility of generation body conforming quadrilaterals. This study investigates this approach's efficiency in solving high Reynolds number flows.

In this study, the introduced concept of mesh generation is used to solve fully turbulent flows. The compressible Reynolds-Averaged Navier-Stokes (RANS) equations are closed using the negative Spalart-Allmaras model. Flow variables are reconstructed using Least Square Approach and limited with Venkatakrishnan's limiter. Viscous fluxes are calculated with a modified averaging approach for gradient terms instead of an additional reconstruction technique. Solution adaptive mesh refinement is used to have mesh-free solutions.

The data structure and mesh generation are explained in Chapter 2. Governing equations of the solution approach of this study are introduced in Chapter 3. In the next chapter, each part of the solution procedure is discussed. In Chapter 5, results obtained from several test cases are presented and discussed. The last chapter summarizes the study presented with outcomes of the study and suggestions for future work.

CHAPTER 2

MESH GENERATION

This chapter explains the data structure used in the solver and the mesh generation approach. A quadtree-based data structure is used to store information which is a suitable selection for the mixed Cartesian-quadtree grid scheme used in this study. At the beginning of this chapter, the data structure is explained. Then, the generation of Cartesian grids and geometric adaptation methodologies are explained. Furthermore, as this study contains the usage of hybrid grids, the generation of boundary layer grids and their implementation into the data structure are also explained.

2.1 Data Structure

A quad-tree data structure is used to generate Cartesian grids. In this approach, the domain is divided into four equal size regions for each refinement. The newly created regions are called children of the split region. This data structure is suitable for Cartesian grid generation. In this aspect, each part is a square Cartesian grid. The first Cartesian grid, which contains the whole solution domain, is called the root cell. Its level is 0. When it is divided into four equal size cells, these cells are stored as children of the root cell. Their level is 1. The same refinement procedure continues recursively until the desired resolution is obtained. For each refinement, the level of generated cells increases by 1. Each cell contains information about its parent and children, stored as pointers. A cell without any children is called a leaf cell. Leaf cells are flagged as computational cells, and they are used in the solution procedure. The location of each cell can be determined, and connectivity can be constructed using the data structure. A simple representation of the quad-tree data structure is shown in

Figure 2.1.

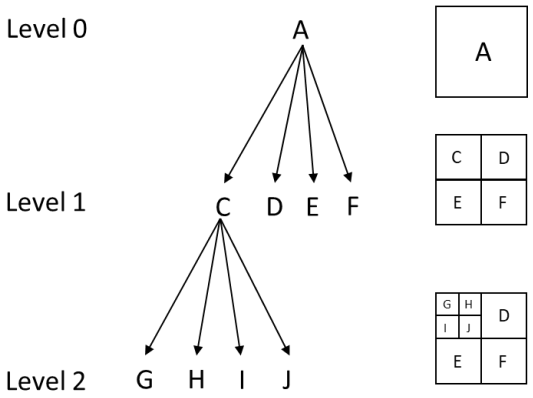


Figure 2.1: Quad-tree data structure

One of the advantages of using a quad-tree data structure is the refinement procedure. Leaf cells can be flagged for refinement according to the given criteria. Then new children are generated and stored by using pointers. Using the data tree, the neighbors of new grids are determined. Similarly, the coarsening procedure can be applied simply by removing the pointers that store children of related cells.

There is no need for another linked list to find a neighbor for each cell. The data tree is used to discover neighbors. By using the tree, firstly, the parent of this cell and then the neighbors of its parents are searched to determine neighbors in four directions. For each refinement, neighbors of new cells are found and stored, whether computational or not, beginning from the root cell. This procedure requires taking up more space in memory. The advantage is that visiting grandparents of an examined cell is not needed. Only its parent and their neighbors are enough to complete the neighbor-finding procedure. An example tree traversal procedure for finding neighbors of a grid is presented in Figure 2.2.

After constructing the data tree, one of the most critical steps is determining the type of cells. Knowing whether cells are outside or inside or intersecting with the given body geometry is crucial. First, all cells' edges are queried to find if they are inside or outside. Then, intersections are found if there are any. The cell type information is stored and retrieved with the help of cell type pointers. Also, two different index values are recorded for each cell according to their types. These are cut and split indices. With the help of these indices, the location of cut and split lines are determined and

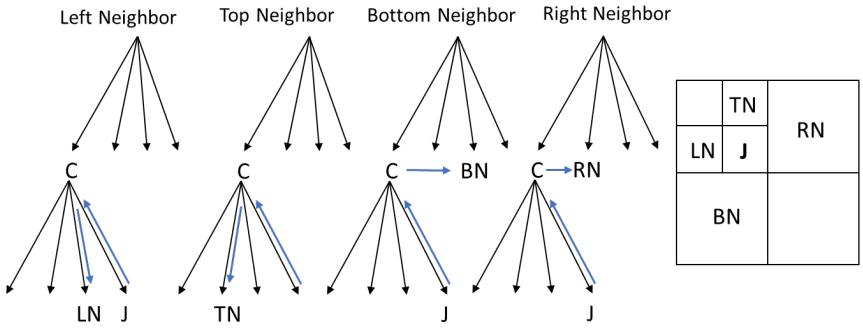


Figure 2.2: An example tree traversal on finding neighbors of cell J

stored.

When the solution procedure starts, the program needs some information about grids like neighbors, their levels, and whether they are computational or not. They can be computed and provided when they are required. Alternatively, they can be stored before the solution begins and are provided when needed. This selection affects the computational time of the same procedure. Thus, the time required may change significantly depending on the preference. In this study, most geometric parameters that are not altered during the solution procedure are computed and stored at the beginning. While it increases memory usage, the advantage is that less computational power is used while solving problems. The selection depends on memory and solution time consideration, which may change with developing hardware specifications. These parameters are stored and accessed with the help of the pointers listed below.

- 9 pointers for geometric parameters (Corner Points, Area, Center, Centroid, Faces, Wall Distance)
- 16 pointers for connectivity (Parent, Children, Neighbors, Split and Quad Cells, Level)
- 5 pointers for cell type (Cell Type, Intersection Points, Square and Split Indices)
- 25 pointers for solution parameters (Residuals, Conserved variables, Gradients, Limiter, Viscosity)
- 2 pointers for solution adaptation (Tau, Ksi)
- 21 static pointers for flux calculations

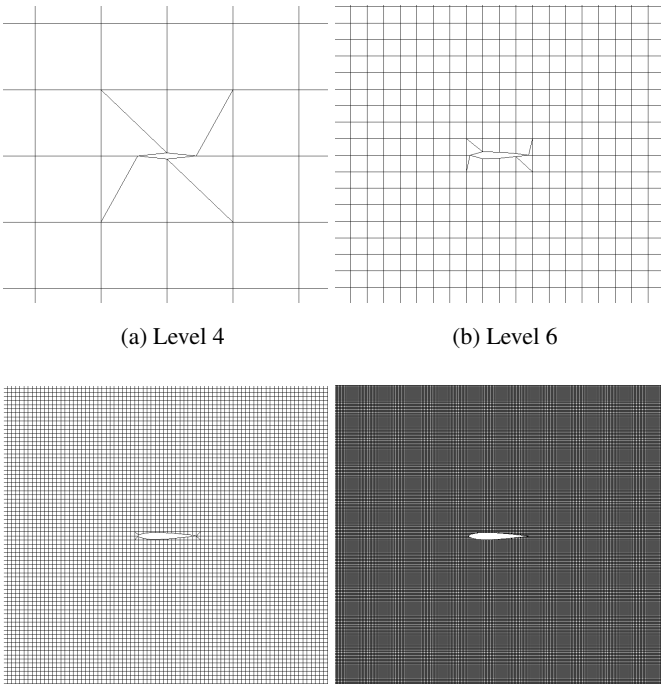
2.2 Cartesian Mesh Generation

The first step to generating an appropriate mesh structure is the generation of the lowest level grid. This grid is called the root grid because it is the root of the whole data structure, and by dividing it isotropically, four equal squares are obtained each time until the final Cartesian mesh is obtained.

2.2.1 Uniform Mesh Generation

The first step of Cartesian mesh generation is dividing the root cell uniformly. A user-defined input, called uniform mesh division level, is used to achieve this step. The root cell is defined until it reaches the user-defined level. As mentioned in the Data Structure section, the level of the root cell is 0. The level of the uniform mesh is increasing for each division. In this step, cells intersecting with the body are found and cut. With the appropriate level of uniform cells, the geometry of a body can be represented well. Figure 2.3 shows the resulting uniform mesh at different levels for the NACA0012 airfoil. As it is seen for increasing levels of uniform mesh, the body's geometry is represented better. The airfoil body is well captured after ten cycles of uniform mesh generation. However, using only uniform grids causes an excessive number of cells. The number of grids generated for ten cycles is 1,048,238. Using it directly to solve problems is inefficient in terms of memory and computational power. Cells far away from the body are unnecessarily refined due to this procedure. Thus several adaptations are applied in this study in order to have a grid structure that represents the body satisfactorily, while it is more efficient than a uniformly generated grid. In general, these adaptations represent the body geometry and help the mesh generation process more robust. In total, three geometric adaptations are applied. These are,

- Box Adaptation
- Cut-Split Adaptation
- Curvature Adaptation



(a) Level 4 (b) Level 6

(c) Level 8 (d) Level 10

Figure 2.3: Uniformly generated Cartesian grids for different levels

2.2.2 Box Adaptation

This adaptation procedure aims to refine grids closer to wall boundaries. Part of the solution domain, which is far from any geometry, generally does not need to be refined because flow variables do not change much with respect to location. Thus, refining the regions nearer to the bodies while having coarser grids away from the bodies is reasonable. The Box adaptation procedure aims to define a rectangular region that contains grids close to the body and refine these grids according to the given refinement criteria. The size of the rectangular area, box, is determined with user input as boundary size factors in both x and y directions. Global maximum coordinate differences of geometry in x and y directions are multiplied by this factor. The length and height of the box are defined as follows

$$L = (S_x - 1)\frac{(X_{max} - X_{min})}{2}, \tag{2.1}$$

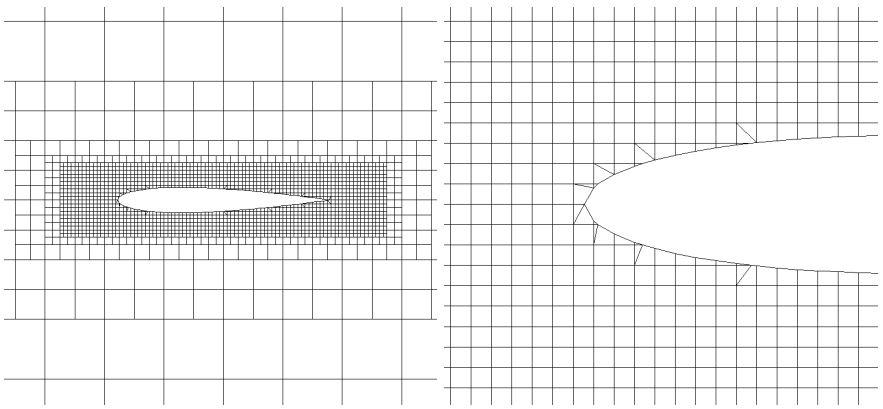
$$H = (S_y - 1)\frac{(Y_{max} - Y_{min})}{2}, \tag{2.2}$$

where S_x and S_y are the boundary size factors in the x and y directions, respectively. X and Y represent the maximum and minimum values of nodal positions for all geometries in the solution domain.

Cells inside the box are queried whether they provide the conditions determined by another user input, the body division factor. The ones that do not provide are refined. The criterion is for refinement can be specified as follows

$$max(X_{max} - X_{min}, Y_{max} - Y_{min})F_D \leq \frac{D}{2^L}, \tag{2.3}$$

where D is domain size, size of the root cell, L is the level of the cell, F_D is the body division factor. Figure 2.4 shows Cartesian grids generated with box adaptation around NACA0012 airfoil.



(a) (b)

Figure 2.4: Box adaptation

2.2.3 Cut-Split Adaptation

Cut and split cells must have a higher resolution because they are in the neighborhood of the bodies where higher gradients are presented. Also, it helps to have a good representation of the geometry without refining the whole grid structure. After refining cut and split cells, their neighbors are also refined in order to have a smooth transition from finer cells to coarser cells at outer regions. Figure 2.5 shows Cartesian grids generated with cut-split adaptation around the leading edge of the NACA0012 airfoil.

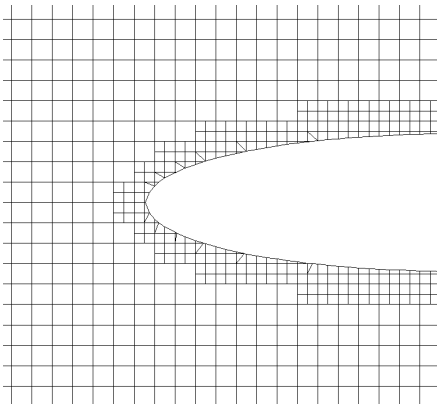


Figure 2.5: Cut-Split adaptation

2.2.4 Curvature Adaptation

This procedure is based on examining the angles between successive cut-cell and determining whether they need to be refined or not. These grids are refined if this angle is below a defined threshold angle. This procedure is achieved by constructing a triangle by using the intersection point of consequent lines and their other ends. Then, the angle at the point of intersection is computed. Lastly, this angle is compared with the threshold angle; if it is less, cells are refined. In the figure, the procedure is represented. Figure 2.6 shows Cartesian grids generated with curvature adaptation around the leading edge of the NACA0012 airfoil.

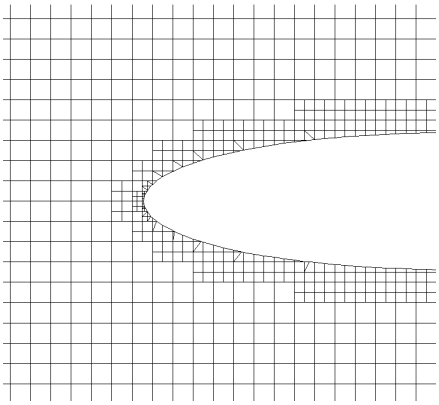


Figure 2.6: Curvature adaptation

2.2.5 One Level Difference Check

In this study, the maximum allowed level difference between two adjacent Cartesian grids is one. After each adaptation cycle, the data structure is checked to see if there is more than one level difference between the neighboring cells. If there is, lower-level neighbor cells are refined until the difference decreases to one. This procedure simplifies the connectivity handling and solution approach and provides a smoother mesh structure.

2.3 Boundary Layer Mesh Generation

One of the aims of this study is to use a body-conforming grid structure to solve near-body flow more accurately. Each conforming grid has direct connectivity to a cut or split cell or each other. While it decreases the flexibility of the approach, it simplifies the data structure and guarantees a conserved solution.

Before generating near-body grids, a puffed geometry is required. After the geometry is obtained, the edges of the cut-split cells are connected to the body. Lastly, quadrilateral body conforming grids are generated starting from the cut-split cells' edges. Details of the process are explained in the following sections.

2.3.1 Setting the Boundary Layer Thickness

Turbulent boundary layer thickness is calculated based on Reynolds Number. It is estimated by using Von Karman's momentum integral approach, which is defined as,

$$\delta = \frac{5}{Re^{0.2}}. \tag{2.4}$$

Although this assumption may be sufficient for some cases, generally, it is required to modify total boundary layer thickness, especially for complex geometries. Thus, users may use different empirical approaches to set a boundary layer thickness for specific problems.

Another essential parameter to be considered before generating body conforming grids is the first layer thickness. The first layer thickness should be sufficiently thin to solve the viscous sub-layer region accurately. Non-dimensional wall distance y^+ should be smaller than one. It is defined as

$$y^+ = \frac{yu_\tau}{\nu},\ u_\tau = \sqrt{\frac{\tau_w}{\rho}}, \tag{2.5}$$

where u_τ is the friction velocity. Estimating the wall shear and other parameters gives an approximate first layer thickness for the desired y^+.

