# PARALLEL PROCESSING OF THREE-DIMENSIONAL NAVIER-STOKES EQUATIONS FOR COMPRESSIBLE FLOWS

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

BY

TAHSİN ÇAĞRI ŞİŞMAN

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

SEPTEMBER 2005

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan ÖZGEN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Kemal İDER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

| | |
|---|---|
| Asst. Prof. Dr. Cüneyt SERT<br>Co-Supervisor | Prof. Dr. M. Haluk AKSEL<br>Supervisor |

**Examining Committee Members**

| | | |
|---|---|---|
| Prof. Dr. Zafer DURSUNKAYA | (METU, ME) | |
| Prof. Dr. M. Haluk AKSEL | (METU, ME) | |
| Asst. Prof. Dr. Cüneyt SERT | (METU, ME) | |
| Instructor Dr. Tahsin ÇETİNKAYA | (METU, ME) | |
| Dr. L. Oktay GÖNÇ | (TÜBİTAK-SAGE) | |

# ABSTRACT

PARALLEL PROCESSING OF THREE-DIMENSIONAL
NAVIER-STOKES EQUATIONS FOR COMPRESSIBLE FLOWS

ŞİŞMAN, Tahsin Çağrı
M.Sc., Department of Mechanical Engineering
Supervisor : Prof. Dr. M. Haluk AKSEL
Co-Supervisor: Asst. Prof. Dr. Cüneyt SERT

September 2005, 95 pages

The aim of this study is to develop a code that is capable of solving three-dimensional compressible flows which are viscous and turbulent, and parallelization of this code. Purpose of parallelization is to obtain a computational efficiency in time respect which enables the solution of complex flow problems in reasonable computational times.

In the first part of the study, which is the development of a three-dimensional Navier-Stokes solver for turbulent flows, first step is to develop a two-dimensional Euler code using Roe flux difference splitting method. This is followed by addition of sub programs involving calculation of viscous fluxes. Third step involves implementation of Baldwin-Lomax turbulence model to the code. Finally, the Euler code is generalized to three-dimensions. At every step, code validation is done by comparing numerical results with theoretical, experimental or other numerical results, and adequate consistency between these results is obtained.

In the second part, which is the parallelization of the developed code, two-dimensional code is parallelized by using Message Passing Interface (MPI), and important improvements in computational times are obtained.

Keywords: Navier-Stokes Equations, Parallel Processing, Roe Flux Vector Splitting, Baldwin-Lomax Turbulence Model, MPI

# ÖZ

## ÜÇ BOYUTLU SIKIŞTIRILABİLİR AKIŞLAR İÇİN NAVİER-STOKES DENKLEMLERİNİN PARALEL HALE GETİRİLMESİ

ŞİŞMAN, Tahsin Çağrı
Yüksek Lisans, Makina Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. M. Haluk AKSEL
Ortak Tez Yöneticisi : Prof. Dr. Cüneyt SERT

Eylül 2005, 95 pages

Bu çalışmada, ağdalı ve türbülanslı, üç boyutlu sıkıştırılabilir akış problemlerini çözme yeteneğine sahip bir kod geliştirmek ve geliştirilen bu kodu paralel hale getirmek amaçlanmıştır. Kodun paralel hale getirilmesiyle zaman bakımından hesap verimliliğine sahip olunacak ve bu sayede karmaşık akış problemlerini makul hesap zamanlarında çözmek mümkün olabilecektir.

Çalışmanın ilk basamağı olan türbülanslı akışlar için üç boyutlu Navier-Stokes çözücü geliştirme sürecinde, ilk basamağı Roe akı farkı paylaştırma metodu kullanılarak, iki boyutlu bir Euler çözücü geliştirilmesi oluşturmaktadır. Bu basamağı ağdalı akış hesaplarını yapan alt programların çözücüye eklenmesi takip etmiştir. Üçüncü basamak Baldwin-Lomax türbülans modelinin uygulanmasını içermektedir. Son olarak ise Euler çözücüsü üç boyuta genellenmiştir. Her basamakta çözücünün doğrulaması sayısal sonuçların teorik, deneysel ya da diğer sayısal sonuçlar ile karşılaştırılmasıyla yapılmış ve yeterli bir uyum elde edilmiştir.

Çalışmanın ikinci basamağı olan geliştirilen kodun paralel hale getirilmesi sürecinde, iki boyutlu çözücü Mesaj Aktarım Arayüzü (MPI) kullanılarak paralel hale getirilmiş ve hesap zamanlarında önemli miktarda azalma sağlanmıştır.

Anahtar Kelimeler: Navier-Stokes Denklemleri, Paralel Programlama, Akı Vektörü Paylaştırma Metodu, Baldwin-Lomax Türbülans Modeli, MPI

***To My Mother***

## ACKNOWLEDGEMENTS

The author wishes to thank his supervisor Prof. Dr. M. Haluk AKSEL and co-supervisor Asst. Prof. Dr. Cüneyt SERT for their guidance, advice and criticism throughout the research.

The author would also like to thank Prof. Dr. İsmail Hakkı TUNCER for giving such a valuable course which shifted level of technical knowledge to the level of the M. Sc. Study.

The author would also like to thank Mr. Enver Doruk ÖZDEMİR for his friendship and technical assistance throughout the research.

The author would also like to thank his family and Ms. Kiraz ŞEVİK for their endless encouragement and support.

## TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF SYMBOLS

| | |
|---|---|
| A | Area |
| $C_{fx}$ | Local skin friction coefficeint |
| $c_p$ | Constant pressure specific heat |
| $c_v$ | Constant volume specific heat |
| d | Distance between a point and a surface |
| E | Energy |
| e | Internal energy per unit mass |
| $e_t$ | Total energy per unit mass |
| $\vec{\mathbf{F}}$ | Fluxes, Convective fluxes |
| $\vec{\vec{\mathbf{F}}}$ | Flux tensor, vector of convective fluxes |
| $\vec{\mathbf{f}}_{\mathbf{b}}$ | Body forces per unit mass |
| $\vec{\mathbf{F}}_{C}$ | Convective fluxes |
| $\vec{\vec{\mathbf{F}}}_{C}$ | Convective flux tensor |
| $\vec{\mathbf{F}}_{\mathbf{D}}$ | Diffusive fluxes |
| $\vec{\vec{\mathbf{F}}}_{\mathbf{D}}$ | Diffusive flux tensor |
| $\vec{\mathbf{G}}$ | Vector of source terms |
| h | Height |
| $h_t$ | Total enthalpy |
| $\vec{\vec{\mathbf{I}}}$ | Unit tensor |

| | |
|---|---|
| $\hat{\mathbf{i}}$ | Unit vector in *x*-direction |
| $\hat{\mathbf{j}}$ | Unit vector in *y*-direction |
| $F_{Kleb}$ | Klebanoff's intermittency function |
| k | Thermal conductivity |
| $\hat{\mathbf{k}}$ | Unit vector in *z*-direction |
| $\vec{\mathbf{N}}$ | Normal vector of a plane |
| $\hat{\mathbf{n}}$ | Unit normal vector |
| Pr | Prandtl number |
| p | Pressure |
| Q | A scalar quantity, heat transfer |
| $\mathbf{Q}$ | Vector of conservative variables |
| q | Scalar quantity *Q* per unit mass |
| $\vec{\mathbf{Q}}$ | A vector quantity |
| $Q_V$ | Volume sources |
| $\vec{\mathbf{Q}}_V$ | Volume source vector |
| $\vec{\mathbf{Q}}_S$ | Surface sources |
| $\vec{\vec{\mathbf{Q}}}_S$ | Surface source tensor |
| R | Ideal gas constant |
| $\vec{\mathbf{R}}$ | Residual |
| $\vec{\mathbf{r}}$ | Position vector |
| Re | Reynolds number |
| S | Surface |

| | |
|---|---|
| $\vec{\mathbf{S}}$ | Surface vector |
| T | Absolute temperature |
| t | Time |
| $U^+$ | Dimensionless velocity |
| $U_\infty$ | Velocity of freestream |
| u | Velocity component in *x*-direction |
| $u_\tau$ | Friction velocity |
| V | Volume |
| $\vec{\mathbf{V}}$ | Velocity vector |
| v | Velocity component in *y*-direction |
| W | Work |
| w | Velocity component in *z*-direction |
| y+ | Dimensionless distance |
| $\vec{\nabla}$ | Gradient operator |
| $\partial$ | Partial derivative operator |

Greek Letters

| | |
|---|---|
| $\delta$ | Boundary layer thickness |
| $\mathbf{\Phi}$ | Convective flux vector normal to the surface |
| $\gamma$ | Ratio of specific heats |
| $\kappa$ | Diffusivity constant, thermal diffusivity constant |
| $\lambda$ | Second viscosity coefficient |

| | |
|---|---|
| $\mu$ | Dynamic viscosity |
| $\mu_{lam}$ | Laminar part of dynamic viscosity |
| $\mu_{tot}$ | Total dynamic viscosity |
| $\mu_{tur}$ | Turbulent part of dynamic viscosity |
| $\rho$ | Density |
| $\vec{\vec{\boldsymbol{\sigma}}}$ | Stress tensor |
| $\tau$ | Shear stress |
| $\vec{\vec{\boldsymbol{\tau}}}$ | Shear stress tensor |
| $\tau_w$ | Shear stress on the wall |
| $\Omega$ | Volume |
| $\omega$ | Magnitude of vorticity vector |

Subscripts

| | |
|---|---|
| E | Edge |
| I | Index representing *x*-direction order of nodes |
| i | Index representing *x*-direction order of cells |
| J | Index representing *y*-direction order of nodes |
| j | Index representing *y*-direction order of cells |
| K | Index representing *z*-direction order of nodes |
| k | Index representing *z*-direction order of cells |
| L | The cell on the left of a surface or an edge |
| max | Maximum value of a variable |

| | |
|---|---|
| n | Normal component of a vector to a surface |
| R | The cell on the right of a surface or an edge |
| t | Tangential component of a vector to a surface |

Superscripts

| | |
|---|---|
| n | Current time step value of a variable |

# CHAPTER 1

# INTRODUCTION

## 1.1 Computational Fluid Dynamics Overview

Invention and development of computers enables scientists and engineers to use a new way of approach their problems which is using numerical methods as stated by Hirsch [1]. Nowadays, numerical methods are widely used in every engineering discipline as in fluid mechanics under the name of Computational Fluid Dynamics (CFD). Current state of CFD is beyond a research tool and as Wesseling states [2], “CFD has become an indispensable tool for engineering design optimizations, because many different configurations can be investigated at acceptable cost and in short time”. Companies and institutions in the fields varying from aerospace, automotive, biomedical and electronic industries to meteorology and sports; use commercial CFD packages as given in Aksel [3].

In fluid mechanics, like in other engineering disciplines, besides numerical techniques, there are two other approaches for solving a problem encountered which are solving the set of mathematical equations that define the problem and performing experiments [3, 4, 5]. Solving the set of mathematical equations that defines a fluid problem is not possible for most applications. Governing equations in fluid mechanics involve a set of non-linear partial differential equations and these complicated set of equations can not be solved in their full form without simplifying assumptions. Fluid flows for which an explicit mathematical solution is obtained are very simple flows under a number of simplifying assumptions. These simple problems generally give basic, but valuable physical insight for the flow phenomena; however, generally do not appear in engineering applications directly. Performing experiments is another way to cope with a fluid problem. As stated in [3], if experiments are done on a full scale prototype under actual conditions, it is possible to obtain data of actual conditions and it is the most reliable way. However, it is not always possible, since it is either very expensive or impossible such as the examples given in [2], flow around a space vehicle at re-entry, or a loss-of-coolant accident in a nuclear reactor. Experiments are also time consuming processes, they need a lot of investment for laboratory equipments and for the prototype itself, and the real flow is always disturbed with measurement devices more or less.

When CFD is compared with performing experiments, firstly, it is possible to investigate fluid flows for which performing experiments are not possible due to the restrictions imposed by the flow conditions such as dimensional restrictions [3, 5]. Secondly, CFD yields more detailed information with a low relative cost, as compared to experiments [3]. If CFD is compared with solving fluid flow equations, it can easily be seen that it is possible to obtain results for complex fluid problems in geometry and physical nature with CFD, where mathematical solutions of governing equations of these kinds of flows are not possible.

As stated by Wesseling [2], “CFD lacks the elegance and unification of classical fluid mechanics, and is in a state of rapid development”. In the future, the evolution of CFD will continue with the evolution of computers and newly developed mathematical models.

## 1.2 Review of Literature

This study involves development a Navier-Stokes solver containing a turbulence model and implementation of parallel processing to this solver. Review of literature containing these concepts is given in the following sections.

### 1.2.1 Navier-Stokes Solver

To solve Navier-Stokes equations numerically, first equation discretization should be done. In the equation discretization, most widely used methods are finite difference method, finite element method, and finite volume method [1]. In fluid mechanics applications, as stated by Wesseling in [2], “finite difference and finite volume methods are predominant”.

Finite difference method is based on the Taylor series expansion and derivatives are replaced by finite difference approximations [1, 5]. In Hirsch [1], it is stated that “finite difference method requires high degree of regularity of the mesh”.

As stated in Hirsch [1], finite volume method is introduced independently by McDonald in [6], and by Mac-Cormack and Paullay [7]. Finite volume method does not require regularity of meshes as in finite difference method. In finite volume method, integral forms of conservation equations are discretized. By this discretization, even at the discrete level of the equations, it is possible to ensure “basic quantities; mass, momentum and energy will be conserved”. [1]

In this study, finite volume method is used as a scheme for equation discretization.

After the conservation laws in integral form are discretized, a set of ordinary differential equations in time is obtained. By applying time integration scheme to this set yields a set of algebraic equations for a time level. Unknowns in these algebraic set of equations are the conservative flow variables at mesh points. Some of the terms in these equations are convective terms and some of them are diffusion terms. [1]

Time integration schemes can be divided into two main classes, namely explicit and implicit methods. In the algebraic set of equations for a time level, difference of convective and diffusive terms gives the change in the conservative variables at that time level. This change is function of flow variables and there are two possibilities for calculating this change. In the first possibility, convective and diffusive terms are calculated by using the *known* values of conservative flow variables at the current time step. This method is called the explicit method. By using this method, coefficient matrix of unknown conservative variables of next time step becomes a diagonal matrix and solution is trivial. "However, this advantage is counterbalanced by the fact that stability and convergence conditions impose severe restrictions on the maximum admissible time step" [1]. "While this might not be a limitation for physical unsteady problems which lead to the necessity of a large number of time steps" [1]. The second possibility is representing convective and diffusive fluxes by using the *unknown* values of conservative flow variables of the next time step. This method is called the implicit method, and coefficient matrix of unknown conservative variables of next time step is not a diagonal matrix in this case. However, coefficient matrix is block-diagonal and it can be solved by using simple algorithms. Implicit methods require higher number of operations than explicit methods; however, there are no time step limitations. [1]

In this study, the explicit method for integrations in time is used. By this way, developed code is capable of solving unsteady flow problems.
Navier-Stokes equations reduce to Euler equations with inviscid and non-heat conducting flow assumptions. System of Euler equations have terms having convective nature. Most of the techniques that are applied to solve system of Euler equations are directly applied to the calculation of convective terms of system of Navier-Stokes equations. Diffusive terms in the system of Navier-Stokes equations, which are the heat conduction and shear stress terms, are calculated by discretization of these terms centrally.