2.3.2 Puffed Geometry Generation

The geometry is defined with points rather than a function. For each successive point, line segments are defined, and their normal vectors are found. Then points of the line segments are shifted in the direction of these vectors by an amount of boundary layer thickness. New line segments are intersected, and puffed geometry is defined as it is shown in Figure 2.7. After obtaining the first geometry, two more additional applications are applied to have a more proper geometry for quad grid generation. These are,

- Handling Convex Geometries

- Negative Volume Check

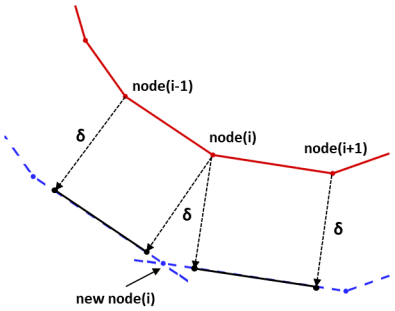


Figure 2.7: Creation of new nodes of puffed geometry

Highly convex parts are handled by defining new nodes from the edge of the convex part. If the angle between the two line segments is lower than 60 degrees, five equally distributed new points are defined by considering the boundary layer thickness. If it is lower than 120 degrees, three new points are defined. Figures 2.8, 2.9 illustrate puffed node calculation without and with the adaptation process.

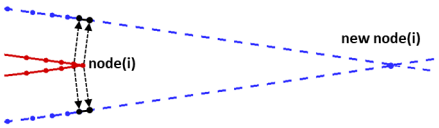
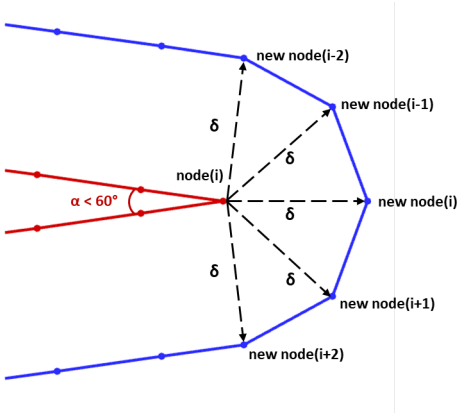
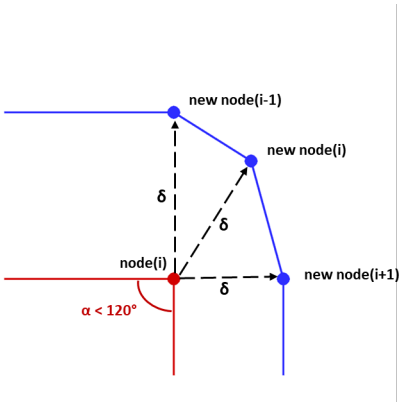


Figure 2.8: Convex part without handling



(a)



(b)

Figure 2.9: Addition of extra nodes to handle convex geometries

Direct extrusion of concave geometries may create negative volumes at the interface of cut cells and boundary layer cells. Thus, the puffed geometry is checked before the mesh generation process to eliminate this problem. Until none of the line segments intersect, the geometry is checked, and the nodes inside the puffed geometry are eliminated.

2.3.3 Quad Cell Generation

After the boundary layer points are determined, Cartesian grids are generated around these points. Cartesian cells are cut and split along the boundary layer. Then the

space between the new boundary and the geometry is filled with quadrilateral grids. The first layer of quad cells is neighbor to cut and split cells. The connectivity relation at the boundary is provided by two pointers, "quad1" and "inclusiveOfQuads." Each cut and split cell is neighbor to at least one quad cell, and this information is stored using the "quad1" pointer. Also, the "inclusiveofQuads" pointer stores neighboring cut or split cells for quad cells. Furthermore, cut cells may have two quad cells at their two different faces. In this situation, "quad1 and "quad2" pointers are used to store these quad cells. Figure 2.10 represents a mixed grid generated around NACA 0012 airfoil.

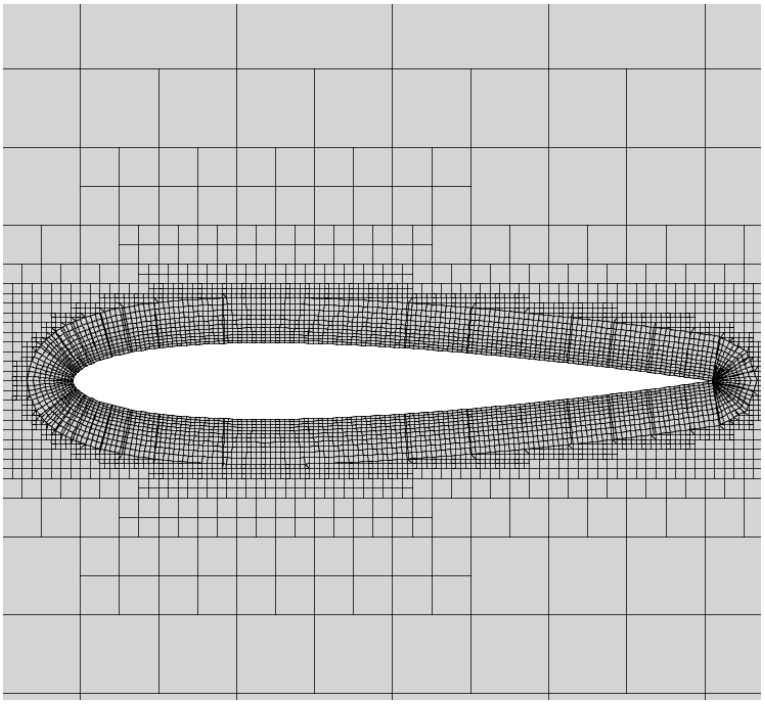


Figure 2.10: Mixed grid generated around NACA0012 airfoil

CHAPTER 3

GOVERNING EQUATIONS

After covering the mesh generation method, the governing equations that describe two-dimensional fully turbulent compressible flows are described in this chapter.

Firstly, Navier-Stokes equations are presented in their integral form for two-dimensional compressible flows.

Secondly, the compressible Reynolds-Averaged Navier-Stokes equations, also known as the Favre-Averaged Navier-Stokes equations, are described briefly. Non-dimensional forms of these governing equations are also presented since the code used in this study uses non-dimensional parameters.

Finally, Spalart-Allmaras is selected to model the turbulence. Its well-known baseline model is expressed. Then a conservative form of the SA model with limitations in its production term is demonstrated. In addition, the negative SA equation is used to prevent possible numerical problems. After the turbulence model explanation, the complete RANS-SA system is shown. The boundary conditions for corresponding governing equations are also explained at the end of the chapter.

3.1 Navier-Stokes Equations

Navier-Stokes Equations are a system of equations that describe the dynamical behavior of a fluid flow. This set of equations contains the conservation laws for mass, momentum, and energy. In this study, the solution domain of our interest is two-dimensional. The two-dimensional integral form of the compressible Navier-Stokes Equations is defined as

$$\frac{\partial}{\partial t}\int_V \overrightarrow{Q}dV + \int_S (\overrightarrow{F} - \overrightarrow{G})dS = 0. \tag{3.1}$$

where $\overrightarrow{Q}$ is the vector of conserved variables. It is given as

$$\overrightarrow{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}. \tag{3.2}$$

$\overrightarrow{F}$ is the inviscid flux vector. Its components are

$$\overrightarrow{F} = \begin{bmatrix} \rho u \boldsymbol{n_x} + \rho v \boldsymbol{n_y} \\ (\rho u^2 + P)\boldsymbol{n_x} + \rho uv \boldsymbol{n_y} \\ \rho uv \boldsymbol{n_x} + (\rho v^2 + P)\boldsymbol{n_y} \\ \rho uH \boldsymbol{n_x} + \rho vH \boldsymbol{n_y} \end{bmatrix}. \tag{3.3}$$

where H is the total enthalpy, and E is the total energy. The relation between these two terms is defined as

$$H = E + \frac{p}{\rho}. \tag{3.4}$$

$\overrightarrow{G}$ vector is the viscous flux vector. Its components contain the viscous stress terms , $\tau_{xx}, \tau_{xy}, \tau_{yy}$ and heat conduction terms , q_x, q_y.

$$\overrightarrow{G} = \begin{bmatrix} 0 \\ \tau_{xx}\boldsymbol{n_x} + \tau_{xy}\boldsymbol{n_y} \\ \tau_{yx}\boldsymbol{n_x} + \tau_{yy}\boldsymbol{n_y} \\ (u\tau_{xx} + v\tau_{xy} - q_x)\boldsymbol{n_x} + (v\tau_{yy} + u\tau_{yx} - q_y)\boldsymbol{n_y} \end{bmatrix} \tag{3.5}$$

The heat conduction terms are formulated by following Fourier's heat conduction law.

$$q_x = -k\frac{\partial T}{\partial x} \tag{3.6}$$

$$q_y = -k\frac{\partial T}{\partial y} \tag{3.7}$$

3.1.1 The Working Fluid

In this solver, the air is selected as the working fluid, and it is a calorically perfect gas if the flow is not at hyper-sonic speeds. Thus, its specific heat capacity is constant. There are five unknown parameters represented in Navier-Stokes Equations, while the number of the equations is four. Since one more equation is required for closure, the equation of the state of a perfect gas is also used.

$$p = \rho R T \tag{3.8}$$

Other thermodynamic correlations for perfect gases are

$$R = c_p - c_v, \tag{3.9}$$

$$\gamma = \frac{c_p}{c_v}, \tag{3.10}$$

$$h = c_p T. \tag{3.11}$$

where R is the specific gas constant, c_p and c_v are specific heat capacities under constant pressure and volume, respectively, and h is the enthalpy per unit mass.

Pressure in terms of conserved variables can be found by using the above relations and the total enthalpy equation (3.4).

$$p = (\gamma - 1)\left(\rho E - \frac{\rho(u^2 + v^2)}{2}\right) \tag{3.12}$$

Since air is a Newtonian fluid, shear stress is proportional to the velocity gradient. Viscous stresses can be calculated by using the below relations [28]

$$\tau_{xx} = 2\mu\left(\frac{\partial u}{\partial x} - \frac{1}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right), \tag{3.13}$$

$$\tau_{yy} = 2\mu\left(\frac{\partial v}{\partial y} - \frac{1}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right), \tag{3.14}$$

$$\tau_{xy} = \tau_{yx} = \mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right). \tag{3.15}$$

The dynamic viscosity μ, can be calculated by using the Sutherland formula for air. It is defined by using reference free-stream values as

$$\frac{\mu}{\mu_\infty} = \left[\frac{T_\infty + 110}{T + 110} \right] . \left(\frac{T}{T_\infty} \right)^{3/2}. \tag{3.16}$$

where μ_∞ is free-stream dynamic viscosity and taken as the reference. T_∞ is the corresponding free-stream temperature, and the temperature unit is selected as Kelvin.

For liquids, the thermal conductivity can be taken as a constant. However, for gases, their dependency on temperature can be related by using μ

$$k = c_p \frac{\mu}{Pr}. \tag{3.17}$$

Pr is the Prandtl Number and its value $Pr = 0.72$ for the air.

3.1.2 Non-dimensional Formulation of Navier-Stokes Equation

As already mentioned, this research aims to develop a CFD code to calculate fully turbulent flows over various geometries like airfoils. The comparison cases generally have been studied with experimental applications like wind tunnel tests. However, there is no global standardization of measurement of these experimental applications. It may create difficulty in following up units of each parameter for every different application. Thus, a non-dimensionalized form of the Navier-Stokes Equation is used in this research to alleviate this problem. The dimensionless parameters that are used for non-dimensionalizing the governing equations are defined as

$$\begin{aligned} x^* &= \tfrac{x}{L}, \quad x^* = \tfrac{y}{L}, \quad t^* = \tfrac{t a_\infty}{L}, \\ u^* &= \tfrac{u}{a_\infty}, \quad v^* = \tfrac{v}{a_\infty}, \quad k^* = \tfrac{k}{k_\infty}, \\ p^* &= \tfrac{p}{\rho a_\infty^2}, \quad \rho^* = \tfrac{\rho}{\rho_\infty}, \quad \mu^* = \tfrac{\mu}{\mu_\infty}. \end{aligned} \tag{3.18}$$

and dimensionless free-stream values are defined as

$$
\begin{aligned}
u^*_\infty &= M_\infty cos\alpha, \\
v^*_\infty &= M_\infty sin\alpha, \\
\rho^*_\infty &= 1, \\
p^*_\infty &= \tfrac{1}{\gamma}, \\
c^*_\infty &= 1.
\end{aligned}
\tag{3.19}
$$

Substituting dimensionless parameters into governing equations (3.1) and further modifications give a non-dimensional set of equations

$$
\frac{\partial}{\partial t}\int_V \overrightarrow{Q}^* dV + \int_S (\overrightarrow{F}^* - \frac{M_\infty}{Re}\overrightarrow{G}^*) dS = 0, \tag{3.20}
$$

where asterisks state that corresponding vectors of variables are dimensionless. Also, the remaining parameters after non-dimensionalizing are grouped in two different dimensionless groups. These are Reynolds Number and Mach Number. They are defined as,

$$
Re = \frac{\rho_\infty V_\infty L}{\mu_\infty}, \quad M_\infty = \frac{u_\infty}{a_\infty}, \tag{3.21}
$$

where V_∞ is the magnitude of the free-stream velocity.

$$
V_\infty = \sqrt{u^2_\infty + v^2_\infty} \tag{3.22}
$$

Corresponding vectors with non-dimensional variables have the same parameters compared to dimensional ones (3.2, 3.3, 3.5), except that heat conduction terms are defined differently.

$$
q^*_x = -\frac{1}{\gamma - 1}\frac{\mu}{Pr}\frac{\partial T^*}{\partial x^*} \tag{3.23}
$$

$$
q^*_y = -\frac{1}{\gamma - 1}\frac{\mu}{Pr}\frac{\partial T^*}{\partial y^*} \tag{3.24}
$$

Temperature gradient term can be re-organized by using the thermodynamic relations and attained dimensionless free-stream parameters as it is shown below.

$$q_x^* = -\frac{\gamma}{\gamma - 1}\frac{\mu}{Pr}\frac{\partial\left(p^*/\rho^*\right)}{\partial x^*} \tag{3.25}$$

$$q_y^* = -\frac{\gamma}{\gamma - 1}\frac{\mu}{Pr}\frac{\partial\left(p^*/\rho^*\right)}{\partial y^*} \tag{3.26}$$

3.2 Favre-Averaged Navier-Stokes Equations

Favre-Averaged Navier-Stokes (FANS) equations are a system of transport equations for solving compressible turbulent flows. FANS equations are derived by using density-weighted averaging operations on flow variables except for pressure and density, which was proposed by Favre [29]. After the flow parameters are decomposed into their mean and fluctuating parts, they are substituted into Navier-Stokes equations, and density-weighted averaging operations are performed. Density and pressure are averaged by using Reynolds averaging. Details of this application can be found in various sources [30, 31]. The resulting system of equations is usually called Favre averaged mean conservation equations.

The resulting system of equations is somehow similar to the NS equations but has additional unknown terms. These additional terms create a closure problem and need to be modeled. Among the various forms of the Spalart-Allmaras models, the selected form for this study is a linear eddy viscosity model and uses the Boussinesq assumption to model the Reynolds stress tensor.

$$\widetilde{\tau}_{ij} = 2\mu_{eff}\left(S_{ij} - \frac{1}{3}\frac{\partial\widetilde{u}_k}{\partial x_k}\delta_{ij} - \frac{2}{3}\bar{\rho}k\delta_{ij}\right) \tag{3.27}$$

Turbulent heat flux is modeled by using a Reynolds analogy,

$$q_{tj} = \frac{\mu_t c_p}{Pr_t}\frac{\partial\tilde{T}}{\partial x_j}, \tag{3.28}$$

where Pr_t is the turbulent Prandtl Number taken as 0.9, and μ_t is the turbulent eddy viscosity.

Lastly, molecular diffusion and turbulent transport terms are neglected in the Spalart Allmaras model used in this study.

The final form of the FANS equations in weak conservation form is

$$\frac{\partial}{\partial t}\int_V \overrightarrow{Q}\,dV + \int_S (\overrightarrow{F} - \overrightarrow{G})dS - \int_V \overrightarrow{S}\,dV = 0. \tag{3.29}$$

where $\overrightarrow{Q}$ is vector of conserved variables, $\overrightarrow{F}$ and $\overrightarrow{G}$ are inviscid and viscous flux vectors, and $\overrightarrow{S}$ is the source vector. For a two-dimensional domain, they are given as

$$\overrightarrow{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix},\ \overrightarrow{F} = \begin{bmatrix} \rho u \boldsymbol{n_x} + \rho v \boldsymbol{n_y} \\ (\rho u^2 + P)\boldsymbol{n_x} + \rho uv \boldsymbol{n_y} \\ \rho uv \boldsymbol{n_x} + (\rho v^2 + P)\boldsymbol{n_y} \\ \rho uH \boldsymbol{n_x} + \rho vH \boldsymbol{n_y} \end{bmatrix}, \tag{3.30}$$

$$\overrightarrow{G} = \begin{bmatrix} 0 \\ \tau_{xx}\boldsymbol{n_x} + \tau_{xy}\boldsymbol{n_y} \\ \tau_{yx}\boldsymbol{n_x} + \tau_{yy}\boldsymbol{n_y} \\ (u\tau_{xx} + v\tau_{xy} - q_x)\boldsymbol{n_x} + (v\tau_{yy} + u\tau_{yx} - q_y)\boldsymbol{n_y} \end{bmatrix}. \tag{3.31}$$

These vectors have the same form as the vectors defined in equations (3.2, 3.3, 3.5). The only differences are the definitions of shear stresses and heat fluxes. Shear stresses are defined as,

$$\tau_{xx} = 2\mu_{eff}\left(\frac{\partial u}{\partial x} - \frac{1}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right),$$

$$\tau_{yy} = 2\mu_{eff}\left(\frac{\partial v}{\partial y} - \frac{1}{3}\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)\right), \tag{3.32}$$

$$\tau_{xy} = \tau_{yx} = \mu_{eff}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right),$$

where μ_{eff} is the effective viscosity, the sum of laminar and turbulent viscosity.

$$\mu_{eff} = \mu + \mu_t \tag{3.33}$$

The turbulent heat fluxes are given as,

$$q_x = k_{eff}\frac{\partial T}{\partial x}, \quad q_y = k_{eff}\frac{\partial T}{\partial y}, \tag{3.34}$$

and effective thermal conductivity, k_{eff} is defined as,

$$k_{eff} = c_p\left(\frac{\mu}{Pr} + \frac{\mu_T}{Pr_T}\right). \tag{3.35}$$

3.3 Spalart Allmaras One-Equation Turbulence Model

The eddy viscosity μ_t resulting from the Boussinesq assumption is modeled by using the Spalart Allmaras one equation turbulence model. This research is based on the so-called "standard" version of the SA equation to model turbulence [5]. The eddy viscosity μ_t evaluated by

$$\mu_t = \rho\tilde{\nu}f_{v1}, \tag{3.36}$$

where the function f_{v1} and χ are

$$f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \tag{3.37}$$

$$\chi = \frac{\tilde{\nu}}{\nu}. \tag{3.38}$$

The corresponding transport equation is

$$\frac{\partial \tilde{\nu}}{\partial t} + u_j \frac{\partial \tilde{\nu}}{\partial x_j} = P - D + \frac{1}{\sigma}\left[\frac{\partial}{\partial x_j}\left((\nu + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right) + c_{b2}\frac{\partial \tilde{\nu}}{\partial x_i}\frac{\partial \tilde{\nu}}{\partial x_i}\right]. \tag{3.39}$$

where P and D indicate production and destruction terms respectively, which are

$$P = c_{b1}\left(1 - f_{t2}\right)\tilde{S}\tilde{\nu}, \tag{3.40}$$

$$D = \left[c_{w1} f_w - \frac{c_{b1}}{\kappa^2} f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2. \tag{3.41}$$

$\tilde{S}$ is the modified vorticity and is defined as

$$\tilde{S} = S + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2}. \tag{3.42}$$

and S is the magnitude of the vorticity, that is,

$$S = abs\left(\frac{\partial u_i}{\partial x_i} - \frac{\partial u_j}{\partial x_i}\right). \tag{3.43}$$

The remaining empirical constants and functions are given by

$$f_w = g\left[\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right]^{1/6}, \quad g = r + c_{w2}(r^6 - r), \quad r = min\left(\frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2}, r_{lim}\right), \tag{3.44}$$

$$f_{v2} = 1 - \frac{\chi}{1+\chi f_{v1}}, \quad c_{w1} = c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma, \quad f_{t2} = c_{t3} exp\left(-c_{t4}\chi^2\right),$$
$$c_{b1} = 0.1355, \qquad \sigma = 2/3, \qquad c_{b2} = 0.622,$$
$$c_{v1} = 7.1, \qquad \kappa = 0.41, \qquad c_{w2} = 0.3,$$
$$c_{w3} = 2.0, \qquad r_{lim} = 10, \qquad c_{t3} = 1.2,$$
$$c_{t4} = 0.5.$$

The first two terms in equation (3.39) are similar to other transport equations. The first term on the right-hand side represents the generation of turbulence. When there are velocity gradients, shear, in the mean flow, turbulence is generated. The second

term is the destruction term. As the distance from a wall goes to zero, f_w also goes to zero. The third term is the diffusion term. This term has an additional non-linear term to control the spreading of the wake at the edge of the turbulent region.

An equivalent conservative form of the standard equation (3.39) can be derived by summing the product of the SA transport equation with density and the product of the mass conservation equation with the working variable, $\tilde{\nu}$ of the model [32]. The final form is represented below.

$$\begin{gathered}\frac{\partial(\rho\tilde{\nu})}{\partial t}+\frac{\partial(\rho u_j\tilde{\nu})}{\partial x_j}= \\ P-D+\frac{1}{\sigma}\left[\frac{\partial}{\partial x_j}\left(\rho\left(\nu+\tilde{\nu}\right)\frac{\partial\tilde{\nu}}{\partial x_j}\right)+\rho c_{b2}\frac{\partial\tilde{\nu}}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i}\right] \\ -\frac{1}{\sigma}(\nu+\tilde{\nu})\frac{\partial\rho}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i}\end{gathered} \tag{3.45}$$

The production and destruction terms are defined as follows:

$$P=\rho c_{b1}\left(1-f_{t2}\right)\tilde{S}\tilde{\nu}, \tag{3.46}$$

$$D=\rho\left[c_{w1}f_w-\frac{c_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2. \tag{3.47}$$

3.3.1 The Negative SA Model

The original SA model claims positive solutions for positive boundaries and initial conditions. However, the solution may become negative in the cases of coarse grid solutions. The original model allows only positive values of the working variable. The negative SA model is proposed to solve this issue [32]. The negative model is the same as the original model (3.45) for $\tilde{\nu}$ equal or greater than zero. If $\tilde{\nu}$ becomes negative, the following equation is solved.

$$\begin{gathered}\frac{\partial(\rho\tilde{\nu})}{\partial t}+\frac{\partial(\rho u_j\tilde{\nu})}{\partial x_j}= \\ P_n-D_n+\frac{1}{\sigma}\left[\frac{\partial}{\partial x_j}\left(\rho\left(\nu+\tilde{\nu}f_n\right)\frac{\partial\tilde{\nu}}{\partial x_j}\right)+\rho c_{b2}\frac{\partial\tilde{\nu}}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i}\right] \\ -\frac{1}{\sigma}(\nu+\tilde{\nu}f_n)\frac{\partial\rho}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i},\end{gathered} \tag{3.48}$$

where

$$P_n = \rho c_{b1}\left(1 - c_{t3}\right) S\tilde{\nu}, \quad D_n = -\rho c_{w1}\left(\frac{\tilde{\nu}}{d}\right)^2, \quad f_n = \frac{c_{n1} + \chi^3}{c_{n1} - \chi^3}. \tag{3.49}$$

with $c_{n1} = 16$.

3.3.2 Limiting Minimum Value of Modified Vorticity $\tilde{S}$

The modified vorticity $\tilde{S}$ may be zero or negative because f_{v2} term can be negative over a range of X. S-A functions are not calibrated for negative values since they may create numerical problems. There are various ways to handle that situation. Three of them are,

1. Limiting $\tilde{S}$ to be greater than zero.
2. Limiting $\tilde{S}$ to be greater than $0.3 * S_{ij}$.
3. Another approach that uses the identical definition of modified vorticity for $\tilde{S} > 0.3 * S_{ij}$ but still ensures positive values for all other vorticity magnitude values, and it is continuous[32].

The third approach is selected to limit $\tilde{S}$ in this study. The continuous approach is defined below.

$$\tilde{S} = \left\{ \begin{array}{cc} S + \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2} & for \quad \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2} \geq -c_{v2} S \\ S + \frac{S(c_{v2}^2 S + c_{v3} \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2})}{(c_{v3} - 2c_{v2})S - \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2}} & for \quad \frac{\tilde{\nu}}{\kappa^2 d^2} f_{v2} < -c_{v2} S \end{array} \right\} \tag{3.50}$$

3.3.3 Non-dimensional Form of SA Equation

Non-dimensional form of the Spalart-Allmaras equation can be obtained by using the previously mentioned non-dimensional parameters in chapter 3.1.2 and including the non-dimensional turbulence variable $\tilde{\nu}^* = \frac{\tilde{\nu}}{\nu_\infty}$.

$$\begin{gathered}\frac{\frac{\rho_\infty \tilde{\nu}_\infty}{\frac{L}{a_\infty}} \frac{\partial(\rho^* \tilde{\nu}^*)}{\partial t^*} + \frac{\rho_\infty a_\infty \tilde{\nu}_\infty}{L} \frac{\partial\left(\rho^* u_j^* \tilde{\nu}^*\right)}{\partial x_j^*} =}{P - D + \left(\frac{\tilde{\nu}_\infty^2 \rho_\infty}{L^2}\right) \frac{1}{\sigma} \left[\frac{\partial}{\partial x_j^*}\left(\rho\left(\nu^* + \tilde{\nu}^*\right) \frac{\partial \tilde{\nu}^*}{\partial x_j^*}\right) + \rho c_{b2} \frac{\partial \tilde{\nu}^*}{\partial x_i^*} \frac{\partial \tilde{\nu}^*}{\partial x_i^*}\right]} \\ - \left(\frac{\tilde{\nu}_\infty^2 \rho_\infty}{L^2}\right) \frac{1}{\sigma} (\nu + \tilde{\nu}) \frac{\partial \rho}{\partial x_i} \frac{\partial \tilde{\nu}}{\partial x_i}\end{gathered} \tag{3.51}$$

$$P = \left(\frac{\tilde{\nu}_\infty^2 \rho_\infty}{L^2}\right) \rho^* c_{b1} \left(1 - f_{t2}\right) \tilde{S}^* \tilde{\nu}^* \tag{3.52}$$

$$D = \left(\frac{\tilde{\nu}_\infty^2 \rho_\infty}{L^2}\right) \rho^* \left[c_{w1} f_w - \frac{c_{b1}}{\kappa^2} f_{t2}\right] \left(\frac{\tilde{\nu}^*}{d^*}\right)^2 \tag{3.53}$$

Multiplying the equation with $\frac{L}{\rho_\infty a_\infty \tilde{\nu}_\infty}$ results in

$$\begin{gathered}\frac{\partial(\rho^* \tilde{\nu}^*)}{\partial t^*} + \frac{\partial\left(\rho^* u_j^* \tilde{\nu}^*\right)}{\partial x_j^*} = \\ P - D + \left(\frac{\tilde{\nu}_\infty}{a_\infty L}\right) \frac{1}{\sigma} \left[\frac{\partial}{\partial x_j^*}\left(\rho\left(\nu^* + \tilde{\nu}^*\right) \frac{\partial \tilde{\nu}^*}{\partial x_j^*}\right) + \rho c_{b2} \frac{\partial \tilde{\nu}^*}{\partial x_i^*} \frac{\partial \tilde{\nu}^*}{\partial x_i^*}\right] , \\ - \left(\frac{\tilde{\nu}_\infty}{a_\infty L}\right) \frac{1}{\sigma} (\nu + \tilde{\nu}) \frac{\partial \rho}{\partial x_i} \frac{\partial \tilde{\nu}}{\partial x_i}\end{gathered} \tag{3.54}$$

$$P = \left(\frac{\tilde{\nu}_\infty}{a_\infty L}\right) \rho^* c_{b1} \left(1 - f_{t2}\right) \tilde{S}^* \tilde{\nu}^*, \tag{3.55}$$

$$D = \left(\frac{\tilde{\nu}_\infty}{a_\infty L}\right) \rho^* \left[c_{w1} f_w - \frac{c_{b1}}{\kappa^2} f_{t2}\right] \left(\frac{\tilde{\nu}^*}{d^*}\right)^2. \tag{3.56}$$

Equating the group of parameters $\frac{\tilde{\nu}_\infty}{a_\infty L}$ to $\frac{M_\infty}{Re}$, resulting in

$$\begin{gathered}\frac{\partial(\rho^* \tilde{\nu}^*)}{\partial t^*} + \frac{\partial\left(\rho^* u_j^* \tilde{\nu}^*\right)}{\partial x_j^*} = \\ P - D + \left(\frac{M_\infty}{Re}\right) \frac{1}{\sigma} \left[\frac{\partial}{\partial x_j^*}\left(\rho\left(\nu^* + \tilde{\nu}^*\right) \frac{\partial \tilde{\nu}^*}{\partial x_j^*}\right) + \rho c_{b2} \frac{\partial \tilde{\nu}^*}{\partial x_i^*} \frac{\partial \tilde{\nu}^*}{\partial x_i^*}\right], \\ - \left(\frac{M_\infty}{Re}\right) \frac{1}{\sigma} (\nu + \tilde{\nu}) \frac{\partial \rho}{\partial x_i} \frac{\partial \tilde{\nu}}{\partial x_i}\end{gathered} \tag{3.57}$$

$$P = \left\{ \begin{array}{lll} \rho^* c_{b1} \left(1 - f_{t2}\right) \tilde{S}^* \tilde{\nu}^* & for & \tilde{\nu}^* > 0 \\ \rho^* c_{b1} \left(1 - c_{t3}\right) S^* \tilde{\nu}^* & for & \tilde{\nu}^* \leq 0 \end{array} \right\}, \tag{3.58}$$

where

$$\tilde{S}^* = S + \left(\frac{M_\infty}{Re}\right)\frac{\tilde{\nu}^*}{\kappa^2 d^{2*}} f_{v2}, \tag{3.59}$$

$$D = \left\{ \begin{array}{cc} \left(\frac{M_\infty}{Re}\right)\rho^*\left[c_{w1}f_w - \frac{c_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}^*}{d^*}\right)^2 & for \quad \tilde{\nu}^* > 0 \\ -\left(\frac{M_\infty}{Re}\right)\rho^* c_{w1}\left(\frac{\tilde{\nu}^*}{d^*}\right)^2 & for \quad \tilde{\nu}^* \leq 0 \end{array} \right\}, \tag{3.60}$$

$$r = min\left(\frac{M_\infty}{Re}\frac{\tilde{\nu}^*}{\tilde{S}^*\kappa^2 d^{2*}}, 10\right). \tag{3.61}$$

The compressible RANS equations can be non-dimensionalized using the same methodology applied to NS equations. Including the turbulence transport equation (3.57), the complete system of transport equations in non-dimensional form is obtained. From this point on, asterisks will not be used to indicate dimensionless parameters for clarity. The conservation equation that includes mass, momentum, energy and turbulence equations in integral form is shown below.

$$\frac{\partial}{\partial t}\int\limits_V \overrightarrow{Q}\,dV + \int\limits_S (\overrightarrow{F} - \overrightarrow{G})dS = \int\limits_V \overrightarrow{S}\,dV \tag{3.62}$$

where Q is the conserved variable vector, and F is the inviscid flux vector. G is the viscous flux vector, and S is the source vector.

$$\overrightarrow{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \\ \rho\tilde{\nu} \end{bmatrix} \quad \overrightarrow{F} = \begin{bmatrix} \rho u \boldsymbol{n_x} + \rho v \boldsymbol{n_y} \\ (\rho u^2 + p)\boldsymbol{n_x} + \rho uv \boldsymbol{n_y} \\ \rho uv \boldsymbol{n_x} + (\rho u^2 + p)\boldsymbol{n_y} \\ \rho uH \boldsymbol{n_x} + \rho vH \boldsymbol{n_y} \\ \rho u\tilde{\nu} \boldsymbol{n_x} + \rho v\tilde{\nu} \boldsymbol{n_y} \end{bmatrix} \tag{3.63}$$

$$\overrightarrow{G} = \begin{bmatrix} 0 \\ \tau_{xx}\boldsymbol{n_x} + \tau_{xy}\boldsymbol{n_y} \\ \tau_{yx}\boldsymbol{n_x} + \tau_{yy}\boldsymbol{n_y} \\ (u\tau_{xx} + v\tau_{xy} - q_x)\boldsymbol{n_x} + (u\tau_{xy} + v\tau_{yy} - q_y)\boldsymbol{n_y} \\ \left[\frac{1}{\sigma}\rho\left(\nu + \tilde{\nu}\right)\frac{\partial\tilde{\nu}}{\partial x}\right]\boldsymbol{n_x} + \left[\frac{1}{\sigma}\rho\left(\nu + \tilde{\nu}\right)\frac{\partial\tilde{\nu}}{\partial y}\right]\boldsymbol{n_y} \end{bmatrix} \tag{3.64}$$

$$\overrightarrow{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ S_t \end{bmatrix} \tag{3.65}$$

where turbulence source term equals to

$$S_t = P - D + \left(\frac{M_\infty}{Re}\right)\frac{1}{\sigma}\left(\rho c_{b2}\frac{\partial\tilde{\nu}^*}{\partial x_i^*}\frac{\partial\tilde{\nu}^*}{\partial x_i^*}\right) - \left(\frac{M_\infty}{Re}\right)(\nu+\tilde{\nu})\frac{\partial\rho}{\partial x_i}\frac{\partial\tilde{\nu}}{\partial x_i}. \tag{3.66}$$

The final form of shear stress terms are

$$\tau_{xx} = \mu_{eff}\frac{M_\infty}{Re}(\frac{4}{3}\frac{\partial u}{\partial x} - \frac{2}{3}\frac{\partial v}{\partial y}), \tag{3.67}$$

$$\tau_{yy} = \mu_{eff}\frac{M_\infty}{Re}(\frac{4}{3}\frac{\partial v}{\partial y} - \frac{2}{3}\frac{\partial u}{\partial x}), \tag{3.68}$$

$$\tau_{yx} = \tau_{xy} = \mu_{eff}\frac{M_\infty}{Re}(\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}). \tag{3.69}$$

and heat fluxes are defined as

$$q_x = -\frac{1}{\gamma-1}\frac{M_\infty}{Re}\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\frac{\partial\left(p/\rho\right)}{\partial x}, \tag{3.70}$$

$$q_y = -\frac{1}{\gamma-1}\frac{M_\infty}{Re}\left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\frac{\partial\left(p/\rho\right)}{\partial y}. \tag{3.71}$$

3.4 Boundary Conditions

In the solution of external flows, two types of boundary conditions are used. These are far-field and wall boundary conditions. Numerical implementation of these boundary

conditions is described in the following sections, including the treatment of turbulence variable $\tilde{\nu}$.