As stated in Hirsch [1], schemes for solution of system of Euler equations are divided into two main groups: 1) Space-centered schemes, 2) Upwind schemes. Pioneering study for space-centered schemes is done by Lax and Wendroff [8, 9, 10], and MacCormack [11] and they developed the most popular explicit scheme of this kind [1]. Lerat [12] developed the implicit generalization of space-centered schemes [1]. Space-centered schemes involve combined discretization in time and space by using Taylor series expansion, and they are second order accurate [1].

“Another approach to space-centered schemes is separate space and time discretization” [1]. Implicit schemes of this variant is developed by Briley and McDonald [13], and Beam and Warming [14] as given in [1]. Explicit scheme of this variant is introduced by Jameson *et al.* [15] involving fourth order Runge-Kutta time integration [1].

The other main group of schemes, namely upwind schemes involves the basic concept of determining convective fluxes by considering “characteristic propagation of properties” in inviscid flows [1]. The first upwind scheme is introduced by Courant *et al.* [16] as given in [1]. Upwind schemes can be divided into two sub-groups: a) Flux vector splitting methods, b) Flux difference splitting methods [1].

Flux vector splitting methods are “based on directional discretization of flux vectors” and works of Steger and Warming [17], and Van Leer [18] are in this group [1].

Flux difference splitting methods involve the idea of solving the Riemann problem between neighboring cells, and this approach is first given by Godunov [19] as stated in [1]. In Godunov’s method, exact solution of Riemann problem is done [1]. Engquist and Osher [20], Osher [21] and Roe [22, 23] introduced their approximate Riemann solvers to solve the Riemann problem between neighboring cells as stated in [1].

In this study, convective fluxes of system of Navier-Stokes equations are solved with flux difference splitting method of Roe.

### 1.2.2 Turbulence Models

Turbulence modeling is one of the important elements of CFD, since turbulence is a very widely encountered flow phenomenon in engineering applications. Turbulence is a very complex phenomenon. Turbulent flows are “highly unsteady” and three-dimensional with a “strongly rotational nature”.  “Enhanced diffusivity” involved in turbulence cause “an increase in the rate at which conserved quantities are stirred”. This mixing process is a dissipative process. Turbulence involves a wide range of time and length scales. [24, 25]

Turbulence models can be given in the following four main categories as stated by Wilcox [25]:

1. Algebraic (Zero-Equation) Models
2. One-Equation Models
3. Two-Equation Models
4. Stress-Transport Models

Turbulence models of Cebeci and Smith [26] and Baldwin and Lomax [27] are the widely used algebraic model group of turbulence models and known by the names of their developers [25]. In these models, eddy viscosity is calculated in terms of mixing length. [25]

Bradshaw, Ferriss and Atwell [28] introduced the most successful of early one equation models. At the beginning of 1990's, a number of turbulence models of this kind were introduced [25]. The model developed by Spalart and Allmaras [29] is "the most accurate one of these models for practical turbulent-flow applications" [25].

Widely used two-equation models are $k$-$\varepsilon$ and $k$-$\omega$ models. Launder and Spalding [30] introduced $k$-$\varepsilon$ model and the most widely used two-equation model [25]. This model is not successful in predicting adverse pressure effects [25]. Kolmogorov and Saffman [31] introduced $k$-$\omega$ model independently, and it is successful in predicting effects of adverse pressure gradient [25].

Stress-Transport models involve large number of equations and they are more complex compared to other methods. Due to these disadvantages, these models are not widely used when compared to algebraic and two-equation models. [25]

The above methods are based on Reynolds-closure approximations. Direct Numerical Simulation (DNS) and Large Eddy Simulation (LES) are the two of other methods for turbulence modeling and these methods are not based on Reynolds-closure approximations. DNS method involves "a complete three-dimensional and time-dependent solution of the Navier-Stokes and continuity equations". Although DNS is the most accurate way of investigating turbulent flows, due to grid size and time step limitations, it is only possible to study low Reynolds number flows of simple geometries. LES method involves the computation of large scale eddies, while the sub-grid scale eddies are modeled. "Large eddies are directly affected by boundary conditions and carries most of the Reynolds stress, so they should be computed". LES requires grid sizes much larger than DNS, since small size eddies are modeled. [25]

In this study, Baldwin-Lomax turbulence model is used due to its simplicity and easy implementation.

### 1.2.3 Parallel Computing

As stated in [32], parallel computing environments are started to be as "a general technical computing environment". Parallel computing is the main direction where high performance computing going towards [33], and its usage is continuously increasing in engineering applications.

Parallel computing is used in cases when "a decrease in runtime needed to solve a problem" or an increase in the size or the resolution of a problem is required [33]. CFD applications requires high CPU power and high memory resources for detailed analysis sufficient enough to get the physics of flow in reasonable run times. Parallel computing gives a good solution for these demands in CFD applications. Runtime decrease is achieved by increasing the number of processors for solving the problem, while increase in the size of the problem is achieved by increasing the available memory allocated to the problem. By using parallel computing tools, a code for solving a problem can be divided into multiple fragments, and these fragments are distributed to the processes in a parallel environment. Each fragment operates on its own process, which involves its own processor and memory space, simultaneously with other fragments. By this way, computing power and memory is increased. [34, 5]

Parallel systems can be divided into two types: 1) Centralized memory systems, 2) Distributed memory systems [34, 33]. Centralized memory systems made up of processors that use the same memory and input/output (I/O), and this kind is referred to as Symmetric Multiprocessor (SMP) architecture [34]. In SMP systems, "thread based programming" is applied by using OpenMP or special compilers that allows this feature [33]. Tasks on the processors communicate by using the same memory [32].

Distributed memory systems are made up of "nodes connected by a network that is usually high-speed" [34]. "Each node has its own processor and memory" [34]. This architecture is divided into two sub-kinds which are a) Massively Parallel Processors (MPP) systems, b) Non-Uniform Memory Access (NUMA) systems. In MPP systems, "there is an operating system on each node, so each node can be considered as a workstation" [34]. Linux clusters belong to this category. In these systems, "processors communicate each other by sending data packets" via network connection [32]. For communication, MPI (Message Passing Interface) or PVM (Parallel Virtual Machine) "message passing libraries" are used [33]. In NUMA systems, "a single operating system controls the whole system like in SMP systems" [34]. Processors reach memory in different amount of times depending upon whether they access it directly or over network, in other words memory latency differs when accessing memory directly or over network [34].

In this study, MPP system of Linux clusters is used, since it provides the best price/performance value when compared to other systems, and it is possible to add and remove nodes to the system.

## 1.3 Present Study

In this study, development of a three-dimensional Navier-Stokes solver with a turbulence model, and parallelization of this solver is aimed. This study involves several steps which are: 1) Development of Euler solver, 2) Generalization of Euler solver to Navier-Stokes solver, 3) Implementation of Baldwin-Lomax turbulence model, 4) Parallel processing of two-dimensional solver, 5) Generalization of two dimensional Euler solver to three dimensions.

First step in this study is development of an Euler solver in two-dimensional space. Development of cell-centered and Roe flux difference splitting scheme implemented Euler solver is done by a joint study with Enver Doruk Özdemir. During the development of this code, “Euler2d” code developed by Prof. Dr. M. Haluk Aksel, and “Set2D” code developed by Dr. Mehmet Ali Ak is taken as basis. Information on “Set2D” code can be obtained from Ak [40] and Gönç [41]. Implementation of Roe flux difference splitting scheme is done by using [35].

Second step involves generalization of the developed Euler solver to Navier-Stokes solver. In this step, “Ns2d” developed by Prof. Dr. M. Haluk Aksel and viscous subroutines developed on “Set2D” in AE 546 course project by a joint study with Enver Doruk Özdemir is taken as reference besides other references and a different kind of implementation of general idea by the author.

Third step is implementation of Baldwin-Lomax model. “Ns2dt” code developed by Prof. Dr. M. Haluk Aksel and turbulent Navier-Stokes solver developed by M. Sarp Yalım is taken as reference besides the references [25, 36, 4].

Fourth and fifth steps are done without taking any code as a general reference. In parallel processing, optimized parallel code segments given by Aoyama and Nakano in [34] are used in the parallel code segments developed by the author.

Governing equations of a fluid flow is presented in Chapter 2. Numerical implementation details which are discretization of governing equations, Roe flux difference splitting method, viscous flux calculation, Baldwin-Lomax turbulence model, boundary conditions are given in Chapter 3. Results obtained from test cases of inviscid flow, laminar and turbulent flows, and parallel processing are given in Chapter 4. Also, discussions on these results are presented in this chapter. Finally, conclusions of the present study are given in Chapter 5.

# CHAPTER 2

# GOVERNING EQUATIONS

## 2.1 Conservation Law

Conservation laws represent dynamical behavior of a physical system completely. During the time development of a physical system, certain quantities which are mass, momentum and energy are conserved. Conservation equations of these quantities describe the dynamical behavior of the system completely without any other dynamical law. [1]

For a physical quantity which can be a scalar or vector in a volume $\Omega$ bounded by a surface $S$, conservation law states that time variation of the physical quantity in the volume is due to net effects of the internal sources and amount of quantity crossing the boundary surface which are fluxes [1, 37].

### 2.1.1 Conservation Law for a Scalar Quantity

Conservation law for a scalar quantity $Q$ is given in [1] as;

$$\frac{\partial}{\partial t}\int_{\Omega} Q \, \mathrm{d}\Omega = -\oint_{S} \vec{\mathbf{F}} \cdot \mathrm{d}\vec{\mathbf{S}} + \int_{\Omega} Q_V \, \mathrm{d}\Omega + \oint_{S} \vec{\mathbf{Q}}_{\mathrm{S}} \cdot \mathrm{d}\vec{\mathbf{S}} \qquad (2.1)$$

where $t$ is time, $\vec{\mathbf{F}}$ represents fluxes, $Q_V$ and $\vec{\mathbf{Q}}_{\mathrm{S}}$ represent volume and surface sources, respectively.

The term on the left is the time variation of $Q$ in the volume. First term on the right gives the contribution of fluxes through the surfaces, and the other terms on the right are the contributions from volume sources and surface sources, respectively. The minus sign appearing in front of the flux term is due to the fact that surface vector $\mathrm{d}\vec{\mathbf{S}}$ always points outwards of the volume by convention. Rearranging Eqn. (2.1) yields the general form of conservation law for a scalar quantity [37]

$$\frac{\partial}{\partial t}\int_{\Omega} Q\ \mathrm{d}\Omega \ + \ \oint_S \vec{\mathbf{F}}\cdot \mathrm{d}\vec{\mathbf{S}} \ = \ \int_{\Omega} Q_V\ \mathrm{d}\Omega \ + \ \oint_S \vec{\mathbf{Q}}_{\mathbf{S}}\cdot \mathrm{d}\vec{\mathbf{S}} \qquad (2.2)$$

Flux vector $\vec{\mathbf{F}}$ consists of two parts which are convective flux $\vec{\mathbf{F}}_{\mathbf{C}}$ and diffusive flux $\vec{\mathbf{F}}_{\mathbf{D}}$. Flux formed by the motion of mass is the convective flux and it can be given as;

$$\vec{\mathbf{F}}_{\mathbf{C}} \ = \ Q\vec{\mathbf{V}} \qquad (2.3)$$

where $\vec{\mathbf{V}}$ is the velocity of the mass.

Diffusive flux is as a result of molecular motion which is present even if the fluid is at rest. Property gradients yield the molecular diffusion. By using Fick’s Law, diffusive flux can be given as;

$$\vec{\mathbf{F}}_{\mathbf{D}} \ = \ -\kappa\rho\vec{\nabla}q \qquad (2.4)$$

where $\kappa$ is the diffusivity constant having units of $\mathrm{m}^2/\mathrm{s}$, $\rho$ is the density and $q$ is the quantity $Q$ per unit mass.

Conservation law for scalar quantity, $Q$ in Eqn. (2.2) takes the following form, when Gauss's theorem is applied to surface integrals; [1, 37]

$$\int_{\Omega} \frac{\partial Q}{\partial t}\, \mathrm{d}\Omega \;+\; \int_{\Omega} \vec{\nabla}\cdot\vec{\mathbf{F}}\, \mathrm{d}\Omega \;=\; \int_{\Omega} Q_V\, \mathrm{d}\Omega \;+\; \int_{\Omega} \vec{\nabla}\cdot\vec{\mathbf{Q}}_{\mathbf{S}}\, \mathrm{d}\Omega \qquad (2.5)$$

In the application of Gauss's theorem, it is assumed that fluxes and surface sources are continuous, and volume $\Omega$ is fixed.

Eqn. (2.5) holds for an arbitrary volume and this yields the differential form which is given as;

$$\frac{\partial Q}{\partial t} \;+\; \vec{\nabla}\cdot\vec{\mathbf{F}} \;=\; Q_V \;+\; \vec{\nabla}\cdot\vec{\mathbf{Q}}_{\mathbf{S}} \qquad (2.6)$$

### 2.1.2 Conservation Law for a Vector Quantity

Conservation law for a vector quantity $\vec{\mathbf{Q}}$ is;

$$\frac{\partial}{\partial t}\int_{\Omega} \vec{\mathbf{Q}}\, \mathrm{d}\Omega \;=\; -\oint_{S} \vec{\vec{\mathbf{F}}}\cdot \mathrm{d}\vec{\mathbf{S}} \;+\; \int_{\Omega} \vec{\mathbf{Q}}_{\mathbf{V}}\, \mathrm{d}\Omega \;+\; \oint_{S} \vec{\vec{\mathbf{Q}}}_{\mathbf{S}}\cdot \mathrm{d}\vec{\mathbf{S}} \qquad (2.7)$$

Physical meanings of the terms in Eqn. (2.7) are the same as the ones in Eqn. (2.1). However, in Eqn. (2.7) fluxes and surface sources are tensors, whereas volume source is a vector variable. Rearranging Eqn. (2.7) yields the general form of conservation law for a vector quantity.

$$\frac{\partial}{\partial t}\int_{\Omega}\vec{\mathbf{Q}}\;\mathrm{d}\Omega \;+\; \oint_S \vec{\vec{\mathbf{F}}}\cdot \mathrm{d}\vec{\mathbf{S}} \;=\; \int_{\Omega}\vec{\mathbf{Q}}_{\mathbf{V}}\;\mathrm{d}\Omega \;+\; \oint_S \vec{\vec{\mathbf{Q}}}_{\mathbf{S}}\cdot \mathrm{d}\vec{\mathbf{S}} \qquad (2.8)$$

Flux tensor again consists of two parts which are convective part $\vec{\vec{\mathbf{F}}}_{\mathbf{C}}$ and diffusive part $\vec{\vec{\mathbf{F}}}_{\mathbf{D}}$. Convective part of the flux tensor can be given as;

$$\vec{\vec{\mathbf{F}}}_{\mathbf{C}} \;=\; \vec{\mathbf{V}}\otimes\vec{\mathbf{Q}} \qquad (2.9)$$

where $\otimes$ is the tensor product operation [1].

Diffusive part can be given as; [1]

$$\vec{\vec{\mathbf{F}}}_{\mathbf{D_{ij}}} \;=\; -\kappa\rho\frac{\partial u_j}{\partial x_i} \qquad (2.10)$$

Conservation law for scalar quantity, $Q$ in Eqn. (2.8) takes the following form, when Gauss's theorem is applied to surface integrals; [1, 37]

$$\int_{\Omega}\frac{\partial\vec{\mathbf{Q}}}{\partial t}\;\mathrm{d}\Omega \;+\; \int_{\Omega}\vec{\nabla}\cdot\vec{\vec{\mathbf{F}}}\;\mathrm{d}\Omega \;=\; \int_{\Omega}\vec{\mathbf{Q}}_{\mathbf{V}}\;\mathrm{d}\Omega \;+\; \int_{\Omega}\vec{\nabla}\cdot\vec{\vec{\mathbf{Q}}}_{\mathbf{S}}\;\mathrm{d}\Omega \qquad (2.11)$$

In the application of Gauss's theorem, it is assumed that fluxes and surface sources are continuous, and volume $\Omega$ is fixed.

Eqn. (2.11) holds for an arbitrary volume and this yields the differential form which is given in [1] as;

$$\frac{\partial \vec{\mathbf{Q}}}{\partial t} + \vec{\nabla} \cdot \vec{\vec{\mathbf{F}}} = \vec{\mathbf{Q}}_V + \vec{\nabla} \cdot \vec{\vec{\mathbf{Q}}}_S \qquad (2.12)$$

## 2.2 Conservation Laws for Fluid Flow

Conservation laws are also valid for fluid flows like other physical systems and it is possible to obtain equation set that determines dynamical behavior of fluid flow without any other dynamical law. There is also a need to define the nature of the fluid. Laws or formulas defining physical nature of the fluid are added to conservation laws, and this set of equations make it possible to give a complete description of the fluid flow. [1]

### 2.2.1 Conservation of Mass

Physical basis of this equation is the fact that during the dynamic evolution of any physical system except the ones involving nuclear reactions, it is not possible to create or destruct mass. Equation for conservation of mass is obtained by putting density, $\rho$ in place of general scalar property, $Q$ in the scalar conservation equations. In the equation for conservation of mass, diffusive flux term does not exist, because any flux created by density gradients in the fluid involves the transportation of fluid particles which is a convective flux in fact. [1]

Conservation of mass equations in integral form is; [1]

$$\frac{\partial}{\partial t}\int_{\Omega} \rho \, \mathrm{d}\Omega + \oint_S \rho \vec{\mathbf{V}} \cdot \mathrm{d}\vec{\mathbf{S}} = 0 \qquad (2.13)$$

and in the differential form; [1]

$$\frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot \left( \rho \vec{\mathbf{V}} \right) = 0 \tag{2.14}$$

## 2.2.2 Conservation of Momentum

Linear momentum of a physical system is conserved if the net force acting on the system is zero. This fact is a consequence of Newton Laws. [1]

Momentum is a vector quantity and momentum per unit volume, $\rho\vec{\mathbf{V}}$ should be replaced with the general vector quantity $\vec{\mathbf{Q}}$ in the conservation law for vector quantities.

Momentum of a system changes with the action of forces which means that forces constitute the source terms in conservation law. Forces consist of body forces due to the gravity and magnetic effects, and surface forces due to pressure and viscous effects [37].

Volume forces are the body forces and $\vec{\mathbf{Q}}_\mathrm{V}$ term in the conservation law is the sum of body forces per unit volume which is $\rho\vec{\mathbf{f}}_\mathbf{b}$ [37].

$\vec{\vec{\mathbf{Q}}}_\mathrm{S}$ term in conservation law should be replaced with stress tensor, $\vec{\vec{\boldsymbol{\sigma}}}$ which is given in [1] as;

$$\vec{\vec{\boldsymbol{\sigma}}} = -p\vec{\vec{\mathbf{I}}} + \vec{\vec{\boldsymbol{\tau}}} \tag{2.15}$$

First term is the pressure term and the minus sign in front of it is due to the fact that surface vector always points outwards [37]. Second term is the viscous shear stress tensor and given in [1] as;

$$\tau_{ij} \; = \; \mu\left[\left(\partial_i v_j \; + \; \partial_j v_i\right) \; + \; \lambda\left(\vec{\nabla}\cdot\vec{\mathbf{V}}\right)\delta_{ij}\right] \tag{2.16}$$

where $\mu$ is the dynamic viscosity and $\lambda$ is the second viscosity coefficient. By using the Stoke's relation, the second viscosity coefficient is given as;

$$\lambda \; = \; -\frac{2}{3}\mu \tag{2.17}$$

By using all of the information above, conservation of momentum equation in the integral form is; [1]

$$\frac{\partial}{\partial t}\int_{\Omega}\rho\vec{\mathbf{V}}\;\mathrm{d}\Omega \; + \; \oint_S\left(\rho\vec{\mathbf{V}}\right)\vec{\mathbf{V}}\cdot\mathrm{d}\vec{\mathbf{S}} \; = \; \int_{\Omega}\rho\vec{\mathbf{f}}_{\mathbf{b}}\;\mathrm{d}\Omega \; + \; \oint_S\vec{\vec{\boldsymbol{\sigma}}}\cdot\mathrm{d}\vec{\mathbf{S}} \tag{2.17}$$

and in the differential form; [1]

$$\frac{\partial\rho\vec{\mathbf{V}}}{\partial t} \; + \; \vec{\nabla}\cdot\left(\left(\rho\vec{\mathbf{V}}\right)\vec{\mathbf{V}}\right) \; = \; \rho\vec{\mathbf{f}}_{\mathbf{b}} \; + \; \vec{\nabla}\cdot\vec{\vec{\boldsymbol{\sigma}}} \tag{2.18}$$

### 2.2.3 Conservation of Energy

First law of thermodynamics is;

$$\delta E = \delta Q + \delta W \tag{2.19}$$

where $E$ represents energy, $Q$ represents heat transfer, and $W$ represents work.

According to the first law of thermodynamics, in the absence of heat transfer and work, total internal energy of a system is conserved [37].

First law of thermodynamics is given for a closed system; whereas for a physical system involving fluid flows, physical system is an open system. For closed systems, conservation of energy is for internal energy; however, for open systems such as fluid flows, total energy which is composed of the sum of internal energy and kinetic energy is considered for conservation of energy equation. [37]

Total energy per unit mass, $e_t$ is;

$$e_t = e + \frac{V^2}{2} \tag{2.20}$$

where $e$ is internal energy per unit mass and $V^2/2$ is kinetic energy per unit mass.

Convective flux term for conservation of energy equation can be given as; [1]

$$\vec{\mathbf{F}}_{\mathbf{C}} = \rho\vec{\mathbf{V}}\left(e + \frac{V^2}{2}\right) = \rho\vec{\mathbf{V}}e_t \tag{2.20}$$

Diffusive flux is in the following form; [1]

$$\vec{\mathbf{F}}_{\mathbf{D}} = -\gamma\rho\kappa\vec{\nabla}e_t \tag{2.21}$$

where $\kappa$ is thermal diffusivity constant, and $\gamma$ is equal to the ratio of constant pressure specific heat, $c_p$ and constant volume specific heat, $c_v$ . Physical process corresponding to diffusive flux of total energy is diffusion of heat in the fluid flow and it can be given in the form of Fourier's law of heat conduction which is;

$$\vec{\mathbf{F}}_{\mathbf{D}} = -k\vec{\nabla}T \tag{2.22}$$

where $k$ is thermal conductivity and $T$ is the absolute temperature. Thermal conductivity constant can be given as; [1]

$$k = \rho c_p \kappa = \frac{\mu c_p}{\mathrm{Pr}} \tag{2.23}$$

where *Pr* is Prandtl number and equal to;

$$\mathrm{Pr} = \frac{\nu}{\kappa} \tag{2.24}$$

From the first law of thermodynamics, it can be seen that the change of energy amount in a system is caused by heat transfer to or from the system, and work done by or on the system. Source terms for the conservation of energy equation are work done by the forces acting to the control volume and heat transfer other than heat conduction. These sources can be grouped as volume and surface sources.

Work done by the volume forces, in other words body forces, $\vec{\mathbf{f}}_{\mathbf{b}}$ and heat sources other than conduction which are chemical reactions and radiation are the volume sources. Work done by the surface forces are the surface sources. There are no surface heat sources. [1]

Volume sources per unit volume are; [1]

$$\vec{\mathbf{Q}}_{\mathbf{V}} = \rho\vec{\mathbf{f}}_{\mathbf{b}} \cdot \vec{\mathbf{V}} + q_H \tag{2.25}$$

and surface sources, $\vec{\mathbf{Q}}_{\mathbf{S}}$ is; [1]

$$\vec{\mathbf{Q}}_{\mathbf{S}} = \vec{\vec{\boldsymbol{\sigma}}} \cdot \vec{\mathbf{V}} = -p\vec{\mathbf{V}} + \vec{\vec{\boldsymbol{\tau}}} \cdot \vec{\mathbf{V}} \tag{2.26}$$

By using all of the information above, conservation of energy equation in the integral form is; [1]

$$\begin{aligned} &\frac{\partial}{\partial t}\int_{\Omega} \rho e_t \, \mathrm{d}\Omega + \oint_S (\rho e_t) \vec{\mathbf{V}} \cdot \mathrm{d}\vec{\mathbf{S}} \\ &= \oint_S k\vec{\nabla}T \cdot \mathrm{d}\vec{\mathbf{S}} + \int_{\Omega} \left(\rho\vec{\mathbf{f}}_{\mathbf{b}} \cdot \vec{\mathbf{V}} + q_H\right) \mathrm{d}\Omega + \oint_S \left(\vec{\vec{\boldsymbol{\sigma}}} \cdot \vec{\mathbf{V}}\right) \mathrm{d}\vec{\mathbf{S}} \end{aligned} \tag{2.27}$$

and in the differential form; [1]

$$\frac{\partial \rho e_t}{\partial t} + \vec{\nabla} \cdot \left(\rho e_t \vec{\mathbf{V}}\right) = \vec{\nabla} \cdot \left(k\vec{\nabla}T\right) + \left(\rho\vec{\mathbf{f}}_{\mathbf{b}} \cdot \vec{\mathbf{V}} + q_H\right) + \vec{\nabla} \cdot \left(\vec{\vec{\boldsymbol{\sigma}}} \cdot \vec{\mathbf{V}}\right) \tag{2.28}$$

## 2.3 Analysis of Conservation Equations – Ideal Gas Law

Up to now, three conservation equations for a fluid flow are obtained. These equations are conservation of mass, conservation of momentum and conservation of energy. Two of these equations involve the conservation of scalar variables that are density and total energy. Third conserved quantity is momentum which is a vector variable having three components. This means that one can obtain three conservation equations for three components of momentum from conservation of momentum equation. As a result, it is possible to come up with five conservation equations for scalar variables at the end. [37]

If the physical variables in these five equations are investigated, it can be seen that there are three thermodynamic variables which are density, $\rho$; pressure, $p$; and internal energy, $e$ appear in these equations, and there is the velocity which is a kinematical variable having three components; $u$, $v$, $w$. At the end, there are six basic variables which should be defined in space and time for a complete description of the fluid flow. [37]

There are five equations to determine six flow variables. There is a need for additional equations coming from nature of the fluid. From thermodynamics, it is known that to define a local thermodynamic state, two independent thermodynamic variables should be determined. In the case of fluid flow, if these variables are chosen as density, $\rho$ and internal energy, $e$; then the third thermodynamic variable pressure, $p$ should be determined in terms of $\rho$ and $e$ with an equation of the form $p = p(\rho, e)$ . [37]

For a calorically perfect gas, it is known that there is a relation between absolute temperature and internal energy such that;

$$e = c_v T \tag{2.29}$$

where constant volume specific heat, $c_v$ can be taken as constant for a thermally perfect gas. [37]

A relation between absolute temperature, density and pressure is needed. In this study, compressible flows are investigated and for compressible flows ideal gas law establishes this relation as;

$$p = \rho R T \tag{2.30}$$

where $R$ is ideal gas constant for compressible fluid considered. [37]

Using the facts that $c_v = R/(\gamma - 1)$ and $\gamma = c_p / c_v$, in addition to Equations (2.29) and (2.30), it is possible to obtain the following relation between density, $\rho$; internal energy, $e$; and pressure, $p$; [37]

$$p = (\gamma - 1)\rho e \tag{2.31}$$

However, conservation of energy equation for a fluid flow is established on total energy per unit volume instead of internal energy. Thus, it is more useful to have a relation involving total energy per unit volume rather than internal energy. The relation between total energy per unit volume, density and pressure is obtained by using Equations (2.20) and (2.31); [37]

$$p = (\gamma - 1)\rho\left(e_t - \frac{V^2}{2}\right) \tag{2.32}$$

Lastly, dynamic viscosity, $\mu$ is a physical quantity which is largely depends on temperature and for both gases and liquids, while the pressure dependence is small. In this study, dynamic viscosity, $\mu$ is considered as only depends on temperature and this dependence in the form of $\mu = \mu(T)$ is taken as the following;

$$\mu = \frac{1.45\, T^{3/2}}{T + 110} \cdot 10^{-6} \tag{2.33}$$

which is Sutherland's formula that is generally used for gases like air. $T$ in the equation is temperature in units of Kelvin. [1]

# CHAPTER 3

# NUMERICAL IMPLEMENTATION

## 3.1 Discretization of Governing Equations

Before discretization of governing equations, these equations should be put in a convenient form for discretization and numerical analysis.