3.4.1 Far-field and Wall Boundary conditions

The far-field boundary is presented when an out-type grid has no neighbors at its face. It indicates that the grid is at the edge of the far-field boundary. For test cases, the domain size is selected larger than at least 200 times the chord of selected airfoils. Thus, vortex correction for lifting bodies becomes nonessential. Fluxes are computed using ghost cells of the same size as boundary cells at the outer boundaries. Free-stream values are attained to these ghost cells, and the same flux function is applied. Figure 3.1 shows a representation of the far-field boundary procedure.

$$(V_n)_{ghost} = M_\infty cos\alpha \quad (V_t)_{ghost} = M_\infty sin\alpha \tag{3.72}$$

$$p_{ghost} = p_\infty \quad \rho_{ghost} = \rho_\infty \quad E_{ghost} = E_\infty \tag{3.73}$$

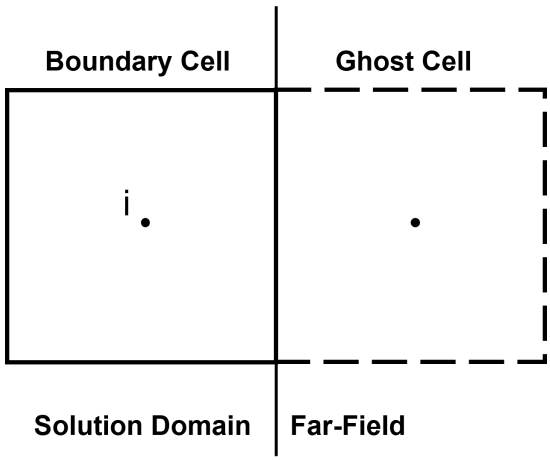


Figure 3.1: Far-field boundary condition

The second one is the wall boundary condition. At the wall boundaries body, orthogonal an-isotropic quad cells are used. If the cell type is quad and one of the faces has no neighbors wall boundary procedure is applied. Again, the cell size of the ghost cell is equated to the size of the real cell. No-slip boundary conditions are applied by

equating the tangential velocity parameter at the face to zero. Similarly, the velocity component normal to the wall is equated to zero since there is no mass flux into the wall.

$$(V_n)_{ghost} = -(V_n)_i \tag{3.74}$$

$$(V_t)_{ghost} = -(V_t)_i \tag{3.75}$$

The wall boundary procedure is completed by setting zero pressure gradient and assuming zero heat flux, defining

$$T_{ghost} = T_i,\ p_{ghost} = p_i,\ \rho_{ghost} = \rho_i. \tag{3.76}$$

Figure 3.2 shows a representation of the wall boundary procedure.

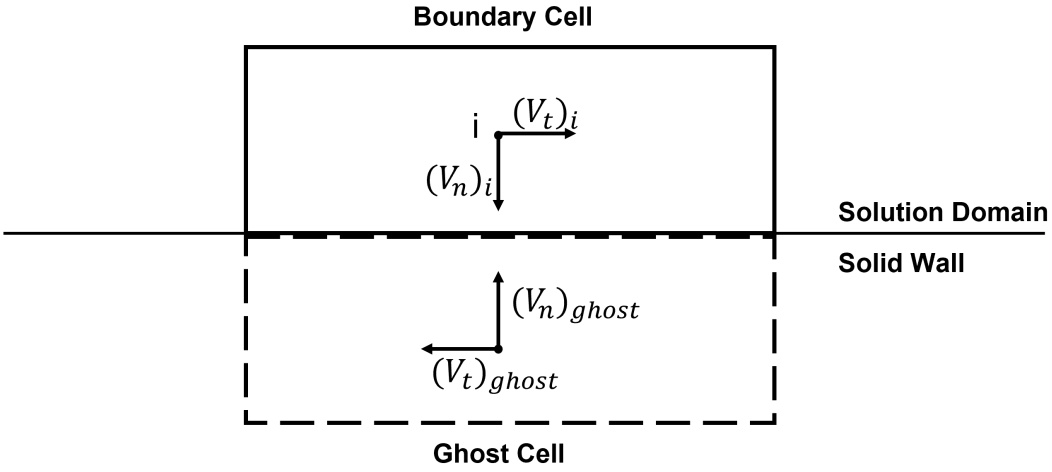


Figure 3.2: Wall boundary condition

3.4.2 Turbulent Boundary Conditions

Two different boundary conditions attained to turbulence variable $\tilde{\nu}$. The first one is the wall boundary condition. The turbulence variable is equated to zero at the wall boundaries.

$$\tilde{\nu}_{wall} = 0 \tag{3.77}$$

The second one is the far-field boundary condition. It is defined differently from the given conditions in the original paper [5]. Since the model is applied to solve fully turbulent problems, a more suitable range is used following the suggestion in [33].

$$\tilde{\nu}_{farfield} = 3\nu_{\infty}\, to\, 5\nu_{\infty} \tag{3.78}$$

$$\nu_{t,farfield} = 0.210438\nu_{\infty}\, to\, 1.294234\nu_{\infty} \tag{3.79}$$

CHAPTER 4

NUMERICAL APPROACH

The primary aim of this study is to solve the compressible RANS equations with Spalart Allmaras turbulent model presented in Chapter 3. A cell-based numerical algorithm has been developed to achieve this goal. The solution domain is discretized with Cartesian grids at outer regions, while body orthogonal quadrilateral grids are used to discretize near-body regions, as it is indicated in Chapter 2. Discretization methods are based on the cell-centered finite volume method. This chapter explains numerical approximations used for governing equations.

The governing equations are described in three primary steps. These are reconstruction, flux calculation, and time discretization. As the first step, cell averaged values are reconstructed linearly at the faces of cells using a limiter. Then using reconstructed values, inviscid and viscous flux methods are applied. Also, flux and source terms of the Spalart Allmaras equation are included in the solution. Lastly, a multi-stage time stepping scheme is used to proceed the solution to a steady state. A solution adaptive mesh refinement procedure is also implemented to have a higher convergence rate and obtain a grid-independent solution.

4.1 Solution Reconstruction

In the cell-centered finite volume approaches, average values of variables are stored at cell centers. These values are assumed as constant through the volume of each cell. Thus, there is no direct information about the distribution of these variables from one cell to another. This information is required to compute face fluxes and source terms that use the change of variables according to the change of position in the control

volume. Therefore, solution reconstruction is essentially required. There are various approaches for reconstructing the solution. One of them is the Green-Gauss approach which approximates the gradient of some scalar function as its path integral over some control volume. The other one is the Least-squares approach that is selected in this study.

4.1.1 Least-squares Approach

This approach is firstly introduced by Barth [34] for a median-dual scheme. The application is very similar for both median-dual and cell-centered schemes. An example of cell-centered application can be found in Wang [9]. By using cell-centered variables of neighbor cells, the gradient of primitive variables at cell centers can be approximated as

$$q(x,y) = q_{cell} + q_x(x - x_{cell}) + q_y(y - y_{cell}). \tag{4.1}$$

where q_{cell} is the value of primitive variable at cell center and x_{cell}, y_{cell} are its location for a 2 dimensional domain. Then the change of the variables q_x, q_y with respect to x and y can be computed by using,

$$q_x = \frac{1}{\Delta}\left[I_{yy}\sum_{i=1}^{n} w_i^2\left(q_i - q_c\right)\left(x_i - x_c\right) - I_{xy}\sum_{i=1}^{n} w_i^2\left(q_i - q_c\right)\left(y_i - y_c\right)\right], \tag{4.2}$$

$$q_y = \frac{1}{\Delta}\left[-I_{xy}\sum_{i=1}^{n} w_i^2\left(q_i - q_c\right)\left(y_i - y_c\right) + I_{xx}\sum_{i=1}^{n} w_i^2\left(q_i - q_c\right)\left(x_i - x_c\right)\right], \tag{4.3}$$

$$I_{xx} = \sum_{i=1}^{n} w_i^2\left(x_i - x_c\right)^2, \tag{4.4}$$

$$I_{xy} = \sum_{i=1}^{n} w_i^2\left(x_i - x_c\right)\left(y_i - y_c\right), \tag{4.5}$$

$$I_{yy} = \sum_{i=1}^{n} w_i^2 \left(y_i - y_c\right)^2, \tag{4.6}$$

$$\Delta = I_{xx} I_{yy} - I_{xy}^2. \tag{4.7}$$

where n is the total number of neighbor and corner cells, and w_i is the weight function. For the calculations in this thesis, the weights were set to one for particular test cases. For other test cases, the weight function defined by Shima et al. [35] is used. They proposed the function for a weighted least-squares (WLSQ) approach inherited from the Green-Gauss method. The function is defined as

$$w_i^2 = \left(\frac{2\bar{l}_i}{\overline{L}_i}\right)^2 \frac{s_i}{L_i}. \tag{4.8}$$

where L_i is the distance between cell centers, s_i is face length, $\overline{L}_i$ and $\bar{l}_i$ are projections of cell center distance L_i, the distance between the cell center and face center to the cell face normal, respectively.

Figure 4.1 illustrates geometric variables used in function (4.8).

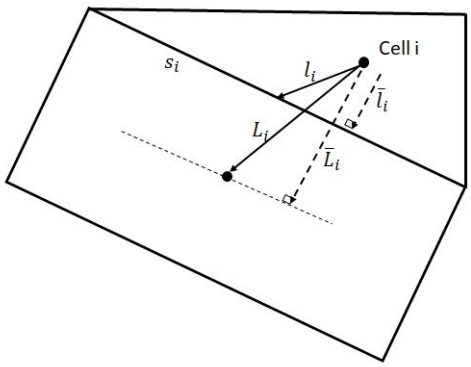


Figure 4.1: Schematic of cell geometrical values used in the weight function

In the code, I_{xx}, I_{xy}, and I_{yy} are precalculated to decrease computational cost.

4.1.2 Gradient Limiting

The implemented reconstruction procedure in this study has no inherent measure to guarantee that the result of reconstruction bounded within the range of support data used. Thus, an additional limiter is used [36] to ensure the monotonicity criterion. The limiter limits the maximum and minimum results of the performed solution so that it does not exceed the values at the center of support cells used in the reconstruction. The reconstruction function with limiter is defined as

$$q(x,y) = q_{cell} + \mathbf{\Phi}\left[q_x(x - x_{cell}) + q_y(y - y_{cell})\right]. \tag{4.9}$$

where $\mathbf{\Phi}$ is the limiter. It is calculated for each primitive parameter and each vertex of the cell. Then, the smallest value is selected and used in the reconstruction function (4.9). Firstly, the maximum and minimum values of the primitive parameters among values at supporting cell centers and the cell center where reconstruction is applied are computed.

$$q_{max} = (q_{cell}, q_n^i), \quad q_{min} = (q_{cell}, q_n^i), \quad for\, i = 1, .., Number\, of\, Supports. \tag{4.10}$$

The reconstructed value should be within the range of maximum and minimum valued parameters.

$$q_{min} \leq q(x,y) \leq q_{max} \tag{4.11}$$

For a vertex k of the cell, where reconstruction is applied with no limiter if the reconstructed value is larger than the value at the cell center q_{max} is used to calculate the limiter. Otherwise, q_{min} is used. Calculation of limiter is defined as follows:

$$\Phi_k = \left\{ \begin{array}{ccc} min\left(1, \frac{q_{max} - q_{cell}}{q_k - q_{cell}}\right) & if & q_k - q_{cell} > 0 \\ 1 & if & q_k - q_{cell} = 0 \\ min\left(1, \frac{q_{min} - q_{cell}}{q_k - q_{cell}}\right) & if & q_k - q_{cell} < 0 \end{array} \right\}. \tag{4.12}$$

Finally, the minimum value is selected for the limiter that is going to be used in reconstruction,

$$\mathbf{\Phi} = min\left(\Phi_1, \Phi_2, ..\Phi_k\right). \tag{4.13}$$

Although implemented limiter provides stability in numerical solutions, it delays convergence and makes it difficult to reach steady state solutions in some cases [37]. Venkatakrishnan presented a modified limiter [38] based on Barth's limiter by addressing this problem. In this study, this modification is also applied and tested.

The modified limiter is defined as

$$\Phi_k = \frac{1}{\triangle_-}\left[\frac{(\triangle_+^2 + \epsilon^2)\triangle_- + 2\triangle_-^2\triangle_+}{\triangle_+^2 + 2\triangle_-^2 + \triangle_+\triangle_- + \epsilon^2}\right]. \tag{4.14}$$

where $\triangle_- = q_k - q_{cell}$ and $\triangle_+$ is defined as

$$\triangle_+ = \left\{ \begin{array}{lll} q_{max} - q_{cell} & if & \triangle_- > 0 \\ q_{min} - q_{cell} & if & \triangle_- < 0 \end{array} \right\}. \tag{4.15}$$

Epsilon is defined by following the work of Wang [10]. He addressed that with the significant variations of the cell size across the domain, the original definition may become problematic for neighboring cells with different levels. Since Cartesian cells can also have significant size differences, an ϵ that does not depend on the mesh size is defined and used in this study.

$$\epsilon = 0.05(q^{max} - q^{min}) \tag{4.16}$$

4.2 Spatial Discretization

In the cell-centered finite volume method, the discrete form of the equation (3.62) becomes

$$A\frac{\partial \overrightarrow{Q}}{\partial t} + \sum_{faces}(\overrightarrow{F} - \overrightarrow{G})\triangle s - A\overrightarrow{S} = 0. \tag{4.17}$$

where A is the area and L is the length of the corresponding cell. The viscous and inviscid fluxes are calculated as the summation of them at the faces of the cell. Inviscid flux is formulated by using an upwind scheme, the Roe's flux difference splitting [39], while viscous flux is formulated with averaged cell variables and their gradients at the cell interfaces.