In this study, three dimensional, compressible flow of gases is investigated under the assumptions of no body forces and no volume heat sources. Besides these, conservation of momentum equation can be represented with three conservation equations of scalar variables for *x*-component of momentum, *y*-component of momentum and *z*-component of momentum. Under the above flow properties and assumptions, these five conservation equations are given below;

$$\frac{\partial}{\partial t}\int_{\Omega}\rho \; \mathrm{d}\Omega + \oint_{S}\rho\vec{\mathbf{V}}\cdot\mathrm{d}\vec{\mathbf{S}} = 0 \tag{3.1}$$

$$\frac{\partial}{\partial t}\int_{\Omega}(\rho u) \; \mathrm{d}\Omega + \oint_{S}(\rho u)\vec{\mathbf{V}}\cdot\mathrm{d}\vec{\mathbf{S}} = \oint_{S}\left(\vec{\vec{\boldsymbol{\sigma}}}\cdot\hat{\mathbf{i}}\right)\cdot\mathrm{d}\vec{\mathbf{S}} \tag{3.2}$$

$$\frac{\partial}{\partial t}\int_{\Omega}(\rho v) \; \mathrm{d}\Omega + \oint_{S}(\rho v)\vec{\mathbf{V}}\cdot\mathrm{d}\vec{\mathbf{S}} = \oint_{S}\left(\vec{\vec{\boldsymbol{\sigma}}}\cdot\hat{\mathbf{j}}\right)\cdot\mathrm{d}\vec{\mathbf{S}} \tag{3.3}$$

$$\frac{\partial}{\partial t}\int_{\Omega}(\rho w)\ \mathrm{d}\Omega \ + \ \oint_S (\rho w)\vec{\mathbf{V}}\cdot \mathrm{d}\vec{\mathbf{S}} \ = \ \oint_S \left(\vec{\vec{\boldsymbol{\sigma}}}\cdot\hat{\mathbf{k}}\right)\cdot \mathrm{d}\vec{\mathbf{S}} \tag{3.4}$$

$$\frac{\partial}{\partial t}\int_{\Omega}\rho e_t\ \mathrm{d}\Omega \ + \ \oint_S (\rho e_t)\vec{\mathbf{V}}\cdot \mathrm{d}\vec{\mathbf{S}} \ = \ \oint_S \left[\left(\vec{\vec{\boldsymbol{\sigma}}}\cdot\vec{\mathbf{V}}\right)+k\vec{\nabla}T\right]\cdot \mathrm{d}\vec{\mathbf{S}} \tag{3.5}$$

The equations above can be written in a compact way which is vector form;

$$\frac{\partial}{\partial t}\int_{\Omega}\mathbf{Q}\ \mathrm{d}\Omega \ + \ \oint_S \vec{\mathbf{F}}\cdot \mathrm{d}\vec{\mathbf{S}} \ = \ \oint_S \vec{\mathbf{G}}\cdot \mathrm{d}\vec{\mathbf{S}} \tag{3.6}$$

where $\mathbf{Q}$ is the vector of conservative variables, $\vec{\mathbf{F}}$ involves convective fluxes and $\vec{\mathbf{G}}$ involves source terms [37, 38]. These vectors are given as;

$$\mathbf{Q} \ = \ \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \end{bmatrix} \tag{3.7}$$

$$\vec{\mathbf{F}} \ = \ \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \end{bmatrix} \vec{\mathbf{V}} \tag{3.8}$$

$$\vec{\mathbf{G}} = \begin{bmatrix} 0 \\ \vec{\vec{\boldsymbol{\sigma}}} \cdot \hat{\mathbf{i}} \\ \vec{\vec{\boldsymbol{\sigma}}} \cdot \hat{\mathbf{j}} \\ \vec{\vec{\boldsymbol{\sigma}}} \cdot \hat{\mathbf{k}} \\ \vec{\vec{\boldsymbol{\sigma}}} \cdot \vec{\mathbf{V}} + k\vec{\nabla} T \end{bmatrix} \tag{3.9}$$

Using Eqn. (2.15) in Eqn. (3.9) yields;

$$\vec{\mathbf{G}} = -\begin{bmatrix} 0 \\ \hat{\mathbf{i}} \\ \hat{\mathbf{j}} \\ \hat{\mathbf{k}} \\ \vec{\mathbf{V}} \end{bmatrix} p + \begin{bmatrix} 0 \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \hat{\mathbf{i}} \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \hat{\mathbf{j}} \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \hat{\mathbf{k}} \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \vec{\mathbf{V}} + k\vec{\nabla} T \end{bmatrix} \tag{3.10}$$

Combining pressure part of $\vec{\mathbf{G}}$ vector with convective fluxes, $\vec{\mathbf{F}}$ is a customary approach. Remaining part of $\vec{\mathbf{G}}$ vector involves only viscous fluxes. The $\vec{\mathbf{F}}$ and $\vec{\mathbf{G}}$ vectors after this operation; [37]

$$\vec{\mathbf{F}} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \end{bmatrix} \vec{\mathbf{V}} + \begin{bmatrix} 0 \\ \hat{\mathbf{i}} \\ \hat{\mathbf{j}} \\ \hat{\mathbf{k}} \\ \vec{\mathbf{V}} \end{bmatrix} p \tag{3.11}$$

$$\vec{\mathbf{G}} = \begin{bmatrix} 0 \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \hat{\mathbf{i}} \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \hat{\mathbf{j}} \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \hat{\mathbf{k}} \\ \vec{\vec{\boldsymbol{\tau}}} \cdot \vec{\mathbf{V}} + k\vec{\nabla} T \end{bmatrix} \tag{3.12}$$

If the components of $\vec{\mathbf{G}}$ vector in *x*, *y* and *z* coordinates are written explicitly; [1]

$$\mathbf{G}_x = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ \left(\tau_{xx} u + \tau_{xy} v + \tau_{xz} w\right) + k\dfrac{\partial T}{\partial x} \end{bmatrix} \tag{3.13}$$

$$\mathbf{G}_y = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ \left(\tau_{xy} u + \tau_{yy} v + \tau_{yz} w\right) + k\dfrac{\partial T}{\partial y} \end{bmatrix} \tag{3.14}$$

$$\mathbf{G}_z = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ \left(\tau_{xz} u + \tau_{yz} v + \tau_{zz} w\right) + k\dfrac{\partial T}{\partial z} \end{bmatrix} \tag{3.15}$$

Shear stress terms in $\vec{\mathbf{G}}$ vector have the following form; [1]

$$\tau_{xx} = -\frac{2}{3}\mu_{tot}\left(\vec{\nabla}\cdot\vec{\mathbf{V}}\right) + 2\mu_{tot}\frac{\partial u}{\partial x} \tag{3.16}$$

$$\tau_{yy} = -\frac{2}{3}\mu_{tot}\left(\vec{\nabla}\cdot\vec{\mathbf{V}}\right) + 2\mu_{tot}\frac{\partial v}{\partial y} \tag{3.17}$$

$$\tau_{zz} = -\frac{2}{3}\mu_{tot}\left(\vec{\nabla}\cdot\vec{\mathbf{V}}\right) + 2\mu_{tot}\frac{\partial w}{\partial z} \tag{3.18}$$

$$\tau_{xy} = \mu_{tot}\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \tag{3.19}$$

$$\tau_{xz} = \mu_{tot}\left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right) \tag{3.20}$$

$$\tau_{yz} = \mu_{tot}\left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\right) \tag{3.21}$$

Total dynamic viscosity, $\mu_{tot}$ in shear stress terms consists of two parts which are laminar and turbulent and is given as; [4]

$$\mu_{tot} = \mu_{lam} + \mu_{tur} \tag{3.22}$$

Laminar part, $\mu_{lam}$ is calculated by using Sutherland's formula given in Section 2.6. Turbulent part, $\mu_{tur}$ is calculated by using Baldwin-Lomax turbulence model that is discussed in Section 3.4.

Numerical calculation of convective fluxes, $\vec{\mathbf{F}}$ are done by using Roe flux difference splitting method and this method is discussed in Section 3.2.

Numerical calculation of viscous fluxes, $\vec{\mathbf{G}}$ are discussed in Section 3.3.

### 3.1.1 Spatial Discretization

For spatial discretization, cell centered finite volume method is used. In this study, structured meshes are used. Basic area element in structured meshes is "quadrilateral" and basic volume element is "hexahedral" [1]. In cell-centered approach, $\vec{\mathbf{Q}}$ values are defined at the centroid of elements and considered as an average value of the cell [37].

Eqn. (3.6) takes the following spatially discretized form by doing integration operations over volume, Ω and boundaries, *S* of hexahedral cell given in Fig. 3.1. [37]

$$\Omega_{i,j,k}\left(\frac{\partial \mathbf{Q}}{\partial t}\right)_{i,j,k} + \sum_{n=1}^{6}\vec{\mathbf{F}}_{(i,j,k),n}\cdot\vec{\mathbf{S}}_{(i,j,k),n} = \sum_{n=1}^{6}\vec{\mathbf{G}}_{(i,j,k),n}\cdot\vec{\mathbf{S}}_{(i,j,k),n} \qquad (3.23)$$

In the integration operations, it is assumed that convective fluxes and viscous fluxes are constant on surfaces, and time variation of conserved variables in the cell is the same as the time variation of conserved variables at the centroid.

In discretized equation and in Fig. 3.1, (*i*, *j*, *k)* indices represents cell number, and *n* index represents the surface number. Surface area vectors of back and front surfaces of the hexahedral are not shown in Fig. 3.1 for clarity. The surface numbers of back and front surfaces are 5 and 6, respectively. Capital indices in Fig. 3.1 show the corner nodes of hexahedral.

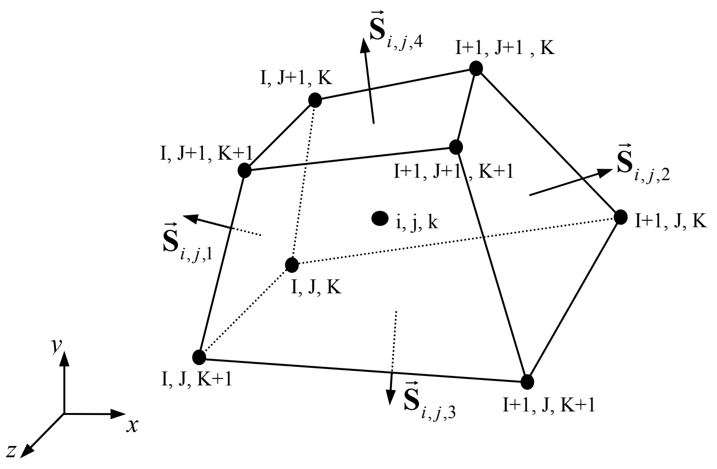


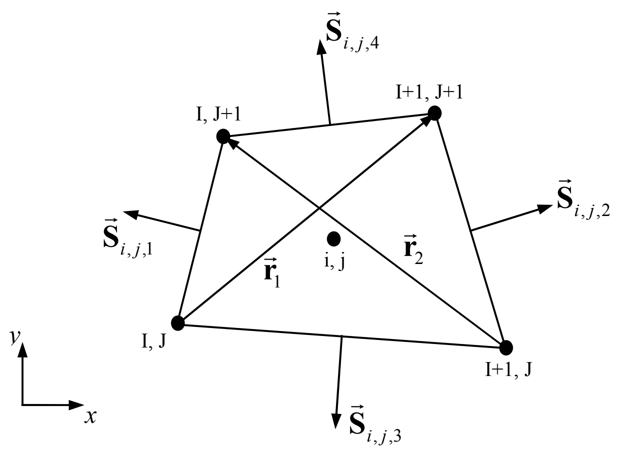
**Figure 3.1:** Indexing of corner nodes and surfaces for hexahedral cell $(i, j, k)$

Volume, $\Omega_{i,j,k}$ of a hexahedral cell can be found by dividing the hexahedral into three pyramids having one corner of the cell as the vertex point. Surface area vectors of hexahedral can be found by taking the cross product of diagonal vectors of the surface. Cross product of diagonal vectors of the surface gives two times of the surface area vector. These diagonal vectors are taken such that their cross product directs towards the outside of the cell volume. Details of volume calculation for a hexahedral are given in Appendix A.

In two dimensional case, discretized equation takes the following form for the quadrilateral cell in Fig. 3.2; [37]

$$A_{i,j}\left(\frac{\partial \mathbf{Q}}{\partial t}\right)_{i,j} + \sum_{n=1}^{4}\vec{\mathbf{F}}_{(i,j),n}\cdot\vec{\mathbf{S}}_{(i,j),n} = \sum_{k=1}^{4}\vec{\mathbf{G}}_{(i,j),n}\cdot\vec{\mathbf{S}}_{(i,j),n} \qquad (3.24)$$

where $\vec{\mathbf{S}}_{(i,j),n}$ vectors represent edge vectors in this case.



$\vec{\mathbf{S}}_{i,j,4}$
I+1, J+1
I, J+1
$\vec{\mathbf{S}}_{i,j,2}$
$\vec{\mathbf{S}}_{i,j,1}$
i, j
$\vec{\mathbf{r}}_2$
$\vec{\mathbf{r}}_1$
$y$
I, J
I+1, J
$x$
$\vec{\mathbf{S}}_{i,j,3}$

**Figure 3.2:** Indexing of corner nodes and edges for quadrilateral cell ($i, j$)

Area, $A_{i,j}$ of the quadrilateral cell can be found by utilizing the cross product of diagonals which is given in the following equation;

$$A_{i,j} = \frac{1}{2}\left(\vec{\mathbf{r}}_1 \times \vec{\mathbf{r}}_2\right) \qquad (3.25)$$

Edge vectors, $\vec{\mathbf{S}}$ in Fig. 3.2 is calculated firstly, by finding the $\Delta x = x_2 - x_1$ and $\Delta y = y_2 - y_1$ differences where index *1* denotes the first node of an edge in counter-clock wise rotation and index *2* denotes the second node in counter-clockwise rotation. Next, these differences can be assigned as components of $\vec{\mathbf{S}}$ vector in the following way; [37]

$$\vec{\mathbf{S}} = \Delta y \hat{\mathbf{i}} - \Delta x \hat{\mathbf{j}} \tag{3.26}$$

### 3.1.2 Temporal Discretization

Spatially discretized form of conservation equations can be written in the following form; [37]

$$\Omega_{i,j,k} \left( \frac{\partial \mathbf{Q}}{\partial t} \right)_{i,j,k} + \vec{\mathbf{R}}_{i,j,k} = 0 \tag{3.27}$$

by defining $\vec{\mathbf{R}}_{i,j,k}$ which is called the residual term as follows; [37]

$$\vec{\mathbf{R}}_{i,j,k} = \sum_{n=1}^{6} \vec{\mathbf{F}}_{(i,j,k),n} \cdot \vec{\mathbf{S}}_{(i,j,k),n} - \sum_{n=1}^{6} \vec{\mathbf{G}}_{(i,j,k),n} \cdot \vec{\mathbf{S}}_{(i,j,k),n} \tag{3.28}$$

When Eqn. (3.28) is discretized in time, it takes the following form;

$$\frac{\mathbf{Q}_{i,j,k}^{n+1} - \mathbf{Q}_{i,j,k}^{n}}{\Delta t} + \frac{1}{\Omega_{i,j,k}} \vec{\mathbf{R}}_{i,j,k} = 0 \tag{3.29}$$

where $\mathbf{Q}_{i,j,k}^{n}$ represents vector of conservative variables of cell ($i, j, k$) at time step $n$. [37]

In this study, explicit time integration is used. $\vec{\mathbf{R}}_{i,j,k}$ is calculated by using conserved variables of current time step *n*, $\mathbf{Q}^{n}_{i,j,k}$ in order to find conserved variables of following time step *n+1*, $\mathbf{Q}^{n+1}_{i,j,k}$ and denoted as $\vec{\mathbf{R}}^{n}_{i,j,k}$. This relation can be given as a functional relation in the following way; [37]

$$\vec{\mathbf{R}}^{n}_{i,j,k} = \vec{\mathbf{R}}^{n}_{i,j,k}\left(\mathbf{Q}^{n}_{i,j,k}\right) \tag{3.30}$$

By rearranging Eqn. (3.20), $\mathbf{Q}^{n+1}_{i,j,k}$ is; [37]

$$\mathbf{Q}^{n+1}_{i,j,k} = \mathbf{Q}^{n}_{i,j,k} - \frac{\Delta t}{\Omega_{i,j,k}}\vec{\mathbf{R}}_{i,j,k} \tag{3.31}$$

Eqn. (3.31) is solved by 3rd order Runge-Kutta method whose formulation is taken from [37]. As stated in [4], this method is developed by Jameson et. al. in [15]. Solution steps are the following;

$$\begin{aligned}
\mathbf{Q}^{(0)}_{i,j,k} &= \mathbf{Q}^{n}_{i,j,k} \\
\mathbf{Q}^{(1)}_{i,j,k} &= \mathbf{Q}^{(0)}_{i,j,k} - \alpha_1 \frac{\Delta t}{\Omega_{i,j,k}}\vec{\mathbf{R}}_{i,j,k}\left(\mathbf{Q}^{(0)}\right) \\
\mathbf{Q}^{(2)}_{i,j,k} &= \mathbf{Q}^{(0)}_{i,j,k} - \alpha_2 \frac{\Delta t}{\Omega_{i,j,k}}\vec{\mathbf{R}}_{i,j,k}\left(\mathbf{Q}^{(0)}\right) \\
\mathbf{Q}^{(3)}_{i,j,k} &= \mathbf{Q}^{(0)}_{i,j,k} - \alpha_3 \frac{\Delta t}{\Omega_{i,j,k}}\vec{\mathbf{R}}_{i,j,k}\left(\mathbf{Q}^{(0)}\right) \\
\mathbf{Q}^{n+1}_{i,j,k} &= \mathbf{Q}^{(3)}_{i,j,k}
\end{aligned} \tag{3.32}$$

where $\alpha$ constants are given as;

$$\begin{aligned} \alpha_1 &= 1/3 \\ \alpha_2 &= 1/2 \\ \alpha_3 &= 1 \end{aligned} \qquad (3.33)$$

In Eqn. (3.32), $\mathbf{Q}_{i,j,k}^{(k)}$ terms where $k$ takes the values of 0, 1, 2, 3; denotes the value of $\mathbf{Q}_{i,j,k}^{n}$ at the $k^{th}$ step of Runge-Kutta method.

## 3.2 Roe Flux Difference Splitting Method

In this study, calculation of convective fluxes on cell edges,

$$\sum_{k=1}^{4} \vec{\mathbf{F}}_{i,j,k} \cdot \vec{\mathbf{S}}_{i,j,k} \qquad (3.34)$$

is done by Roe flux difference splitting method.

Eqn. (3.34) takes the following form after the scalar product between the vectors is taken and Eqn. (3.26) is applied;

$$\sum_{k=1}^{4} \vec{\mathbf{F}}_{i,j,k} \cdot \vec{\mathbf{S}}_{i,j,k} = \sum_{k=1}^{4} \left( F_{x,k} \Delta y_k - F_{y,k} \Delta x_k \right)_{i,j} \qquad (3.35)$$

Length of an edge, normal and tangential velocities on this edge can be defined as; [35]

$$\Delta s = \sqrt{(\Delta x)^2 + (\Delta y)^2} \qquad (3.36)$$

$$u_n = \frac{\left(u\Delta y - v\Delta x\right)}{\Delta s} \tag{3.37}$$

$$v_n = \frac{\left(u\Delta x + v\Delta y\right)}{\Delta s} \tag{3.38}$$

Convective flux through an edge can be written as; [35]

$$\left(F_x\Delta y - F_y\Delta x\right) = \begin{pmatrix} \rho u_n \\ \rho u_n u + p\dfrac{\Delta y}{\Delta s} \\ \rho u_n - p\dfrac{\Delta x}{\Delta s} \\ \rho u_n h_t \end{pmatrix}\Delta s = \mathbf{\Phi}\Delta s \tag{3.39}$$

where $h_t$ term denotes total enthalpy and given as;

$$h_t = e_t + \frac{p}{\rho} \tag{3.40}$$

In order to calculate the edge flux, values of conservative variables on that edge is needed, and these values are determined by using conservative variables at cells at the left and right of the edge. By using Roe's approximate Riemann solver, edge flux given as; [35]

$$\mathbf{\Phi}\left(\mathbf{Q}_L,\mathbf{Q}_r\right) = \frac{1}{2}\left[\mathbf{\Phi}\left(\mathbf{Q}_L\right) + \mathbf{\Phi}\left(\mathbf{Q}_R\right)\right] - \frac{1}{2}\sum_{n=1}^{4}\left|\hat{\mathbf{a}}_n\right|^{*}\mathbf{\Delta V}_n\hat{\mathbf{R}}_n \tag{3.41}$$

where

$$\mathbf{a} = \begin{pmatrix} \hat{u}_n - \hat{c} \\ \hat{u}_n \\ \hat{u}_n \\ \hat{u}_n + \hat{c} \end{pmatrix} \tag{3.42}$$

$$\mathbf{\Delta V} = \begin{pmatrix} \dfrac{\Delta p - \hat{p}\hat{c}\Delta u_n}{2\hat{c}^2} \\ \dfrac{\hat{\rho}\Delta u_t}{\hat{c}} \\ \Delta\rho - \dfrac{\Delta p}{\hat{c}^2} \\ \dfrac{\Delta p + \hat{p}\hat{c}\Delta u_n}{2\hat{c}^2} \end{pmatrix} \tag{3.43}$$

$$\hat{\mathbf{R}} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ \hat{u} - \hat{c}\dfrac{\Delta y}{\Delta s} & \hat{c}\dfrac{\Delta x}{\Delta s} & \hat{u} & \hat{u} + \hat{c}\dfrac{\Delta y}{\Delta s} \\ \hat{u} + \hat{c}\dfrac{\Delta x}{\Delta s} & \hat{c}\dfrac{\Delta y}{\Delta s} & \hat{v} & \hat{u} - \hat{c}\dfrac{\Delta x}{\Delta s} \\ \hat{h}_t - \hat{u}_n\hat{c} & \hat{u}_t\hat{c} & \dfrac{\hat{u}^2 + \hat{v}^2}{2} & \hat{h}_t - \hat{u}_n\hat{c} \end{pmatrix} \tag{3.44}$$

The variables having cap on it are defined as the following;

$$\begin{aligned} \hat{\rho} &= \sqrt{\rho_L \rho_R} \\ \hat{u} &= \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\ \hat{v} &= \frac{\sqrt{\rho_L}v_L + \sqrt{\rho_R}v_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\ \hat{h}_t &= \frac{\sqrt{\rho_L}\left(h_t\right)_L + \sqrt{\rho_R}\left(h_t\right)_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \end{aligned} \tag{3.45}$$

The other variables having cap which are $\hat{c}$, $\hat{u}_n$ and $\hat{u}_t$ can be found by using the variables in Eqn. (3.45).

As De Zeeuw and Powell stated in [35], to prevent expansion shocks entropy fix is applied. Smoothed values, $\left|\hat{\mathbf{a}}^{(k)}\right|^*$ are defined instead of $\left|\hat{\mathbf{a}}^{(k)}\right|$ for acoustic waves given with $k = 1$ and $k = 4$ terms in the following way; [35]

$$\left|\hat{\mathbf{a}}^{(k)}\right|^* = \begin{cases} \left|\hat{\mathbf{a}}^{(k)}\right|, & \left|\hat{\mathbf{a}}^{(k)}\right| \geq \frac{1}{2}\delta\mathbf{a}^{(k)} \\ \frac{\left(\hat{\mathbf{a}}^{(k)}\right)^2}{\delta\mathbf{a}^{(k)}} + \frac{1}{4}\delta\mathbf{a}^{(k)}, & \left|\hat{\mathbf{a}}^{(k)}\right| \leq \frac{1}{2}\delta\mathbf{a}^{(k)} \end{cases}$$
$$\delta\mathbf{a}^{(k)} = \max\left(4\Delta\mathbf{a}^{(k)}, 0\right), \quad \Delta\mathbf{a}^{(k)} = \mathbf{a}_R^{(k)} - \mathbf{a}_L^{(k)} \tag{3.46}$$

Convective fluxes in three-dimensions are calculated by using the same formulation used in two-dimensions. Local coordinates are rotated in such a way that $z$-direction is perpendicular to the plane formed by velocity vectors of left and right cell. Afterwards, convective fluxes are calculated on this plane where $z$ component of velocities disappears. Finally, calculated fluxes are rotated to global coordinates. Details of this calculation are given in Appendix B.

## 3.3 Viscous Fluxes

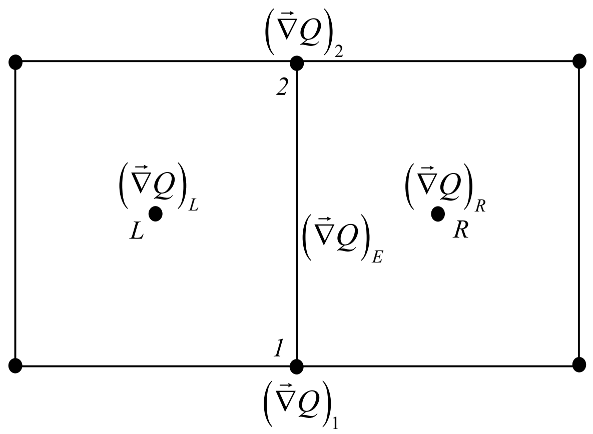
In order to calculate viscous fluxes, gradients of velocity components, $u$, $v$ and gradient of temperature, $T$ on the edges of a cell are needed. Gradient values on an edge can be calculated in two methods based on the idea of "central discretization of viscous terms [1]". In the first method, gradient values on an edge are found by taking arithmetic average of gradient values at the nodes which are located at both ends of the edge. By using the variables shown in Fig. 3.3, this method can be represented as;

$$\left(\vec{\nabla}Q\right)_E = \frac{\left(\vec{\nabla}Q\right)_1 + \left(\vec{\nabla}Q\right)_2}{2} \tag{3.47}$$

where $Q$ represents a general scalar variable.

Second method of finding gradient values on an edge is taking arithmetic average of gradient values at the cell centers which are located at both sides of the edge. By using the variables shown in Fig. 3.3, this method can be represented as;

$$\left(\vec{\nabla}Q\right)_E = \frac{\left(\vec{\nabla}Q\right)_L + \left(\vec{\nabla}Q\right)_R}{2} \tag{3.48}$$



$\left(\vec{\nabla}Q\right)_2$
2
$\left(\vec{\nabla}Q\right)_L$
L
$\left(\vec{\nabla}Q\right)_E$
$\left(\vec{\nabla}Q\right)_R$
R
1
$\left(\vec{\nabla}Q\right)_1$

**Figure 3.3:** Representations of gradients at cell centers, nodes and edges

In this study, the first method for finding gradients on an edge are used, and gradient values at the nodes which are located at both ends of the edge are needed. These gradient values at the nodes are calculated by using the general idea given in [37] which employs the divergence theorem which is given as;

$$\int_{\Omega} \vec{\nabla} Q \, \mathrm{d}\Omega = \oint_{S} Q \, \mathrm{d}\vec{\mathbf{S}} \tag{3.49}$$

If divergence theorem is applied to a volume having a volume Ω under the assumption that gradient of variable $Q$ is constant in the volume, it takes the following form [37];

$$\vec{\nabla} Q = \frac{1}{\Omega} \oint_{S} Q \, \mathrm{d}\vec{\mathbf{S}} \tag{3.50}$$

Eqn. (3.50) is applied to cells surrounding nodes and having corners of four cell centers as shown with dashed lines in Fig. 3.4. Although this approach needs surface vector calculations of newly defined cells around nodes, it removes the problem of reconstructing the conservative variables at the nodes by using corresponding conservative variables at cell centers around a node, and corresponding accuracy loss for reconstruction.

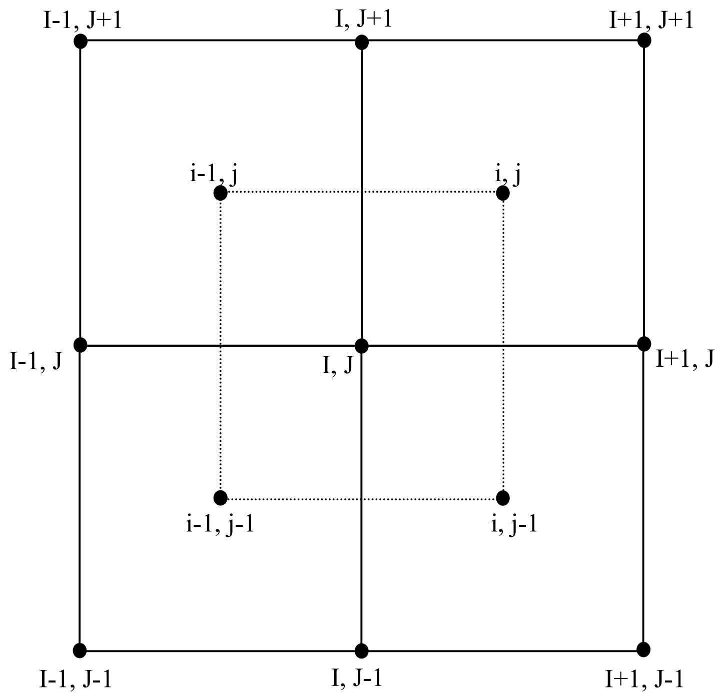


**Figure 3.4:** Representation of cell (*I*, *J*) around node (*I*, *J*)

For the newly defined cell of having area $A_{I,J}$ around node (*I*, *J*), using Eqn. (3.26) yields the following equation [37];

$$\vec{\nabla} Q_{I,J} = \frac{1}{A_{I,J}} \sum_{K=1}^{4} Q_K \left( \Delta y_{I,J} \hat{\mathbf{i}} - \Delta x_{I,J} \hat{\mathbf{j}} \right) \tag{3.51}$$

$Q_K$ term denotes the value of the variable on the edge of *newly* defined cell around the node, and it is calculated by taking the arithmetic average of corresponding variable values at the cell centers which are located at both ends of the edge of *newly* defined cell surrounding the node. Applying the definition of gradient for Eqn. (3.51) yields [37];

$$\left(\frac{\partial Q}{\partial x}\right)_{I,J} \hat{\mathbf{i}} + \left(\frac{\partial Q}{\partial y}\right)_{I,J} \hat{\mathbf{j}} = \frac{1}{A_{I,J}} \sum_{K=1}^{4} Q_K \left(\Delta y_{I,J} \hat{\mathbf{i}} - \Delta x_{I,J} \hat{\mathbf{j}}\right) \qquad (3.52)$$

Equating *x* and *y* components of the vectors in each side of the equality yields; [37]

$$\begin{aligned}\left(\frac{\partial Q}{\partial x}\right)_{I,J} &= \frac{1}{A_{I,J}} \sum_{K=1}^{4} Q_K \Delta y_{I,J} \\ \left(\frac{\partial Q}{\partial y}\right)_{I,J} &= -\frac{1}{A_{I,J}} \sum_{K=1}^{4} Q_K \Delta x_{I,J}\end{aligned} \qquad (3.53)$$

Applying Eqn. (3.53) for components of velocity, *u*, *v* and temperature, *T*; partial derivatives of these variables, which are needed in viscous flux calculations, can be found.