4.2.1 Inviscid Flux Formulation

Roe's flux difference Splitting is one of the most popular approximate Riemann solvers. In Roe's approach, the Jacobian matrix is replaced with a constant one to approximate the Riemann problem. It makes the original non-linear conservation laws a linearized system of equations with constant coefficients. Details of approximations can be found in [40]. The resulting intercell flux is

$$F_{i+\frac{1}{2}} = \frac{1}{2}(F_L + F_R) - \frac{1}{2}\sum_{i=1}^{m} \tilde{\alpha_i} \mid \tilde{\lambda}_i \mid \tilde{K}^{(i)}. \tag{4.18}$$

To find this flux for an m numbered hyperbolic system of equations, the wave strengths $\tilde{\alpha_i}$, the eigenvectors $\tilde{\lambda}_i$ and also the right eigenvectors,$\tilde{K_i}$, need to be calculated. For the two-dimensional Euler equations, these matrices are defined as

$$\mid \tilde{\lambda}_i \mid = \begin{bmatrix} U_{RL} - C_{RL} \\ U_{RL} \\ U_{RL} \\ U_{RL} + C_{RL} \end{bmatrix}, \tag{4.19}$$

$$\tilde{K_i} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ U_{RL} - C_{RL} & U_{RL} & 0 & U_{RL} + C_{RL} \\ V_{RL} & V_{RL} & 1 & V_{RL} \\ H_{RL} - U_{RL}C_{RL} & \frac{U_{RL}^2 + V_{RL}^2}{2} & V_{RL} & H_{RL} + U_{RL}C_{RL} \end{bmatrix}, \tag{4.20}$$

$$\tilde{\alpha_i} = \begin{bmatrix} \frac{\triangle P - \rho_{RL} C_{RL} \triangle u}{2C_{RL}^2} \\ \triangle \rho - \frac{\triangle P}{C_{RL}^2} \\ \rho_{RL} \triangle v \\ \frac{\triangle P + \rho_{RL} C_{RL} \triangle u}{2C_{RL}^2} \end{bmatrix}. \tag{4.21}$$

The intermediate variables used are

$$\rho_{RL} = \sqrt{\rho_L \rho_R}, \tag{4.22}$$

$$U_{RL} = u_R + \frac{\sqrt{\rho_L}\,(u_L - u_R)}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{4.23}$$

$$V_{RL} = v_R + \frac{\sqrt{\rho_L}\,(v_L - v_R)}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{4.24}$$

$$H_{RL} = H_R + \frac{\sqrt{\rho_L}\,(H_L - H_R)}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \tag{4.25}$$

$$C_{RL} = \left((\gamma - 1)\left(H_{RL} - 0.5\left(U_{RL}^2 + V_{RL}^2\right)\right)\right)^{1/2}. \tag{4.26}$$

Corresponding left and right velocities are normal and tangential to the cell face direction. For instance, right cell velocities are computed by using the following operations,

$$u_R = u cos\theta + v sin\theta, \tag{4.27}$$

$$v_R = v cos\theta - u sin\theta, \tag{4.28}$$

where the angle θ is the face normal angle.

4.2.2 Viscous Flux Formulation

Flow quantities and their gradients need to be known at cell faces to calculate viscous flux at cell faces. While flow quantities reconstructed at the left and right cell centers can be averaged to compute the quantity at the cell face, the same procedure may create a numerical problem for the gradients. Because the contribution from the further faces becomes more significant than the contribution of the neighbor cells, it may cause numerical instability. Wang [9] address this problem and suggests the following reconstruction approach for viscous flux to have more stable and accurate results.

Let k and l be the cell face's tangential and normal unit vectors, respectively. The derivative of a primitive variable q at face with respect to k direction is

$$\frac{dq}{dk} = \frac{1}{2}\left(\nabla f(q_L)\cdot k + \nabla f(q_R)\cdot k\right). \tag{4.29}$$

where gradients from left and right cells have already been calculated for inviscid reconstruction. The gradient in the normal direction is calculated from

$$\frac{dq}{dl} = \frac{q_R - q_L}{abs(r_R - r_L)}. \tag{4.30}$$

By using these two equations, face gradients in x and y directions can be calculated from

$$q_x \cdot l_x + q_y \cdot l_y = \frac{dq}{dl}, \tag{4.31}$$

$$q_x \cdot k_x + q_y \cdot k_y = \frac{dq}{dk}. \tag{4.32}$$

4.2.3 Discretization of the SA Equation

Like other transport equations, the SA equation is discretized using a cell-centered, finite-volume method. The turbulence equation's convective part is calculated by using the first-order scalar upwind discretization. In this manner, the calculation of

this part is uncoupled from the remaining set of convective fluxes. The diffusion term is computed with the same procedure implemented for the rest of the viscous fluxes. Source term is evaluated at cell centers by using cell values of flow parameters.

4.3 Temporal Discretization

The discrete form of the equation (3.62) is defined as

$$A\frac{\partial \overrightarrow{Q}}{\partial t} + \sum_{faces}\left(\overrightarrow{F} - \overrightarrow{G}\right)\triangle s - A\overrightarrow{S} = 0. \tag{4.33}$$

Calling all terms other than time derivative of conserved variables R,

$$A\frac{\partial Q_i}{\partial t} + R\left(Q\right) = 0 \tag{4.34}$$

For steady-state solutions, the R term known as the residual goes to zero.

$$R\left(Q\right) = 0 \tag{4.35}$$

Applying forward finite difference discretization to the time derivative term to obtain an explicit solution scheme,

$$A\frac{Q_i^{n+1} - Q_i^n}{\triangle t} + R^n\left(Q\right) = 0 \tag{4.36}$$

$$Q_i^{n+1} = Q_i^n - \frac{\triangle t}{A}R^n\left(Q\right) \tag{4.37}$$

Superscript n stands for the time step that the solution is known. An optimally smoothing multi-stage scheme [41] is applied to advance the solution to $n + 1^{th}$ time step.

4.3.1 Multi-Stage Time Advancing

The general definition of the time scheme is defined as

$$Q_i^0 = Q_i^n,$$

$$Q_i^k = Q_i^0 - C\frac{\alpha_k \triangle t}{A} R\left(Q^{k-1}\right) \quad for \quad k = 1, 2..., m, \tag{4.38}$$

$$Q_i^{n+1} = Q_l.$$

where C is the Courant-Friedrichs Lewy number, CFL, and α_k is the coefficient for each stage. In this study, the number of stages is selected as three. In (4.1) multi-stage coefficients and CFL numbers used for first and second-order spatial schemes are listed.

	First-Order Solution	**Second-Order Solution**
C	1.5000	0.6936
α_1	0.1481	0.1918
α_2	0.4000	0.4929
α_3	1.0000	1.0000

Table 4.1: Multi-stage coefficients for First and Second order discretization

This research aims to obtain steady-state solutions for turbulent flows. Thus, local time-stepping is applied. It is necessary since cell sizes can significantly vary due to cut and split cells. The local time step is calculated by considering the convective and diffusive characteristics of the Navier-Stokes equations [42] with the following equations.

$$\triangle t = \frac{\triangle t_c \triangle t_v}{\triangle t_c + \triangle t_v} \tag{4.39}$$

and convective and viscous time steps are defined as

$$\triangle t_v = K_v \frac{A}{\lambda_v}, \tag{4.40}$$

$$\triangle t_c = \frac{A}{\varphi_x + \varphi_y}. \tag{4.41}$$

where K_v is an empirical constant and equals 0.25. Other terms are defined as

$$\varphi_x = \frac{|u| + c}{2} \sum_{faces} |S_x|, \quad \varphi_y = \frac{|u| + c}{2} \sum_{faces} |S_y|, \tag{4.42}$$

$$\lambda_v = \frac{M_\infty \gamma}{RePrA} \sum_{faces} \frac{\mu}{\rho} \triangle s^2. \tag{4.43}$$

4.4 Wall Distance

SA model requires calculating the smallest distance between the wall boundary and each cell. A small error in the wall distance calculation can result in a significant error at the end of the solution [43]. Thus, the shortest distance to any point on the wall boundary should be calculated instead of using the nearest mesh vertices or line center on the wall boundary. In this study, wall distance is computed by searching every line segment constructed by successive wall points using the brute force approach. For each point nearest distance is calculated for every line, and the minimum one is stored as a pointer. Brute force is a direct approach that may cause a problematic computational cost. However, it has an insignificant calculation duration for most applications since two-dimensional cases are studied in this research.

4.5 Solution Adaptive Mesh Refinement

One of the most desired features of Cartesian grids is that they provide ease in refinement and coarsening procedure. This makes the use of solution adaption in certain intervals quite attractive. After a specific number of iterations in solution, grids can be easily refined or coarsened, considering relative flow characteristics. For instance, shock locations, contact surfaces, and other high gradient locations can be flagged according to selected criteria. Then grids can be refined locally to have more accurate solutions in these areas. Selecting more proper criteria for studied flow characteristics

is essential to use computational power more efficiently and have accurate solutions. There are various adaptation criteria in the literature. A detailed study about them was conducted by [44]. Their study showed that using the curl and the divergence together gives one of the best results for finding shear layers and shocks. Also, both criteria are direction-independent.

Divergence and curl are computed for each cell, including length scales of related cells. Calculation of both terms with the addition of length scales is defined as,

$$\tau_c = \mid \nabla \times V \mid L^{\frac{r+1}{r}},\ \tau_d = \mid \nabla . V \mid L^{\frac{r+1}{r}} \tag{4.44}$$

where $r = 2$. The standard deviations about zero are computed for each parameter to set the criterion,

$$\sigma_c = \sqrt{\frac{\sum_{i=1}^{n} \tau_{ci}^2}{n}},\ \sigma_d = \sqrt{\frac{\sum_{i=1}^{n} \tau_{di}^2}{n}} \tag{4.45}$$

If one of the following conditions is met for a cell, the cell is refined.

$$\tau_c \geq \sigma_c,\ \tau_d \geq \sigma_d \tag{4.46}$$

The refinement procedure is simply taken for Cartesian grids by generating new children grids. After Cartesian grids are refined, near-body quadrilateral grids are regenerated with the process mentioned in Chapter 2.

CHAPTER 5

RESULTS AND DISCUSSION

In this section, the accuracy and efficiency of the solver are tested for different problems. The first problem is turbulent flow over a zero-pressure flat plate. The second one is a two-dimensional bump problem. Lastly, turbulent flow over the NACA0012 airfoil is solved. Different levels of adaptive refinement are applied for all three problems, and results are compared with reference results. The code has been run on a personal computer with Intel Core i7-11800H CPU @ 2.30GHz and 16 GB RAM.

5.1 Two-Dimensional Zero-Pressure Flat Plate

Two-dimensional turbulent flow over a flat plate is a well-known case and is suitable for determining the accuracy of the turbulence model implementation in the code. In this problem, the flow Mach number is selected as 0.2, free stream temperature is $273.15K$, and Reynolds Number is 5×10^6, while the reference length is chosen as one. The outer boundary size factor is selected as 18 to ensure far-field boundaries are far enough and does not affect the solution accuracy. The initial condition for turbulent eddy viscosity is selected as $\tilde{\nu} = 3.0\nu$ to provide a fully turbulent region from the start. The solver is iterated until the logarithm of root mean square of normalized continuity residual reaches -8. The following plot 5.1 illustrates the layout of the solution domain with the boundary conditions.

The results are obtained by using five different cases. In all of the cases, the first-order scheme is used. Two different initial grids are used to observe the sensitivity of the

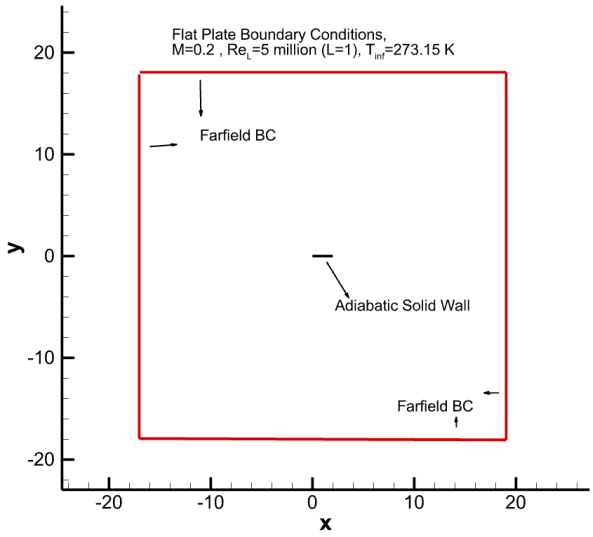


Figure 5.1: Solution domain with boundary conditions for flat plate problem

turbulence model to minimum wall-normal spacing and boundary layer resolution. For the first three cases, the first layer thickness is chosen as 1.3×10^{-5} times the plate length. The stretch factor is 1.15, and 35 layers of the body-conforming quadrilateral grid are generated. For the fourth and fifth cases, the first layer thickness is chosen as 1×10^{-6} times the plate length. The stretch factor is 1.15, and 58 layers of the body-conforming quadrilateral grid are generated.

The other changed parameter for the cases is the refinement level. Solution adaptation is used up to two refinement levels to show improvement in accuracy. In figure 5.2, the mesh configuration at the leading edge for Case 4 and Case 5 is presented. Red lines show the generated grids after two levels of solution refinement. In figure 5.3, the two different discretizations of the boundary layer are represented. 5.3a shows the initial boundary layer resolution used in Cases 1, 2, and 3. 5.3b shows the initial boundary layer resolution used in Cases 4 and 5. As can be observed, finer boundary layer resolution is applied for the last two cases.

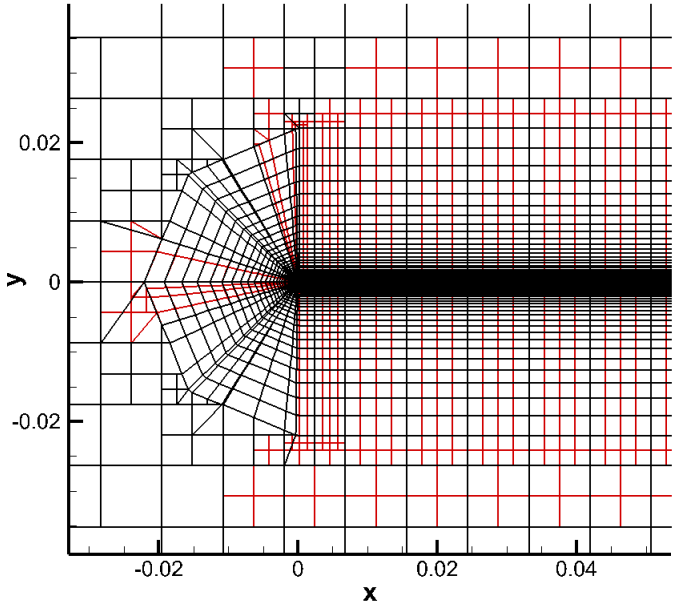
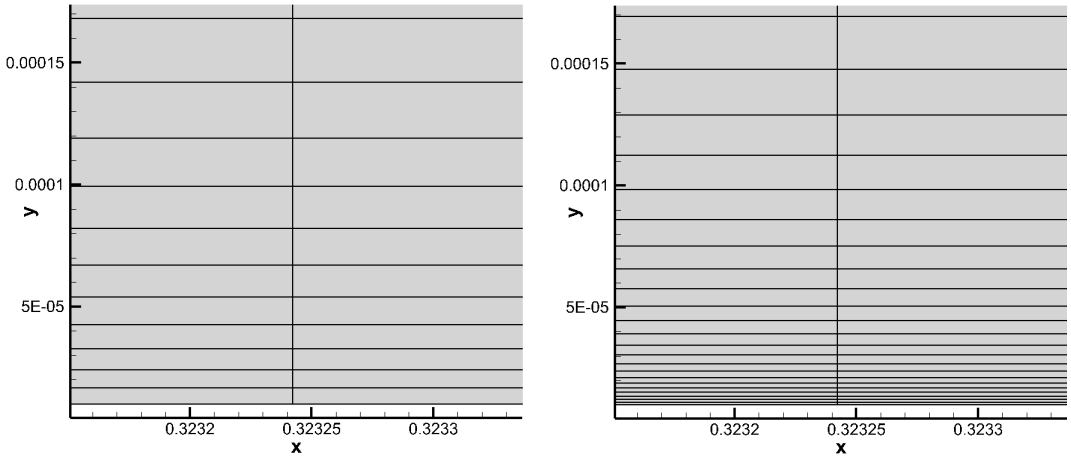


Figure 5.2: Three levels of solution adaptive refinement for flat plate problem



(a) (b)

Figure 5.3: Comparison of two different boundary layer discretization

Since the leading and trailing edges of the flat plate have pointed convex corners, the first layer of cells appears as triangles at that location. The flat plate is parallel to the y-axis, quad cells have a good alignment with cut cells, and thus most of the cut cells remained quadrilateral. Table 5.1 lists the number of grids used and computational time for all of the test cases.

Table 5.1: Number of cells and computational time of cases for turbulent flow over a flat plate

Case No.	Description	Number of Cells	Computational Time
1	No Refinement with thicker first layer	20360	5 hours 4 minutes 55 seconds
2	One Refinement with thicker first layer	37652	8 hours 32 minutes 52 seconds
3	Two Refinement with thicker first layer	74444	23 hours 30 minutes 40 seconds
4	No Refinement with thinner first layer	32952	10 hours 58 minutes 26 seconds
5	Two Refinement with thinner first layer	118121	33 hours 53 minutes 24 seconds

The skin friction coefficient for the flat plate problem is compared with two different theories. The first one is Prandtl's power-law approximation. It is defined as

$$C_f \approx \frac{0.058}{Re_x^{1/5}}. \tag{5.1}$$

The second one is White's approximation for turbulent flow over a flat plate by using Spalding's wall formula and is defined as

$$C_f \approx \frac{0.455}{ln^2(0.06Re_x)}. \tag{5.2}$$

Figures 5.4 and 5.5 show the skin friction coefficient along the chord. Results are compared with mentioned theories. Five percentage error bars are included for both theory lines. For both grid structures, accuracy is increased for increased refinement levels. Results from the fifth case underestimated White's approximation with approximately five percentage differences excluding the trailing edge. Case 3 showed

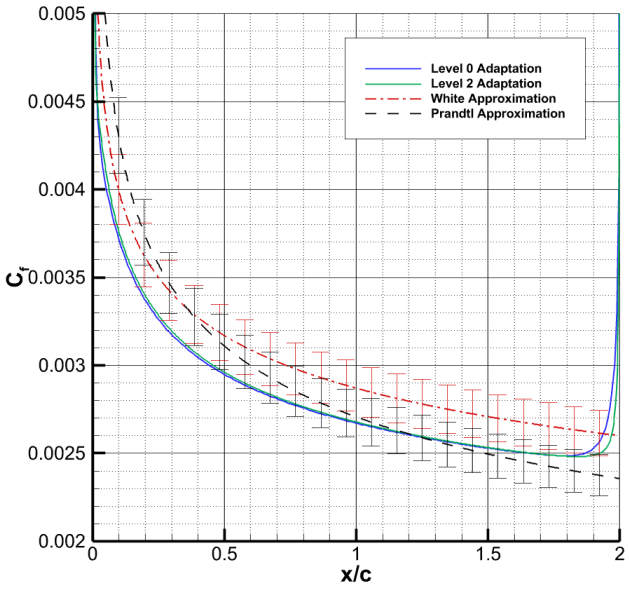


Figure 5.5: Skin friction coefficient distribution along the flat plate for the last two cases with theoretical lines

similar results to Case 5. However, by the end of the flat plate, the percentage of underestimation is increased.

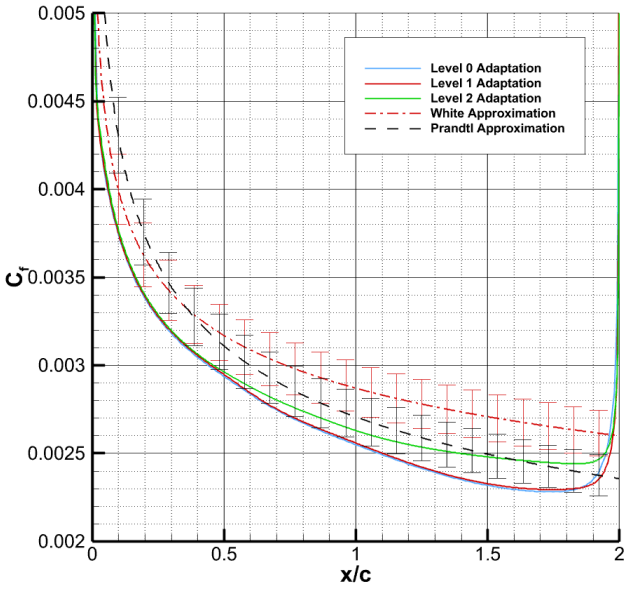


Figure 5.4: Skin friction coefficient distribution along the flat plate for the first three cases with theoretical lines

Results of the test cases are also compared with CFL3D results [45] as they are shown in Figures 5.7 and5.8. The best agreement is achieved in Case 5, where the finer resolved boundary layer grid and two levels of adaptation are used. Although solution accuracy is improved in the case of the coarser resolved boundary layer grid with an increasing level of solution adaptation, there is still more difference with the CFL3D result than the finer resolved grid with no solution adaptation.

Plot 5.6 shows y^+ along the surface of cases 3 and 5. While y^+ is lower than one along the plate in Case 5, it is higher than one in Case 3 at the beginning and end of the plate.

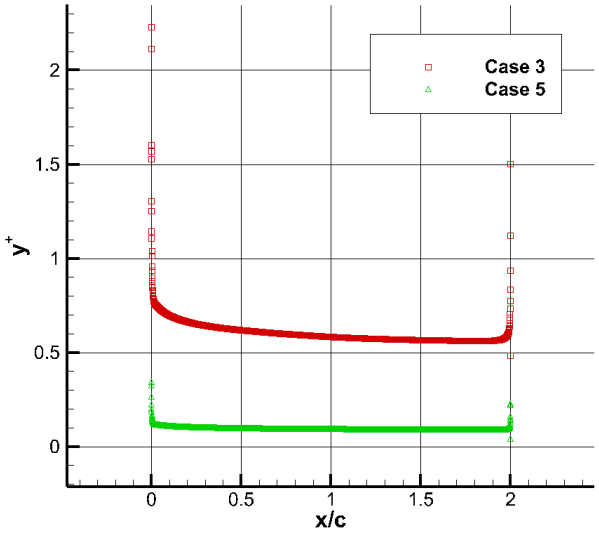


Figure 5.6: Minimum y^+ values along the flat plate

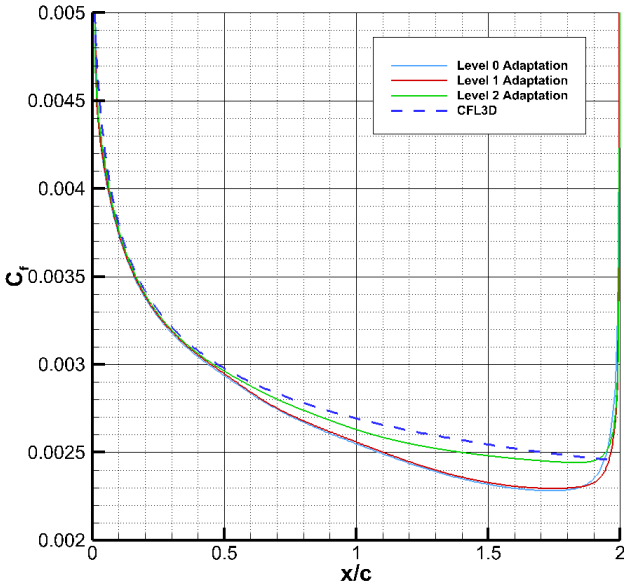


Figure 5.7: Skin friction coefficient distribution along the flat plate for Case 1, 2, and 3 with CFL3D results

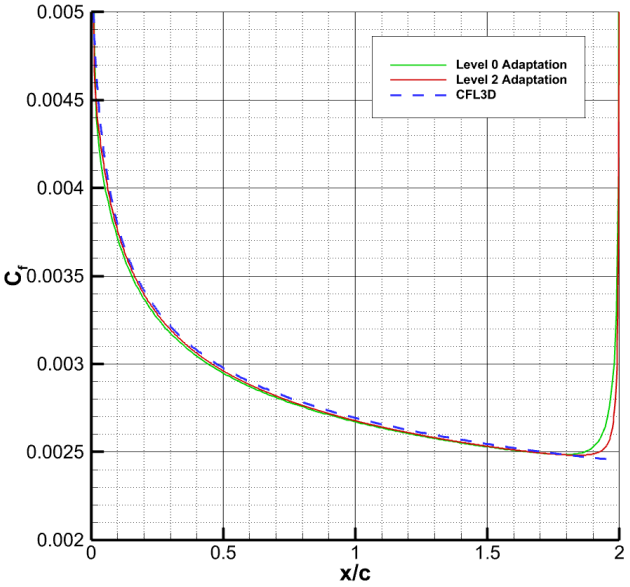


Figure 5.8: Skin friction coefficient distribution along the flat plate for Case 4 and 5 with CFL3D results

Results of Case 3 and Case 5 are also compared with the law of wall at $x = 0.97$ in the figure 5.9. The law of wall is defined as follows

$$u^+ = y^+ \quad for \quad y^+ < 5, \quad u^+ = \frac{1}{\kappa} ln y^+ + C^+ \quad for \quad y^+ > 30. \tag{5.3}$$

At the log layer, both results have good accuracy. Case 3 has inaccurate results at the viscous sublayer, while Case 5 has a better match as it is shown in Figure 5.10. Results show that the best accuracy is obtained in Case 5, where a thinner first layer and a larger number of boundary layer mesh are used. In all cases, inaccuracies are observed at the end of the plate.

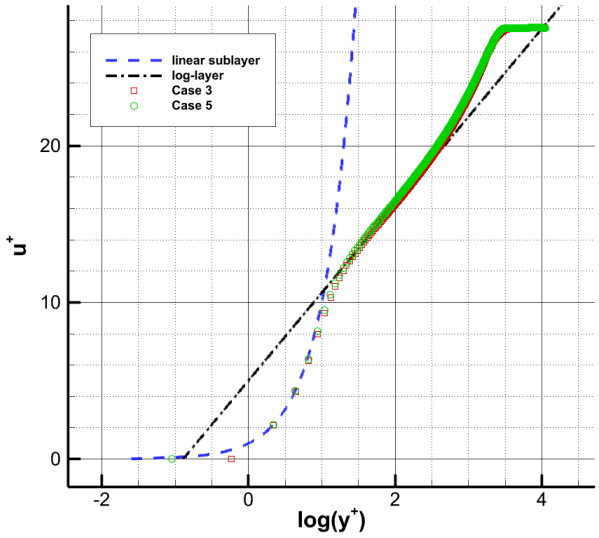


Figure 5.9: Inner wall variables at $x = 0.97$ with theoretical low-law curves

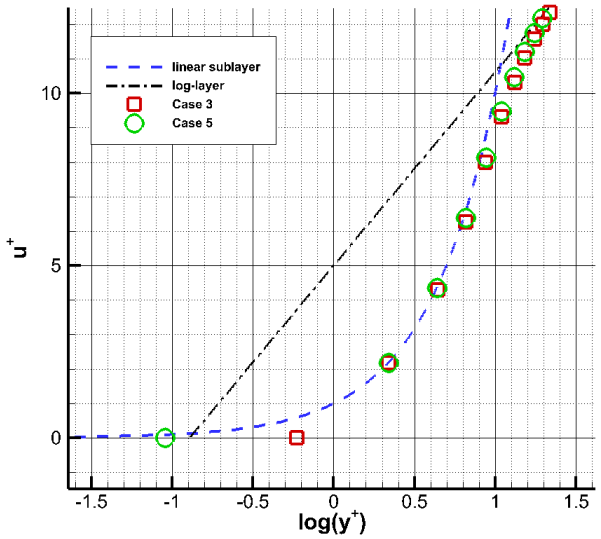


Figure 5.10: Inner wall variables at $x = 0.97$ in the linear sub-layer region

Figure 5.11 presents the Mach contour at the leading edge of the flat plate for Case 5. Also, velocity vectors at different locations are presented in the same figure to show the development of the turbulent boundary layer.

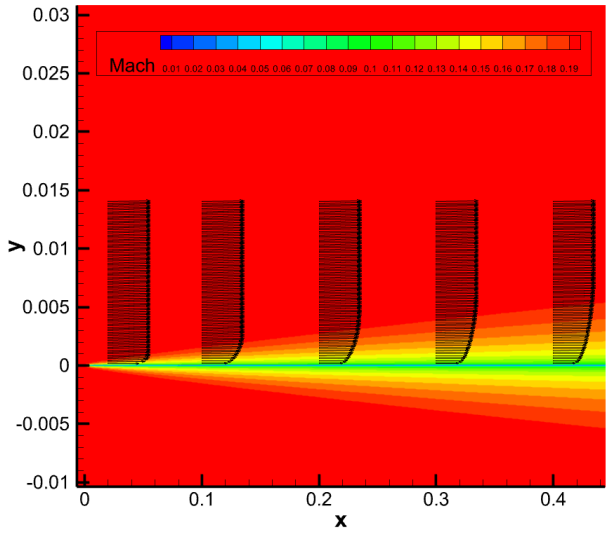


Figure 5.11: Mach contour for turbulent flow over a flat plate

5.2 Two-Dimensional Bump

The flat plate problem is the first step in testing the implementation of the turbulence model. As the next step, the two-dimensional bump problem is selected. In this problem, a pressure gradient is present because of curvature in the middle of the geometry. The free-stream Mach number for this test case is selected as $M_\infty = 0.2$, while the Reynolds number is $Re = 3 \times 10^6$. The initial condition for turbulent eddy viscosity is selected as $\tilde{\nu} = 3.0\nu$ to provide a fully turbulent region from the start. The solver is iterated until the logarithm of root mean square of normalized continuity residual reaches -8. Plot 5.12 shows the layout of the solution domain with the boundary conditions.

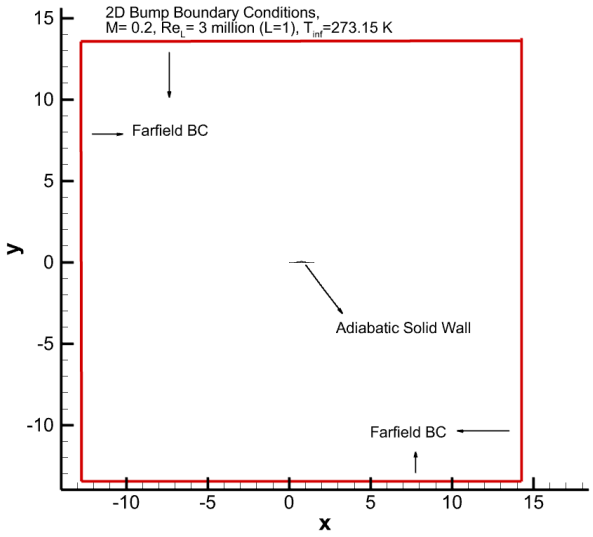


Figure 5.12: Solution domain with boundary conditions for two-dimensional bump problem

Five test cases are conducted to compare the accuracy of first and second-order reconstruction. The exact initial grid is used in all cases. In the first three cases, first-order reconstruction is used with solution refinement up to level 2. Second-order reconstruction is applied by using WLSQ and Venkatakrishnan's limiter for the last two cases with solution refinement up to level one. The results are compared with the results of the CFL3D code, which NASA's Langley Research Center provides [45].

The number of cells used in each case and computational time are listed in table 5.2. For the initial grid, using second-order reconstruction increased computational time by a factor of 1.34 compared to first-order reconstruction.

Table 5.2: Number of cells and computational time of cases for turbulent flow over a two-dimensional bump

Case No.	**Description**	**Number of Cells**	**Computational Time**
1	No Refinement with 1st order	20974	4 hours 44 minutes 13 seconds
2	One Refinement with 1st order	38636	7 hours 43 minutes 43 seconds
3	Two Refinement with 1st order	75452	11 hours 53 minutes 58 seconds
4	No Refinement with 2nd order	20974	6 hours 25 minutes 32 seconds
5	One Refinement with 2nd order	38713	18 hours 20 minutes 8 seconds
6	CFL3D	903169	-

Figures 5.13 and 5.14 show solution adapted grids used in Case 3. Black lines represent the base grid. Red lines represent refined ones.