## 3.4 Baldwin-Lomax Turbulence Model

Baldwin-Lomax Turbulence Model is an algebraic, two-layer model. In two layer models, eddy viscosity is given as;

$$\mu_{tur} = \begin{cases} \left(\mu_{tur}\right)_{inner}, & y \le y_m \\ \left(\mu_{tur}\right)_{outer}, & y \ge y_m \end{cases} \qquad (3.54)$$

where $y_m$ is the smallest value of *y* at which the equality $\left(\mu_{tur}\right)_{inner} = \left(\mu_{tur}\right)_{outer}$ equality holds. [25]

The eddy viscosities in inner layer and outer layer are given by; [25]

$$(\mu_{tur})_{inner} = \rho\omega\left[(0.4)(y - y_0)\left(1 - e^{-\frac{y^+}{26}}\right)\right]^2 \qquad (3.55)$$

and

$$(\mu_{tur})_{outer} = \rho\, K\, C_{cp}\, F_{wake}\, F_{Kleb} \qquad (3.56)$$

respectively. In inner eddy viscosity equation, *ω* represents the magnitude of vorticity vector and given as;

$$\omega = \sqrt{\left(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}\right)^2} \qquad (3.57)$$

In the above equations, $y_0$ denotes *y* coordinate of the wall in the profile, and this implies that $(y - y_0)$ is the distance from the wall for the point in the profile. *y+* denotes the dimensionless distance and defined as;

$$y^+ = \frac{\rho}{\mu_{lam}}\,(y - y_0)\, u_\tau \qquad (3.58)$$

where $u_\tau$ is friction velocity, and it is a velocity scale representing velocities near the solid boundary that is defined as; [25]

$$u_\tau = \sqrt{\frac{\tau_w}{\rho}} \qquad (3.59)$$

Constants $K$ and $C_{cp}$ in the outer viscosity equation are given as;

$$K = 0.0168 \qquad C_{cp} = 1.6$$

$F_{wake}$ in Eqn. (3.56) is given as;

$$F_{wake} = \min\left[\left(y_{\max} - y_0\right) F_{\max};\ \frac{C_{wk}\left(y_{\max} - y_0\right) V_{dif}^2}{F_{\max}}\right] \tag{3.60}$$

where $F_{max}$ is the maximum value of $F(y)$ given as; [36]

$$F(y) = \left(y - y_0\right) \omega \left(1 - e^{-\frac{y^+}{26}}\right) \tag{3.61}$$

and $y_{max}$ is the distance from the wall where $F_{max}$ occurs.

In the wake regions, $F(y)$ takes the following form; [36]

$$F(y) = \left(y - y_0\right) \omega \tag{3.62}$$

and $y_{max}$ is now the distance from the wake cut where $F_{max}$ occurs.

The $V_{dif}$ term in Eqution (3.60) is defined as the maximum value of $V$ for ***boundary layers***. For free shear layers such as ***wake region***, $V_{dif}$ is given as; [25]

$$V_{dif} = \left(\sqrt{u^2 + v^2}\right)_{\max} - \left(\sqrt{u^2 + v^2}\right)_{y=y_{\max}} \tag{3.63}$$

The last term in Eqn. (3.56) which is $F_{Kleb}$ represents Klebanoff's intermittency function and defined as;

$$F_{Kleb} = \left[ 1 + 5.5 \left( C_{Kleb} \frac{y - y_0}{y_{\max} - y_0} \right)^6 \right]^{-1} \qquad (3.64)$$

where $C_{Kleb}$ is equal to 0.3 . [25]

## 3.5 Boundary Conditions

Boundary conditions are applied by using the concept of *ghost* cells. One row of cells is defined all around the computational domain. The only geometric variables attached to these cells are coordinates of cell centers, and these are calculated by taking mirror image of the cell center of the corresponding inner domain cell about the boundary. Cell center coordinates are determined due to the need for these coordinates in calculations of velocity and temperature gradients at the nodes.

Different boundary conditions are applied by same way of determining conservative variables at the ghost cells.

### 3.5.1 Solid Wall Boundary

Application of solid wall boundary condition involves different features for the system of Navier-Stokes equations which governs viscous flows, and for the system of Euler equations which governs inviscid flows.

#### 3.5.1.1 Inviscid Flows

For inviscid flows, flow should be tangent to the solid wall. This can be achieved by taking mirror image of the velocity about the boundary. The tangential velocity component of the boundary cell is equated to the tangential velocity component of the corresponding ghost cell, and normal velocity component of the boundary cell is assigned as the normal velocity component of the ghost cell with an *opposite sign*. Density and energy variables at the ghost cell are determined by directly assigning the corresponding variables of the boundary cell. [37]

#### 3.5.1.2 Viscous Flows

For viscous flows, flow velocity should be zero on the solid wall. This boundary condition is called no-slip condition. This is achieved by assigning tangential and normal velocity components of the boundary cell to the corresponding ghost cell velocity components having the *opposite sign*. Density and energy variables at the ghost cell are determined by directly assigning the corresponding variables of the boundary cell as in the inviscid case. [37]

### 3.5.2 Far-Field Boundary

"In external as well as internal flow problems the inlet and outlet boundaries are assumed to be located far enough from the main flow region so that the influence of the flow disturbances do not affect the free-stream values" [1]. In far-field boundary condition, flow variables of the ghost cell which are density, velocity and energy are determined by directly assigning corresponding free-stream values.

## 3.6 Implementation of Parallel Processing

In this study, parallel processing is done on a Linux cluster. Linux clusters belong to Massively Parallel Processors, MPP kind of architecture and parallel computations on these systems involve message passing between nodes [34]. Linux cluster that is used in this study consists of two workstations. Each of the workstations has dual processors which are Pentium III 650 MHz, and 512 Mb memory. These computers are connected via 100 Mbps Ethernet network. Rocks Cluster Distribution which is based on RedHat Linux is used as the operating system. MPICH which is a portable implementation of MPI is available in the default installation of Rocks. MPI-1 version which is fully supported by MPICH is used in this study.

Single Program Multiple Data (SPMD) model of parallel programming is used. In this model, "there is only one program and each process uses the same executable working on different set of data" [34]. A process differs from other processes by an integer variable which shows the rank of the process between all processes. Using this rank value, operation range of a process is defined. Basic steps of parallel computation in SPMD model involves first reading the same input by all processes; secondly each process get their rank; third step consists of defining the operation range of the process according to the rank; fourth step is carrying out the necessary calculations in the defined range; in the fifth step, necessary data transfer for further calculations or gathering all data to one process is done; and finally output is given by the root process. [34]

Fig. 3.5 shows flowchart of the serial program. RUNGE-KUTTA3 subroutine involves the ROEFLUX, VTGRAD, BALDWIN-LOMAX, VISOCUS subroutines for residual calculations. Nearly all of the run time of the serial program is consumed by the time-stepping loop, and parallelization of this part is sufficient for getting a desired parallel performance.

Fig. 3.6 shows flowchart of the parallel program. PARA-RANGE subroutine determines the operation ranges of processes according to their ranks. DATATRANS subroutine does the required message passing in the "interface boundaries [5]" between neighboring processes. DATACOLLECT subroutine gathers data to the root process that gives the output. In writing PARA-RANGE, DATATRANS and DATACOLLECT subroutines, code segments from [34] is used. The subroutines that do calculations are parallelized by parallelizing all of the DO loops with range variables. These DO loops operate only in the operation range of the process.

PARA-RANGE subroutine applies "block distribution [34]" of the domain. In an example of one-dimensional block distribution, this subroutine divides a range of *m* cells into equal portions for *n* processes, if it is possible. If it is not, *mod(m, n)* processes have one more cell than remaining [*n- mod(m, n)*] processes.

During parallelization, new boundaries at process interfaces appear. At these interfaces, a process needs boundary cells of neighboring process in order to calculate fluxes at the interface and vice versa. This information is interchanged between processes in DATATRANS subroutine.

In this study, domain decomposition is done in one dimension which is the *y*-direction. A schematic view of domain decomposition in *y*-direction is shown in Fig. 3.7 for three processes case.

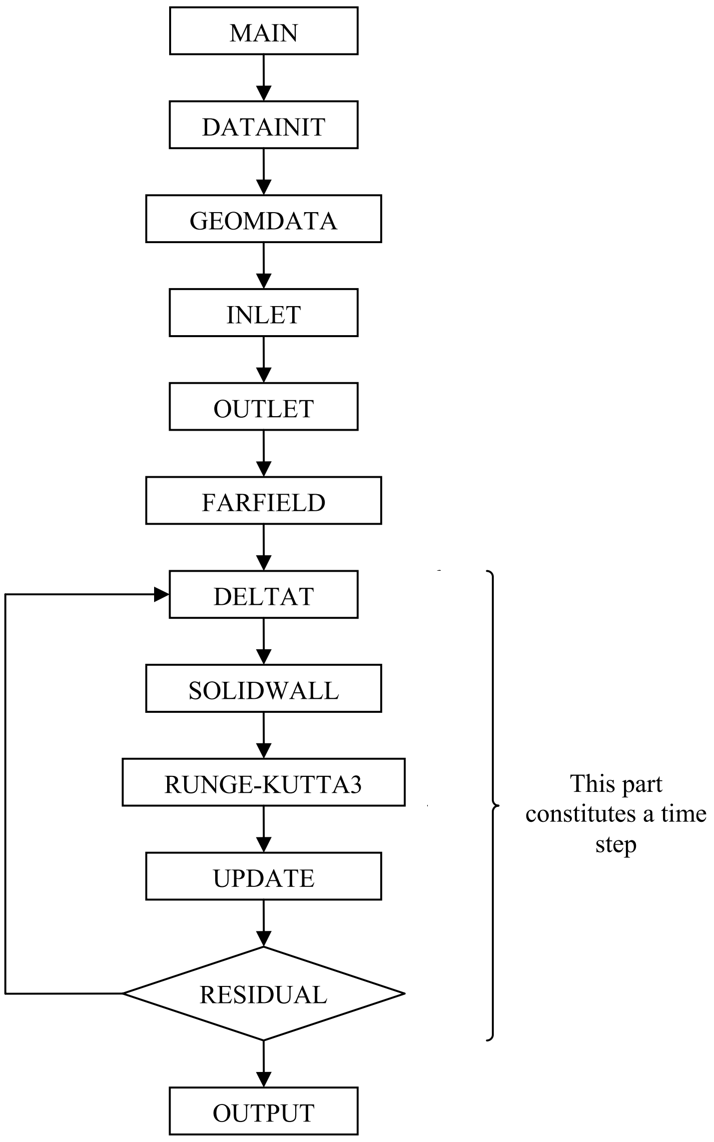


**Figure 3.5:** Flowchart of the serial program

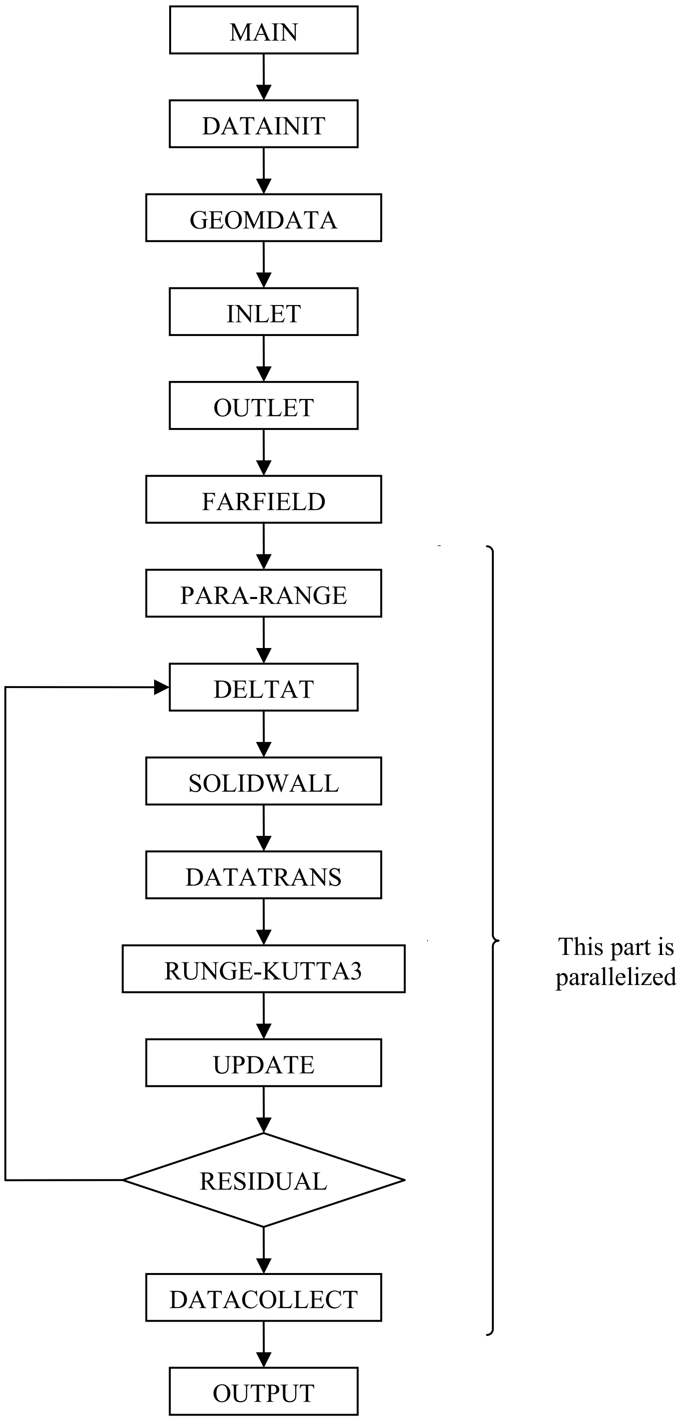


**Figure 3.6:** Flowchart of the parallel program

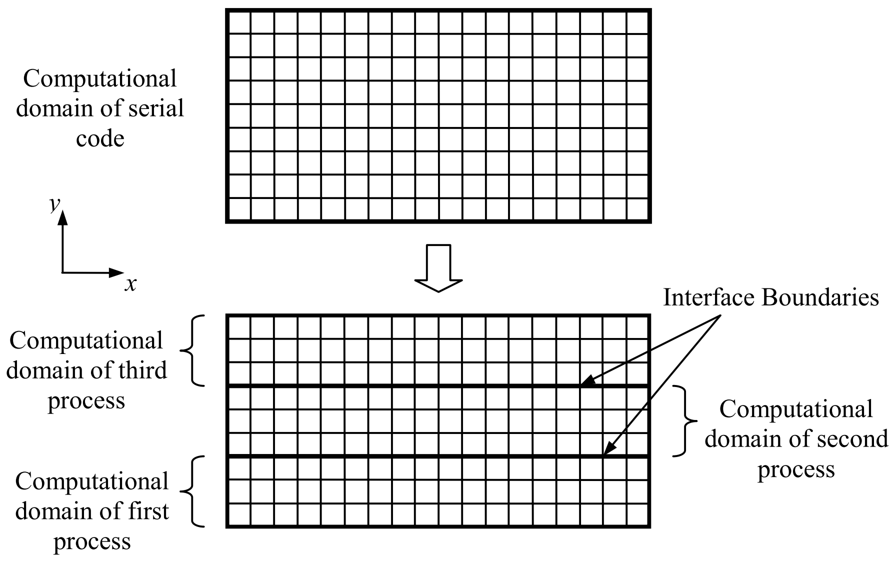

Computational domain of serial code
y
x
Interface Boundaries
Computational domain of third process
Computational domain of second process
Computational domain of first process

**Figure 3.7:** Domain decomposition in *y*-direction for three processes case

# CHAPTER 4

# RESULTS AND DISCUSSION

## 4.1 Inviscid Flow in Two Dimensions

For validation of two-dimensional Euler solver, “subsonic and transonic flows in a channel with a circular arc bump” are investigated as test cases which are introduced by Ni [39]. These test cases are widely used and known as circular bump. Following sections involves subsonic and transonic results obtained from the developed Euler solver that are compared with the results of Ni [39]. Computational domain can be seen in Fig. 4.1 with a mesh size of 129 x 33.

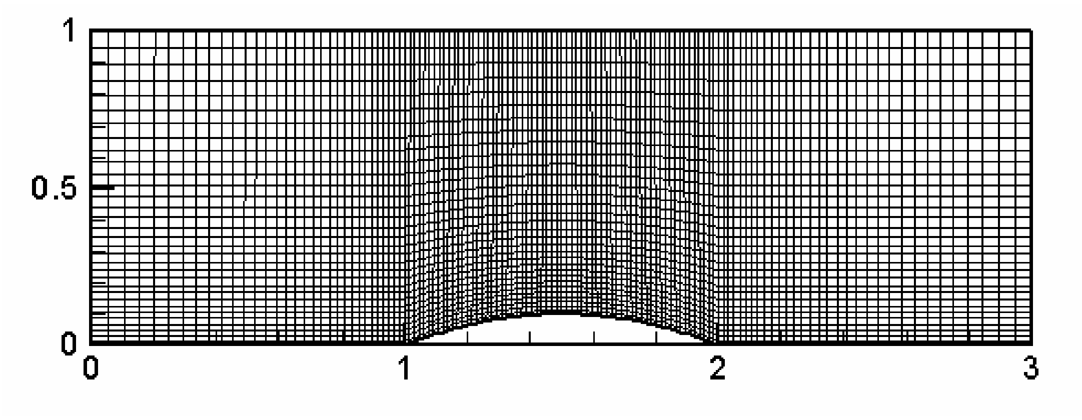


**Figure 4.1:** Mesh for circular bump (129 x 33)

It can not be possible to get comparable results with 65 x 17 meshing of the domain that Ni used, since results of Ni are taken from a second order method as stated in [39], and Roe flux difference splitting scheme is a first order method, Hence, a mesh size of 129 x 33 is used in this study.

When average of absolute changes for *x* direction momentum in the normalized form decreases below $10^{-8}$, it is assumed that the solution is converged.

### 4.1.1 Subsonic Ni Bump

For subsonic case, results are taken at a Mach number of 0.5 and stagnation conditions of 350 K and 150 kPa. When isomach lines given in Fig. 4.2 and Fig. 4.3 are compared, it can be seen that flow shows a symmetric behavior on the bump like the results of Ni [39]. As stated by Ni [39], symmetry of the solution is a sign of accuracy for subsonic case, and this is achieved on the bump. However, there is not any symmetry for inlet and outlet regions. This is most probably due to boundary conditions in inlet and outlet which are taken as far-field. Also, in the post-processing, it is seen that isomach lines close to Mach number of 0.5 have very different behaviors such as isomach line for the Mach number of 0.497 reaches bottom surface. When Fig. 4.4 is investigated, values of Mach numbers at the top wall exactly match the result of Ni. For the bottom wall, inlet and bump region results are same as the results of Ni. However, there is a difference in the outlet region which is again most probably due to boundary conditions applied for outlet.

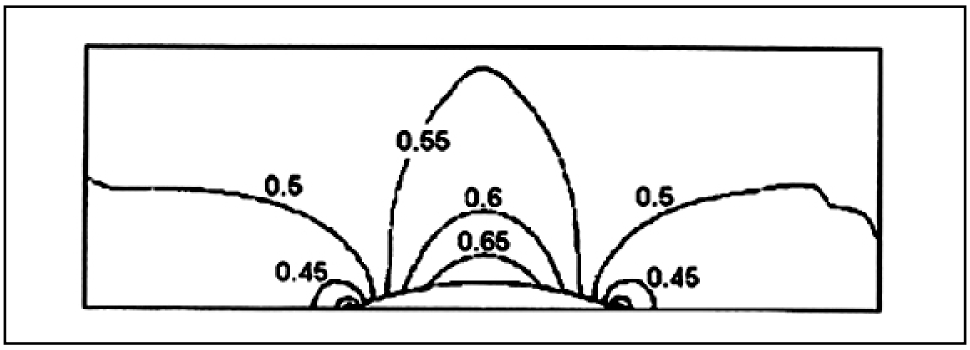


**Figure 4.2:** Isomach lines of Ni at inlet Mach number of 0.5 [39]

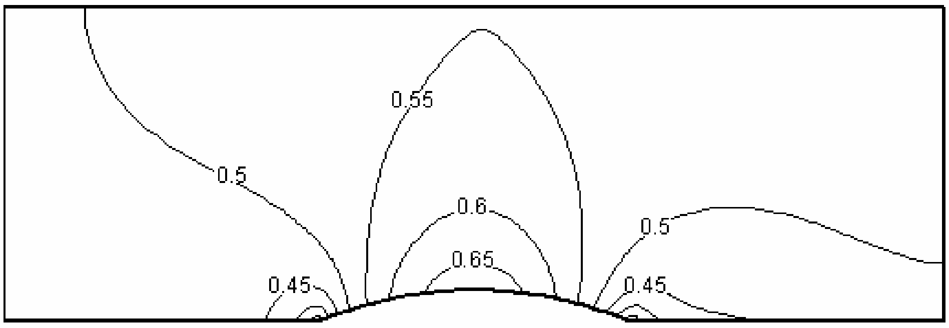


**Figure 4.3:** Isomach lines of the Euler solver at inlet Mach number of 0.5

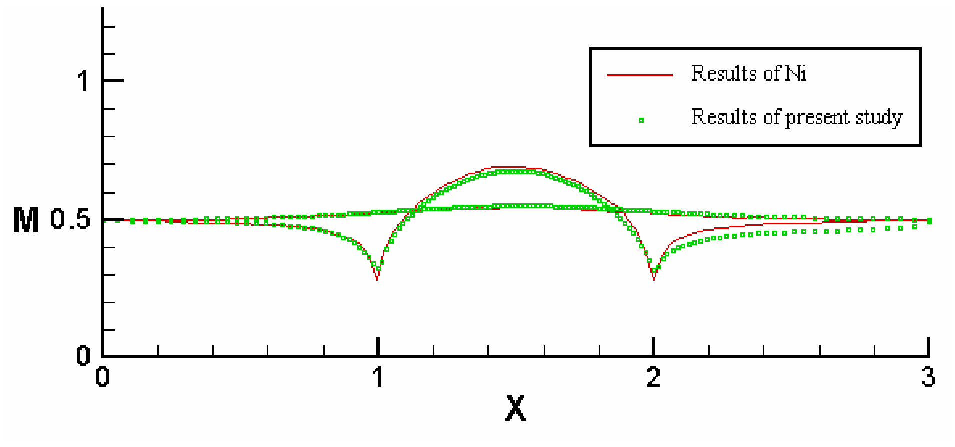


**Figure 4.4:** Comparison of the Mach number distributions on the top and the bottom surfaces at inlet Mach number of 0.5

### 4.1.2 Transonic Ni Bump

For transonic case, results are taken at a Mach number of 0.675 and stagnation conditions of 350 K and 150 kPa. When Fig. 4.5 and Fig. 4.6 are compared, it can be seen that a good qualitative agreement except the outlet region is obtained. There is also a good agreement for Mach number distributions on the top and bottom surfaces, as it can be seen in Fig. 4.7. Also, location of shock found to be around % 72.5 of chord which is %72 as stated by Ni [39].

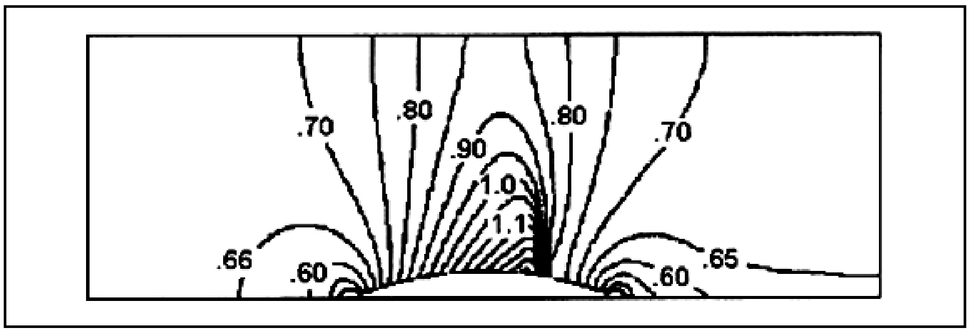


**Figure 4.5:** Isomach lines of Ni at inlet Mach number of 0.675 [39]

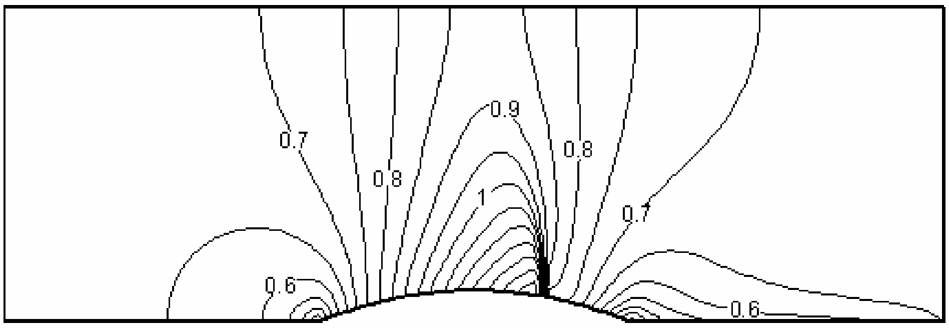


**Figure 4.6:** Isomach lines of the Euler solver at inlet Mach number of 0.675

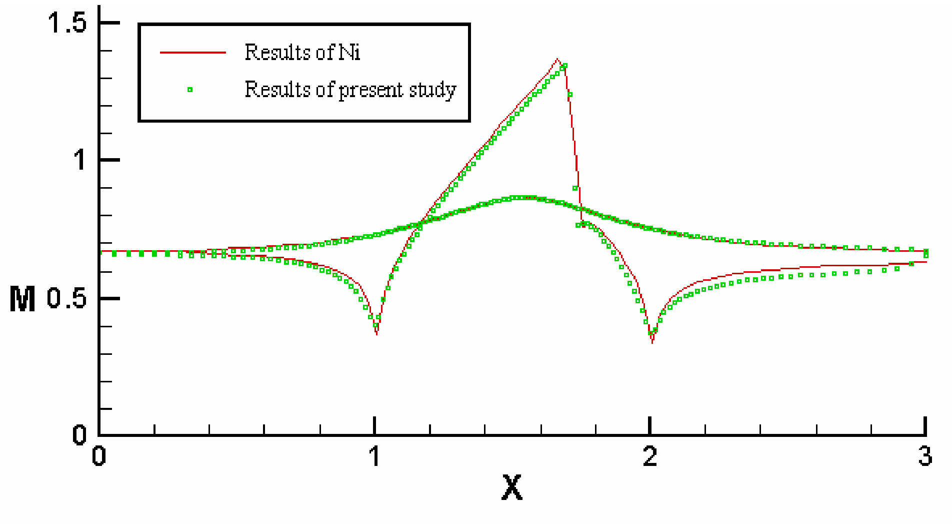


**Figure 4.7:** Comparison of the Mach number distributions on the top and the bottom surfaces at inlet Mach number of 0.675

## 4.2 Viscous Flow

After adding sub programs calculating viscous fluxes and turbulent viscosity, developed code is capable of solving two dimensional viscous flows which may be turbulent. Code validation is done by running code for flat plate problem. Following two sections involves these tests. First test is for laminar flow and gives validation for sub programs calculating viscous fluxes. Second test is for turbulent flow and involves validation of sub programs related to the applied turbulence model which is Baldwin-Lomax turbulence model.

### 4.2.1 Laminar Flat Plate Problem

Test cases are carried out at a subsonic Mach number of 0.3 and stagnation conditions of 298 K and 100 kPa. Length of the flat plate is taken as 0.005 m. Reynolds number is about 39000 for this flow.

Mesh for this test has 121 x 81 nodes. Flat plate is located at the middle of the lower surface of the domain between (0, 0) and (0.005, 0) points. There is a distance of two times the length of flat plate to the inlet boundary and again a distance of two times the length of flat plate to the outlet boundary. Height of the computational domain is three times the length of the flat plate. On the lower surface of the domain, there are 30 nodes before the flat plate leading edge, 71 nodes on the flat plate and 20 nodes after the flat plate. Nodes are highly clustered near the leading edge, since there is a high velocity gradient in this region. Outlet region have finer node distribution then the inlet region, while both of these regions have very coarser grids then the one over the flat plate.

Boundary layer thickness for the Blasius solution is given as;

$$\delta = 5x\,\mathrm{Re}_x^{-1/2} \tag{4.1}$$

By using Eqn. (4.1), boundary layer thickness at the trailing edge can be estimated as $\delta = (1.27)10^{-4}$ m. Mesh is also highly clustered on the bottom surface in order to catch boundary layer behavior which involves high velocity gradients in comparison with the other regions far from the flat plate. There are 26 cells in the boundary layer at the trailing edge according to the estimated boundary layer thickness; while the minimum $\Delta y$ value is 0.00025 of the flat plate length which is $\Delta y = (1.25)10^{-6}$ m. Mesh can be seen in Fig. 4.8.

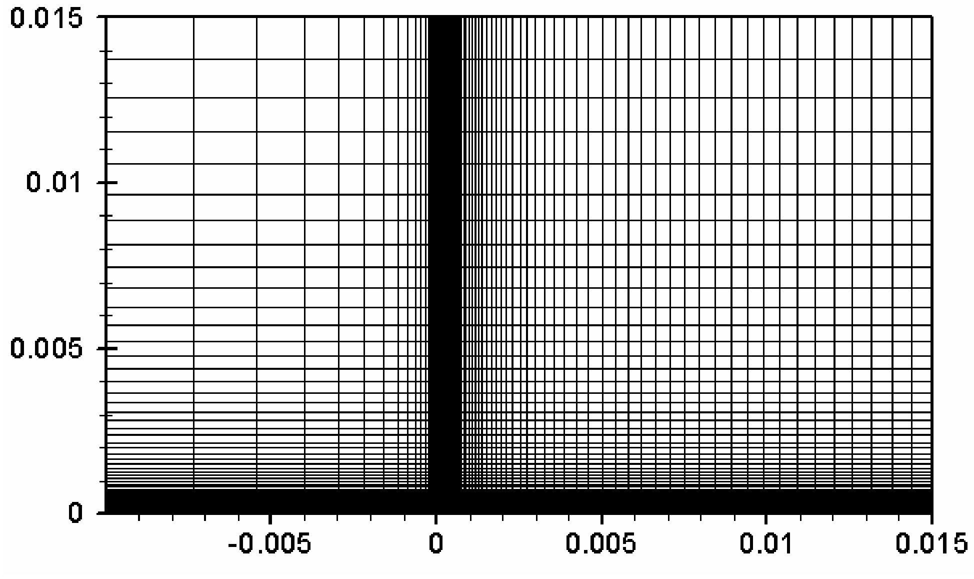


**Figure 4.8:** Mesh for Laminar Flat Plate having a grid of 121x81

The local skin friction coefficient, $C_{fx}$ distribution over the flat plate and velocity profiles are investigated and these results are compared with the Blasius solution for the flat plate. The local skin friction coefficient, $C_{fx}$ defined as the following;

$$C_{fx} = \frac{\tau_w}{\rho U_\infty^2 / 2} \tag{4.2}$$

where the wall shear stress, $\tau_w$ is determined by numerical solution or given by the Eqn. (4.3) below for the Blasius solution.

$$\tau_w = (0.3321)\,\rho U_\infty^2 \,\mathrm{Re}_x^{-1/2} \tag{4.3}$$

The local skin friction coefficient, $C_f$ distribution on the flat plate together with the Blasius solution is given in Fig. 4.9. It can be seen that at the leading edge it does not match the Blasius solution and at the trailing edge there is also a discrepancy. This is an expected difference, since in these regions the rate of change of $u$ velocity in $x$-direction is comparable with the rate of change of $u$ velocity in $y$-direction. This fact contradicts with one of the main approximations of Prandtl's boundary layer equations which assume that the rate of changes in $x$-direction is very small in comparison with the rate of changes in $y$-direction.