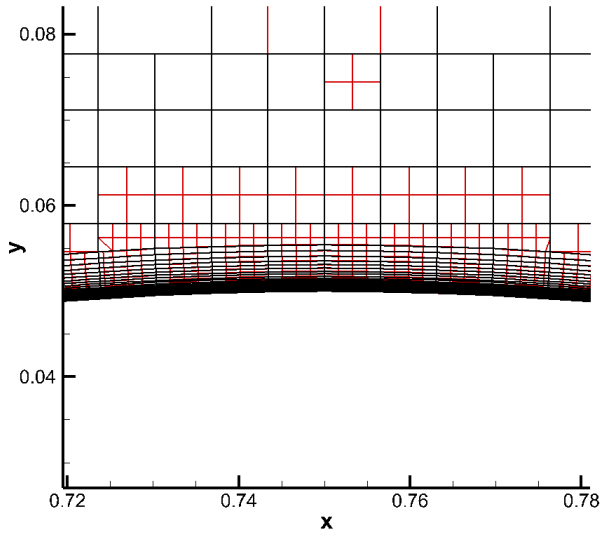


Figure 5.13: Grids around the middle section of the bump

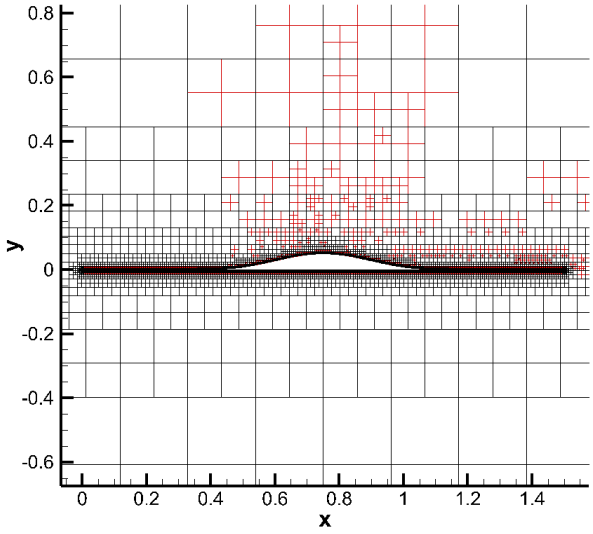
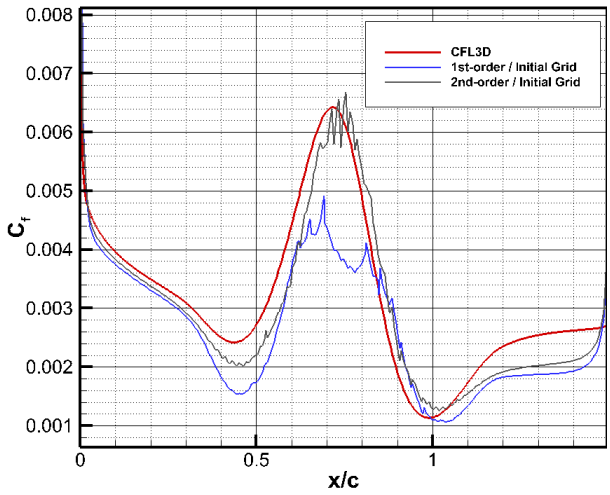


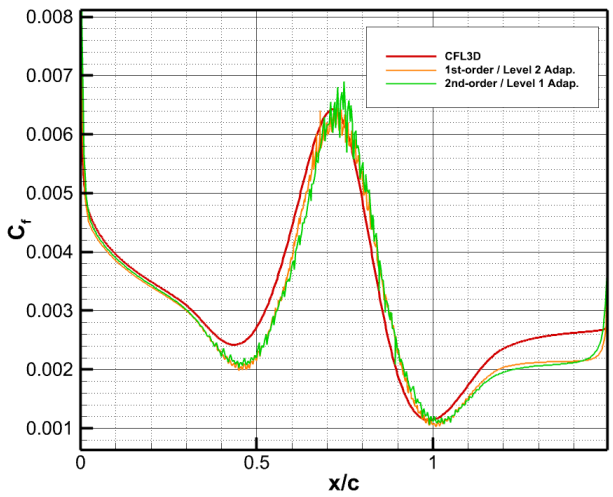
Figure 5.14: Grids around the two-dimensional bump

Skin friction coefficient along the bump geometry for Cases 1, 4 and Cases 3, 5 are compared with the reference result and presented in figures 5.15a, 5.15b. In figure 5.15a, it can be observed that the second-order solution (Case 4) has a better agreement with the CFL3D result than the first-order solution (Case 1), especially at the

middle section of the bump. Results presented in figure 5.15b show that first-order solutions with two-level refinement (Case 3) have slightly better agreement with reference results than the second-order solution with one-level refinement (Case 2). As can be observed from both figures, skin friction distribution is oscillatory. The reason for oscillations is the grid non-smoothness caused by level differences between Cartesian cells close to the boundary layer and cut cells, as shown in [46]. Since the generation methodology of body orthogonal boundary layer cells depends on cut cells at the interface, the non-smoothness is also presented on these grids. Thus increased non-smoothness of the grid can decrease accuracy and cause oscillations.



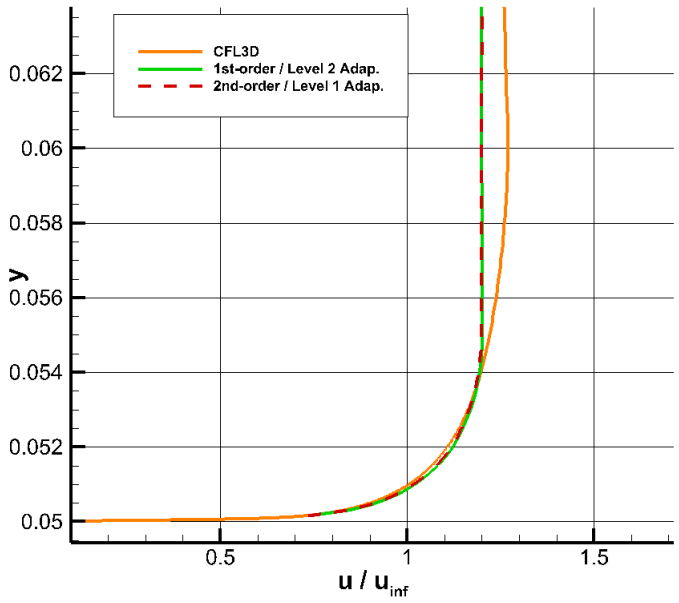
(a)



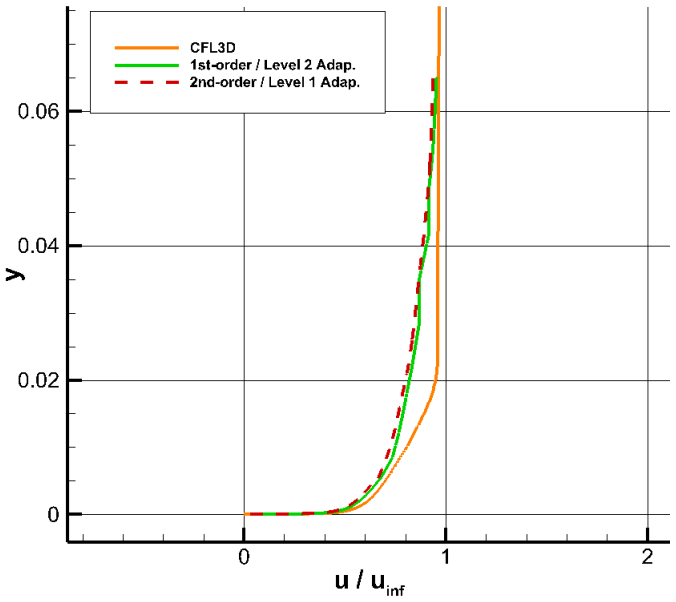
(b)

Figure 5.15: Skin friction coefficients along the bump for different refinement levels and their comparison with the results of CFL3D

Velocity profiles at two locations of x=0.75 and x=1.20148 are shown in the figure 5.16. For both locations, results are similar closer to the wall boundary, while the velocity is underestimated at farther locations.



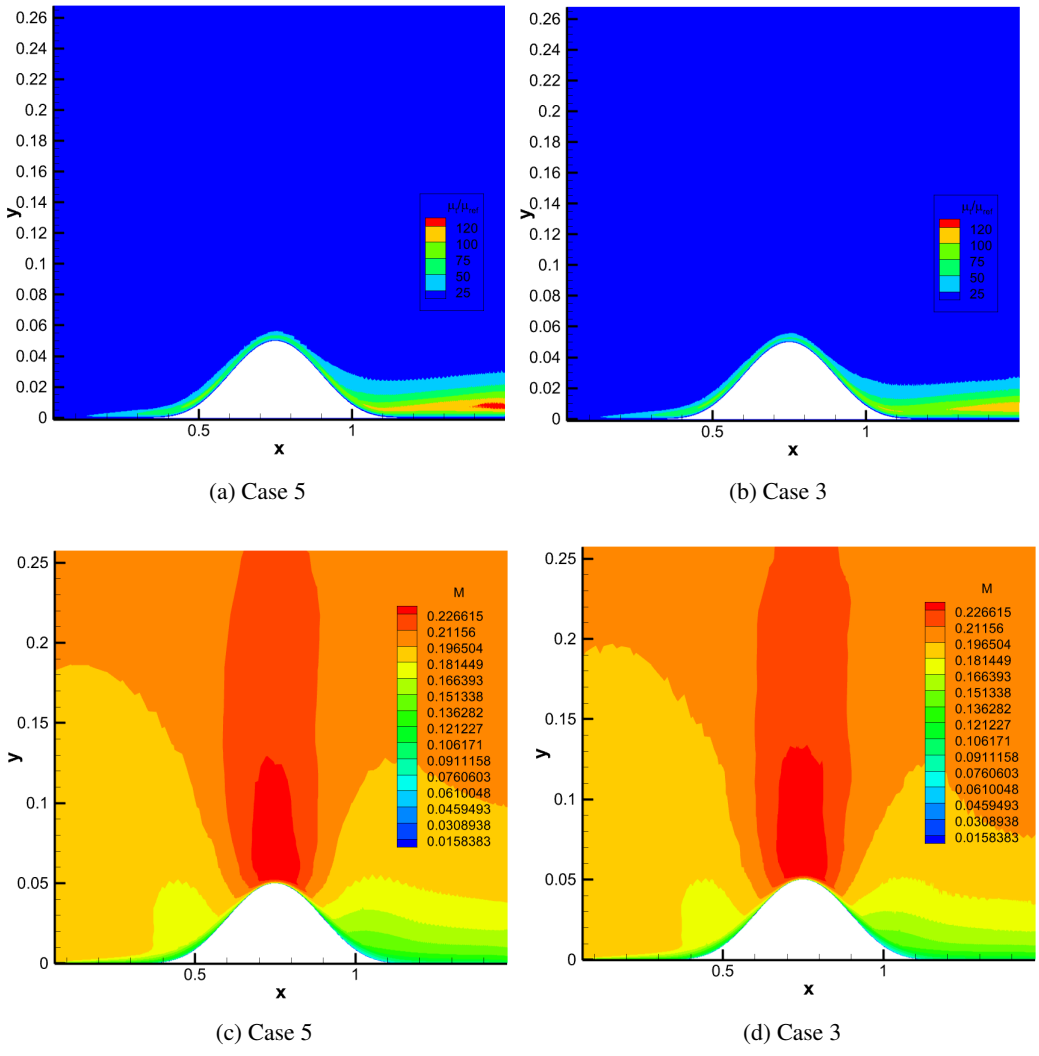
(a)



(b)

Figure 5.16: Velocity profiles at x=0.75 (a) and x=1.20148 (b)

Figure 5.17 shows turbulent viscosity and Mach contours for cases 3 and 5. In Case 5, turbulent viscosity is slightly higher at the trailing edge than in Case 3.



(a) Case 5

(b) Case 3

(c) Case 5

(d) Case 3

Figure 5.17: Non-dimensional turbulent viscosity and Mach contours for turbulent flow over a bump

As a result of tested cases, using WLSQ gave considerably more accurate results than the first-order scheme when the same mesh was used. Although geometric parameters

of WLSQ are precalculated, using it still increased the computational time by more than thirty percent.

5.3 Turbulent flow Over NACA0012 Airfoil

In this test problem, a subsonic turbulent flow over the NACA0012 airfoil is solved. The flow mach number is selected as $M_\infty = 0.15$, and the Reynolds number is $Re = 6 \times 10^6$. Two different angles of attack are selected, which are equal to 0° and 10°. The outer boundary factor is selected as 500 to ensure that far-field boundary conditions do not affect the solution. Boundary layer cells are generated using 35 rows and selecting the stretch factor as 1.2. The boundary layer cells' total thickness equals 0.026 times the chord length. The initial condition for turbulent eddy viscosity is selected as $\tilde{\nu} = 3.0\nu$ to provide a fully turbulent region from the start. The solver is iterated until the logarithm of root mean square of normalized continuity residual reaches -8. Table 5.3 lists the number of cells and computational time for all three refinement levels. In all test cases, second-order reconstruction is applied by setting the weight to one and using Venkatakrishnan's limiter.

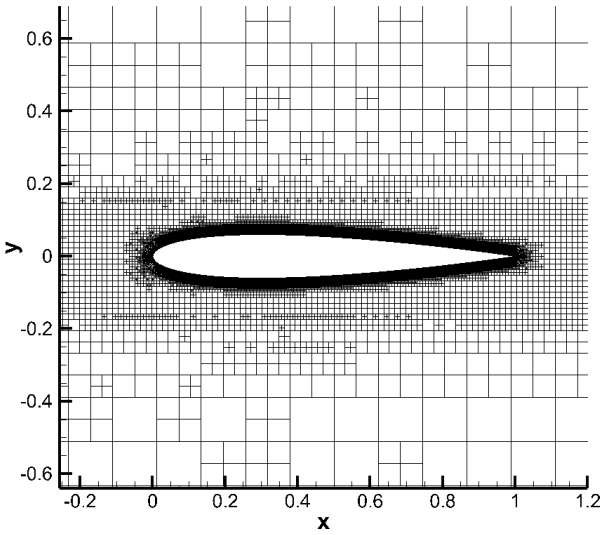
Table 5.3: Number of cells and computational time of cases for turbulent flow around NACA0012 Airfoil

Case No.	Description	Number of Cells	Computational Time
1	One Refinement Solution for $\alpha = 0^\circ$	52309	22 hours 55 minutes 7 seconds
2	Two Refinement Solution for $\alpha = 0^\circ$	98519	42 hours 24 minutes 58 seconds
3	One Refinement Solution for $\alpha = 10^\circ$	55463	25 hours 12 minutes 3 seconds
4	Two Refinement Solution for $\alpha = 10^\circ$	104107	44 hours 5 minutes 9 seconds

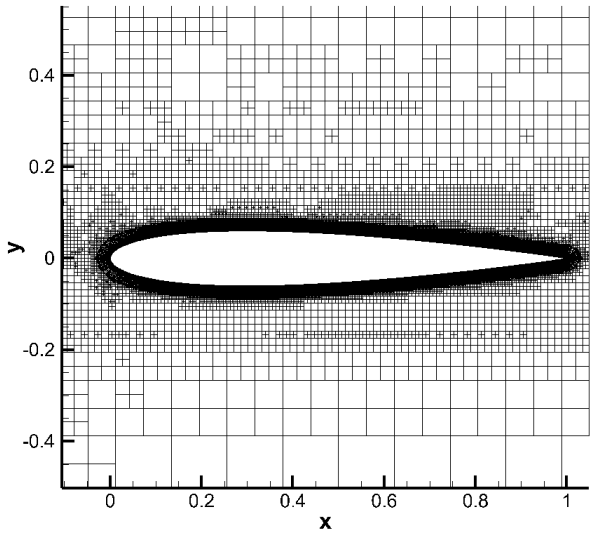
All cases' results are compared with Ladson's tripped data [47] except for the skin friction coefficient. The skin friction coefficient is compared with the results of the

CFL3D code provided by NASA's Langley Research Center [45].

The grids used for Cases 2 and 4 are shown in figure 5.18. Grids are refined approximately the same amount around the lower and upper surfaces of the airfoil in Case 2 with solution adaptation. Solution adaptation is triggered most around the upper surface in Case 4, where higher gradients are presented.



(a)



(b)

Figure 5.18: The grids around the NACA 0012 airfoil for Case 2 (a) and Case 4 (b) for the subsonic turbulent flow

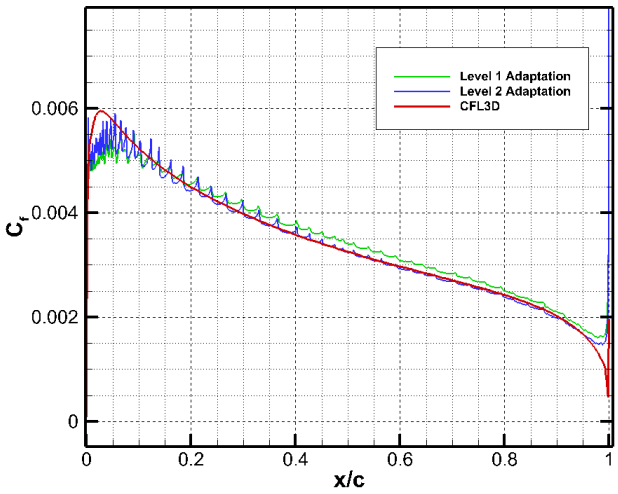


Figure 5.19: Skin friction coefficients along the upper surface of NACA0012 airfoil for the angle of attack 0°

Figure 5.19 shows skin friction coefficient distribution along the airfoil for Cases 1 and 2 with the results of CFL3D. Case 2 has better agreement with the reference result. However, at the leading edge, results scatter, and at the trailing edge, the skin friction coefficient is underpredicted compared with the reference result.

Figure 5.20 compares the skin friction coefficient for Cases 3 and 4 with the reference result provided for the upper surface of the airfoil. Results obtained from Case 4 show better agreement with CFL3D results while it underestimated the skin friction coefficient at the leading edge and overpredicted it at the trailing edge.

Pressure coefficient distribution is compared with Ladson's experimental data and shown in figures 5.21, 5.22 for the angle of attacks 0° and 10°, respectively. For the angle of attack 0°, Case 2 results are slightly closer to the experimental result along the chord than Case 1. For the angle of attack 10°, the results of Cases 3 and 4 show a good agreement with the experimental data at the lower surface. In Case 4, the leading edge upper surface pressure is resolved slightly better.

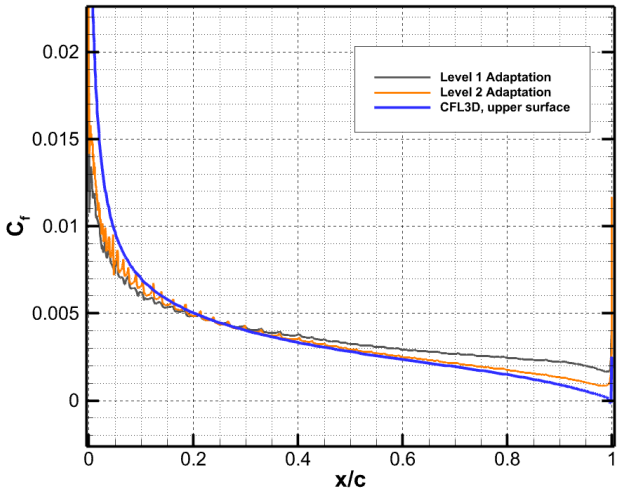


Figure 5.20: Skin friction coefficients along the upper surface of NACA0012 airfoil for the angle of attack 10°

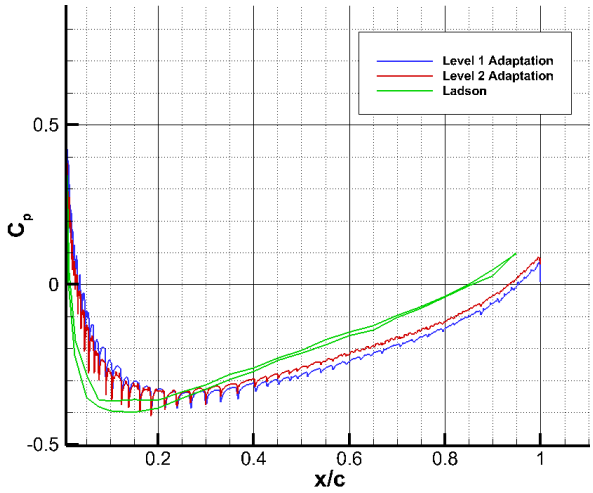


Figure 5.21: Pressure Coefficient distribution along the NACA0012 airfoil for 0 degrees of angle of attack

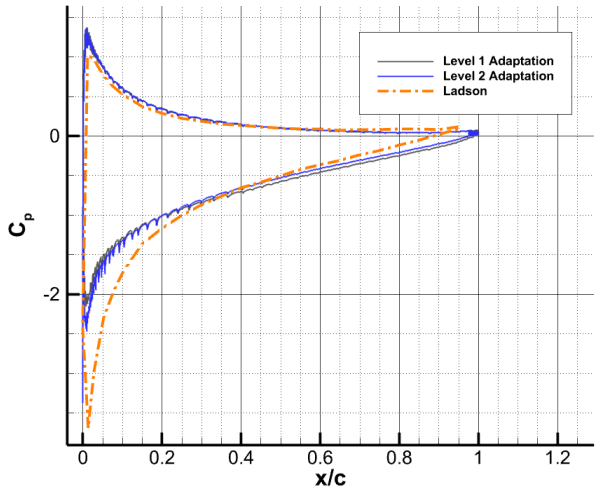


Figure 5.22: Pressure Coefficient distribution along the NACA0012 airfoil for 10 degrees of angle of attack

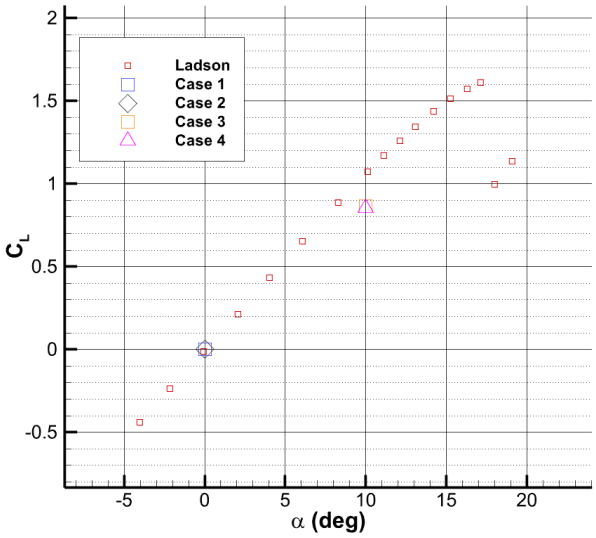


Figure 5.23: Lift coefficient vs. angle of attack

In all of the cases, both pressure and skin friction coefficients show oscillatory behavior. The magnitude of oscillations is higher at the leading edge of the airfoil, where non-smoothness is presented more than in other regions because of cut cells. Also, solution adaptation increases oscillation magnitude by increasing non-smooth

numbers of cells while it improves the accuracy of mean quantities.

Lastly, lift coefficients obtained from all cases are compared with the experimental curve and shown in figure 5.23. While lift coefficients obtained from Cases 1 and 2 are nearly the same as the experimental results, the lift coefficient is underestimated in Cases 3 and 4.

CHAPTER 6

CONCLUSION

This thesis study aims to solve two-dimensional, steady, compressible RANS equations for fully turbulent problems. A cell-centered finite volume approach with an explicit time scheme is used. The solution domain is discretized using off-body Cartesian and near-body conforming quadrilateral grids. Cartesian grids are cut and split at the intersection with the outer boundary of quadrilateral cells. The faces of cut-split and quad cells are matched precisely to provide fully conserved solutions. Mesh generation is done with as minimum user intervention as possible. Since there are minimum user interventions, geometric and solution adaptions are applied to have accurate solutions. Solution variables are reconstructed using the weighted or unweighted least squares approach and limited with Venkatakrishnan's limiter. Negative Spalart-Allmaras, one equation turbulence model, is used as a turbulence model. Model is used in its conserved form, and the production term is modified to alleviate numerical problems. In total, three different cases are presented to show the performance of the solution approach. The first two are famous test cases to check if the turbulence model is implemented accurately. The third one is selected to evaluate the accuracy of the approach for a turbulent flow over an airfoil.

Results show that with the increasing cycles of solution adaptation, the difference with the reference results is decreasing more and more. However, an increasing number of cells causes a significant increase in computational time. Also, significant improvements in accuracy results are obtained by using WLSQ for reconstruction, even when solution adaptation is not applied. In the first test case, it is shown that the accuracy of the turbulence model is sensitive to boundary layer resolution. Boundary layer cells are refined only vertically with each solution adaptation level. Thus it is

crucial to specify the thickness of boundary layer cells sufficiently at the beginning of the solution. Also, small cut-cells at the interface can cause low-quality boundary layer grids because their width becomes significantly small. It is one of the notable challenges that affect the accuracy of the solution and one of the reasons for the oscillatory behavior of flow quantities. One of the first improvements in the future may be on the mesh refinement approach and considering the treatment of the boundary between two different grid types.

Future works may be done to improve the current solver's accuracy and computational efficiency. These are,

- A parallel computation procedure
- A more efficient refinement approach
- Alleviating the small cut-cell problem

REFERENCES

[1] P. Moin and K. Mahesh, "Direct numerical simulation: A Tool in Turbulence Research," *Annual Review of Fluid Mechanics*, vol. 30, pp. 539–578, 1998.

[2] X. Wu and P. Moin, *A direct numerical simulation study on the mean velocity characteristics in turbulent pipe flow*, vol. 608. 2008.

[3] B. Baldwin and H. Lomax, "Thin-layer approximation and algebraic model for separated turbulentflows," in *16th aerospace sciences meeting*, p. 257, 1978.

[4] L. Prandtl *et al.*, "Uber ein neues formelsystem fur die ausgebildete turbulenz," *Nacr. Akad. Wiss. Gottingen, Math-Phys. Kl*, pp. 6–19, 1945.

[5] P. R. Spalart and S. R. Allmaras, "One-equation turbulence model for aerodynamic flows," *Recherche aerospatiale*, no. 1, pp. 5–21, 1994.

[6] D. DeZeeuw and K. G. Powell, "An adaptively refined cartesian mesh solver for the euler equations," *Journal of Computational Physics*, vol. 104, no. 1, pp. 56–68, 1993.

[7] W. J. Coirier, "Based Scheme for the Euler and Navier- Stokes Equations by," 1994.

[8] M. Delanaye, M. J. Aftosmis, M. J. Berger, Y. Liu, and T. H. Pulliam, "Automatic hybrid-cartesian grid generation for high-reynolds number flows around complex geometries," *37th Aerospace Sciences Meeting and Exhibit*, no. c, 1999.

[9] Z. J. Wang, "A quadtree-based adaptive Cartesian/Quad grid flow solver for Navier-Stokes equations," *Computers and Fluids*, vol. 27, no. 4, pp. 529–549, 1998.

[10] Z. Wang, "A fast nested multi-grid viscous flow solver for adaptive cartesian/quad grids," *International Journal for Numerical Methods in Fluids*, vol. 33, no. 5, pp. 657–680, 2000.

[11] Z. Wang and R. Chen, "Anisotropic solution-adaptive viscous cartesian grid method for turbulent flow simulation," *AIAA journal*, vol. 40, no. 10, pp. 1969–1978, 2002.

[12] S. Karman, "Hierarchical unstructured mesh generation," in *42nd AIAA Aerospace Sciences Meeting and Exhibit*, p. 613, 2004.

[13] M. Berger and M. Aftosmis, "Progress towards a cartesian cut-cell method for viscous compressible flow," in *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, p. 1301, 2012.

[14] M. J. Berger and M. J. Aftosmis, "An ode-based wall model for turbulent flow simulations," *AIAA Journal*, vol. 56, no. 2, pp. 700–714, 2018.

[15] W. Dawes, S. Harvey, S. Fellows, C. Favaretto, and A. Velivelli, "Viscous layer meshes from level sets on cartesian meshes," in *45th AIAA Aerospace Sciences Meeting and Exhibit*, p. 555, 2007.

[16] A. Katz, A. Jameson, and A. Wissink, "A multi-solver scheme for viscous flows using adaptive cartesian grids and meshless grid communication," in *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, p. 768, 2009.

[17] H. Luo, S. Spiegel, and R. Löhner, "Hybrid grid generation method for complex geometries," *AIAA journal*, vol. 48, no. 11, pp. 2639–2647, 2010.

[18] S. Park, B. Jeong, J. G. Lee, and H. Shin, "Hybrid grid generation for viscous flow analysis," *International Journal for Numerical Methods in Fluids*, vol. 71, no. 7, pp. 891–909, 2013.

[19] M. Özkan, "A cartesian based mesh generator with body fitted boundary layers," Master's thesis, Middle East Technical University, 2018.

[20] D. Hartmann, M. Meinke, and W. Schröder, "A strictly conservative cartesian cut-cell method for compressible viscous flows on adaptive grids," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 9-12, pp. 1038–1052, 2011.

[21] L. Schneiders, C. Günther, M. Meinke, and W. Schröder, "An efficient conservative cut-cell method for rigid bodies interacting with viscous compressible flows," *Journal of Computational Physics*, vol. 311, pp. 62–86, 2016.

[22] J. Benek, J. Steger, F. Dougherty, and P. Buning, "Chimera. a grid-embedding technique," tech. rep., ARNOLD ENGINEERING DEVELOPMENT CENTER ARNOLD AFB TN, 1986.

[23] J. Blazek, *Computational fluid dynamics: principles and applications*. Butterworth-Heinemann, 2015.

[24] L. Ramírez, X. Nogueira, P. Ouro, F. Navarrina, S. Khelladi, and I. Colominas, "A higher-order chimera method for finite volume schemes," *Archives of Computational Methods in Engineering*, vol. 25, no. 3, pp. 691–706, 2018.

[25] B. Siyahhan, "A two dimensional euler flow solver on adaptive cartesian grids," Master's thesis, Middle East Technical University, 2008.

[26] M. Çakmak, "Development of a multigrid accelerated euler solver on adaptively refined two- and three-dimensional cartesian grids," Master's thesis, Middle East Technical University, 2009.

[27] M. S. Şahin, "Development of a two-dimensional navier-stokes solver for laminar flows using cartesian grids," Master's thesis, Middle East Technical University, 2011.

[28] F. M. White and J. Majdalani, *Viscous fluid flow*, vol. 3. McGraw-Hill New York, 2006.

[29] A. Favre, "Equations des gaz turbulents compressibles," *J. de Mecanique*, vol. 4, no. 3, 1965.

[30] D. C. Wilcox *et al.*, *Turbulence modeling for CFD*, vol. 2. DCW industries La Canada, CA, 1998.

[31] T. B. Gatski and J.-P. Bonnet, *Compressibility, turbulence and high speed flow*. Academic Press, 2013.

[32] S. R. Allmaras, F. T. Johnson, and P. R. Spalart, "Modifications and clarifications for the implementation of the spalart-allmaras turbulence model," *7th International Conference on Computational Fluid Dynamics, ICCFD 2012*, no. July, pp. 9–13, 2012.

[33] P. R. Spalart and C. L. Rumsey, "Effective inflow conditions for turbulence models in aerodynamic calculations," *AIAA Journal*, vol. 45, no. 10, pp. 2544–2553, 2007.

[34] T. J. Barth, "Aspects of unstructured grids and finite-volume solvers for the euler and navier-stokes equations," *AGARD, special course on unstructured grid methods for advection dominated flows*, 1992.

[35] E. Shima, K. Kitamura, and T. Haga, "Green–gauss/weighted-least-squares hybrid gradient reconstruction for arbitrary polyhedra unstructured grids," *AIAA journal*, vol. 51, no. 11, pp. 2740–2747, 2013.

[36] T. Barth and D. Jespersen, "The design and application of upwind schemes on unstructured meshes," in *27th Aerospace sciences meeting*, p. 366, 1989.

[37] V. Venkatakrishnan, "On the accuracy of limiters and convergence to steady state solutions," in *31st Aerospace Sciences Meeting*, p. 880, 1993.

[38] V. Venkatakrishnan, "Convergence to steady state solutions of the euler equations on unstructured grids with limiters," *Journal of computational physics*, vol. 118, no. 1, pp. 120–130, 1995.

[39] P. L. Roe, "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, vol. 43, pp. 357–372, 1981.

[40] E. F. Toro, *Riemann solvers and numerical methods for fluid dynamics: a practical introduction*. Springer Science & Business Media, 2013.

[41] C. H. Tai, *Acceleration techniques for explicit Euler codes*. PhD thesis, University of Michigan, 1990.

[42] D. J. Mavriplis and A. Jameson, "Multigrid solution of the navier-stokes equations on triangular meshes," *AIAA journal*, vol. 28, no. 8, pp. 1415–1425, 1990.

[43] C. W. Jackson, W. C. Tyson, and C. J. Roy, "Turbulence model implementation and verification in the sensei cfd code," in *AIAA Scitech 2019 Forum*, p. 2331, 2019.

[44] H. Paillere and K. POWELL, "A wave-model-based refinement criterion for adaptive-grid computation of compressible flows," in *30th Aerospace Sciences Meeting and Exhibit*, p. 322, 1992.

[45] C. Rumsey, B. Smith, and G. Huang, "Description of a website resource for turbulence modeling verification and validation," in *40th Fluid Dynamics Conference and Exhibit*, p. 4742, 2010.

[46] W. J. Coirier, *An adaptively-refined, Cartesian, cell-based scheme for the Euler and Navier-Stokes equations*. University of Michigan, 1994.

[47] C. L. Ladson, *Effects of independent variation of Mach and Reynolds numbers on the low-speed aerodynamic characteristics of the NACA 0012 airfoil section*, vol. 4074. National Aeronautics and Space Administration, Scientific and Technical . . . , 1988.