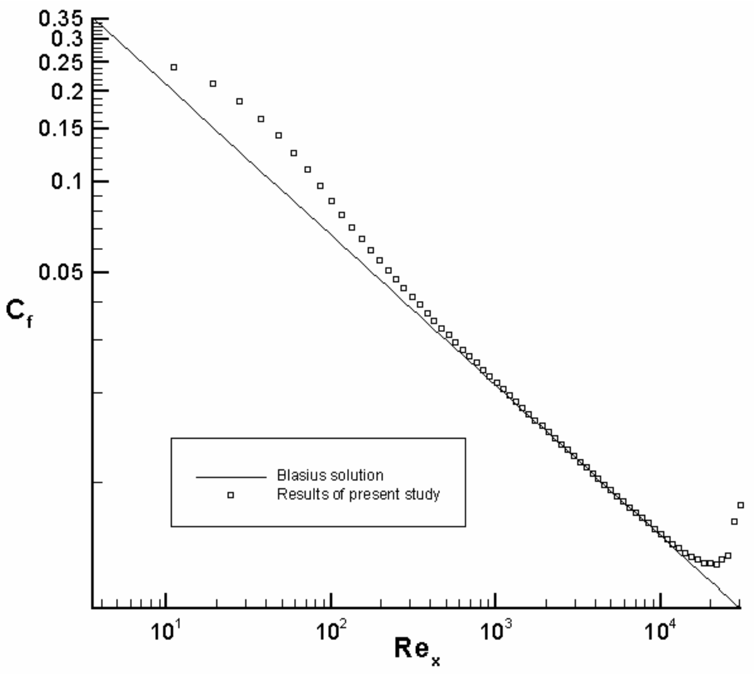


**Figure 4.9:** Local skin friction coefficient values obtained from the numeric solution and the Blasius solution

In the investigation of velocity profiles, three locations on the flat plate which are at the distances of 30%, 60% and 90% of the length of the flat plate from the leading edge are studied. Velocity distributions obtained from numerical results are compared with the Blasius solution in the Figures 4.10 to 4.12.

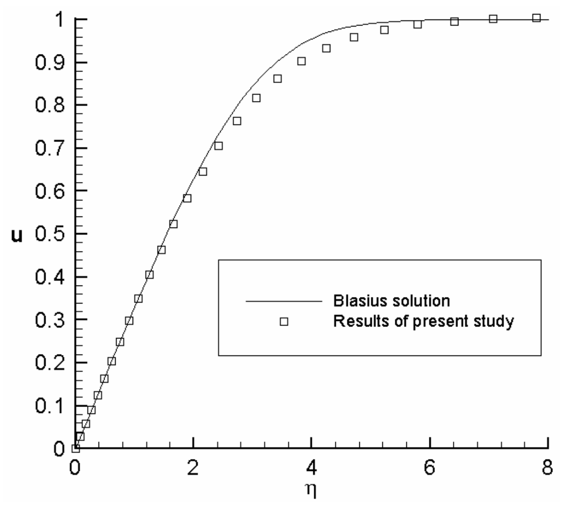


**Figure 4.10:** Comparison of *u* velocity profiles when distance is 30%

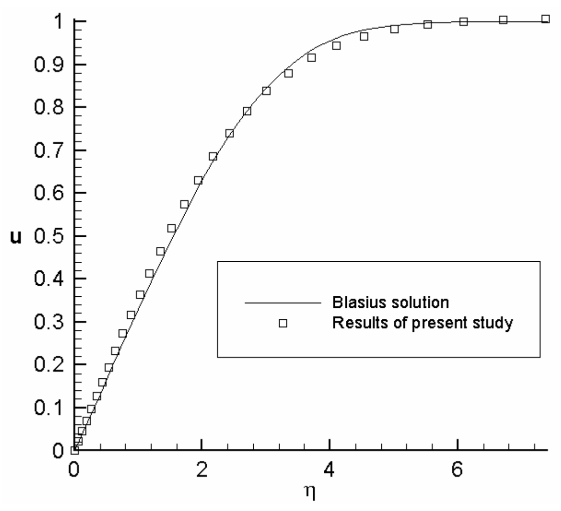


**Figure 4.11:** Comparison of *u* velocity profiles when distance is 60%

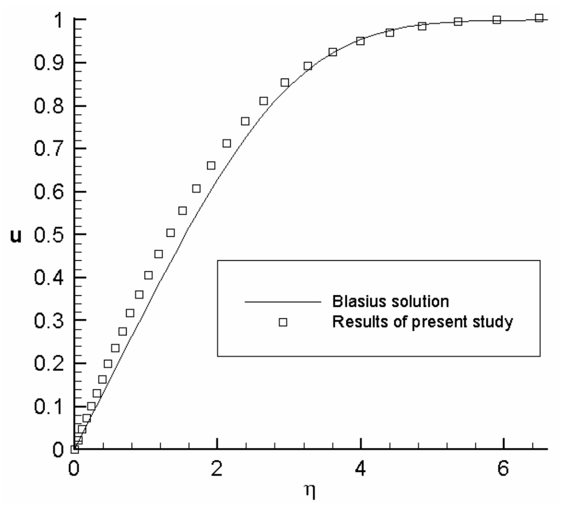


**Figure 4.12:** Comparison of *u* velocity profiles when distance is 80%

As seen from the figures, *u* velocity profiles are in a good agreement with the Blasius solution.

Solution is assumed to be converged, when average of absolute changes for all of the conservative variables in the normalized form decreases below $10^{-8}$. Residual history for density is given in Fig. 4.13.

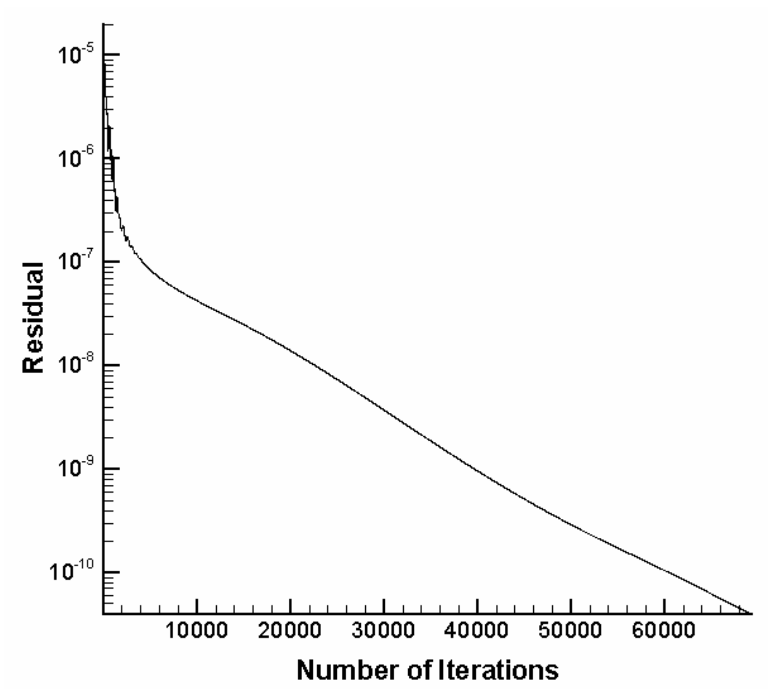


**Figure 4.13:** Residual history for density

### 4.2.2 Turbulent Flat Plate Problem

Test cases are carried out at a subsonic Mach number of 0.3 and stagnation conditions of 298 K and 100 kPa. Length of the flat plate is taken as 1 m. Reynolds number is about 6400000 for this flow.

Mesh for this test has 121x81 nodes. Flat plate is located between (0, 0) and (1, 0) points. There is a distance of two times the length of flat plate to the inlet boundary. Trailing edge of the flat plate intersects the outlet boundary. Height of the computational domain is three times the length of the flat plate. On the lower surface of the domain, there are 40 nodes before the flat plate leading edge, 81 nodes on the flat plate. Mesh is shown in Fig. 4.14.

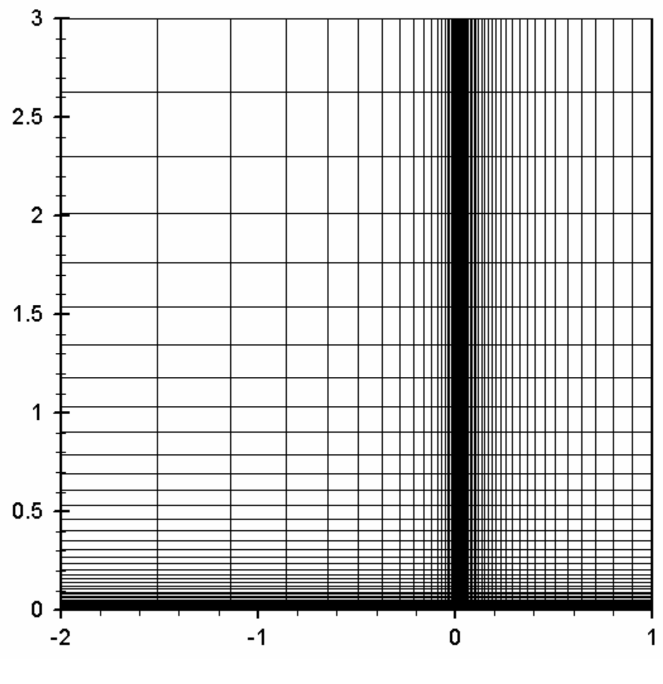


**Figure 4.14:** Mesh for Turbulent Flat Plate having a grid of 121x81

Nodes are highly clustered near the leading edge, since there is a high velocity gradient in this region. Mesh is also highly clustered on the bottom surface in order to catch boundary layer behavior which involves high velocity gradients in comparison with the other regions far from the flat plate. The minimum $\Delta y$ value is 0.00001 of the flat plate length.

The local skin friction coefficient, $C_{fx}$ distribution over the flat plate is investigated and these results are compared with the Blasius solution for laminar part of the flow and with Schlichting's experimental result in Eqn. (4.4) given in [4] for turbulent part of the flow.

$$C_{fx} = \frac{0.0592}{\mathrm{Re}_x^{0.2}} \tag{4.4}$$

For the Reynolds number value of the flow which is 6400000, transition point is taken as 0.054 m from the leading edge of the flat plate. Reynolds number value at this point is 345000. The local skin friction coefficient, $C_f$ distribution on the flat plate is given in Fig. 4.15. It can be seen that at the leading edge it does not match with the Blasius solution and at the turbulent part it gives result in an agreement with the experimental results except at the trailing edge. The discrepancy in the leading edge is mainly due to the fact that "rate of changes in *x*-direction is very small in comparison with the rate of changes in *y*-direction" approximation of Prandtl's boundary layer equations does not hold for this region.

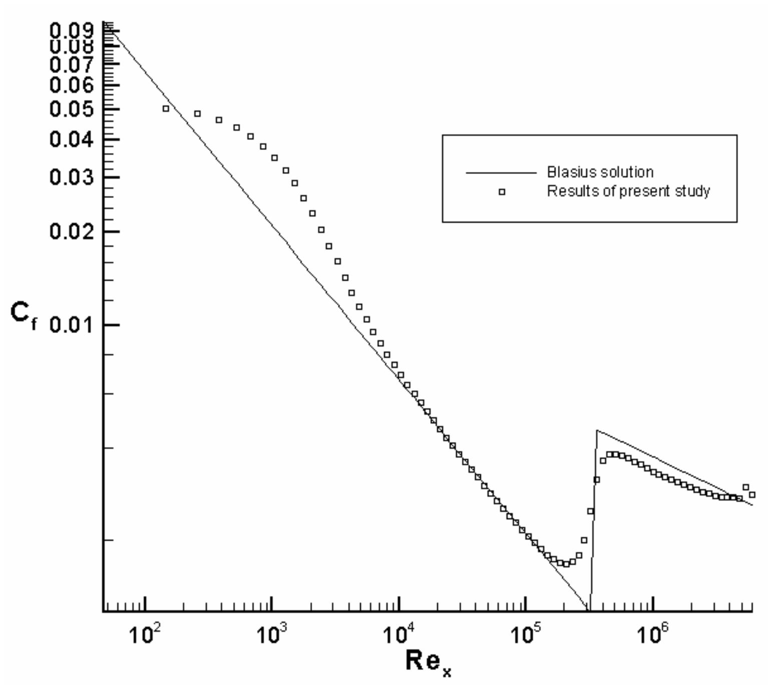


**Figure 4.15:** Local skin friction coefficient values obtained from the numeric solution, the Blasius solution and Schlichting's experiment

Velocity profiles on the flat plate are compared with the universal velocity profile. Universal velocity profile is given in terms of dimensionless variables which are dimensionless velocity, $U^+$ and dimensionless distance, $y^+$. Dimensionless velocity defined as; [4, 41]

$$U^+ = \frac{u}{u_\tau} \tag{4.5}$$

Universal velocity profile is given in terms of these variables as; [4]

$$U^{+} = \begin{cases} y^{+} & 0 \le y^{+} \le 5 \\ (5.0)\log(y^{+}) - (3.05) & 5 < y^{+} < 30 \\ (2.5)\log(y^{+}) - (5.5) & y^{+} \ge 30 \end{cases} \qquad (4.6)$$

The locations where this comparison is carried out are at the distances of 23%, 40% and 64% of the length of the flat plate from the leading edge. Results are shown in the figures 4.16 to 4.18. There is a good agreement between numerical solution and universal velocity profile.

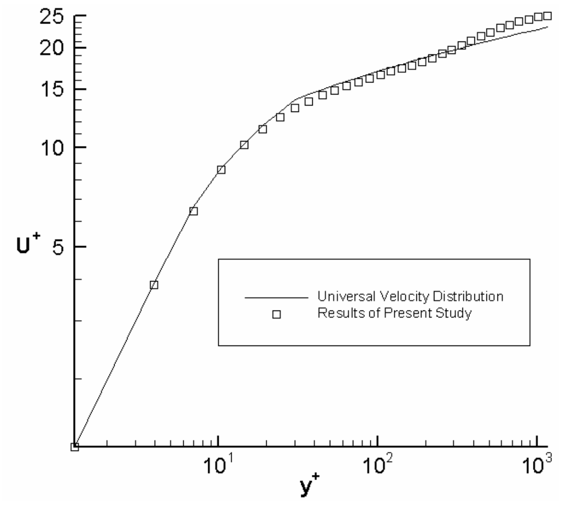


**Figure 4.16:** Comparison of $U^{+}$ velocity profiles when distance is 23%

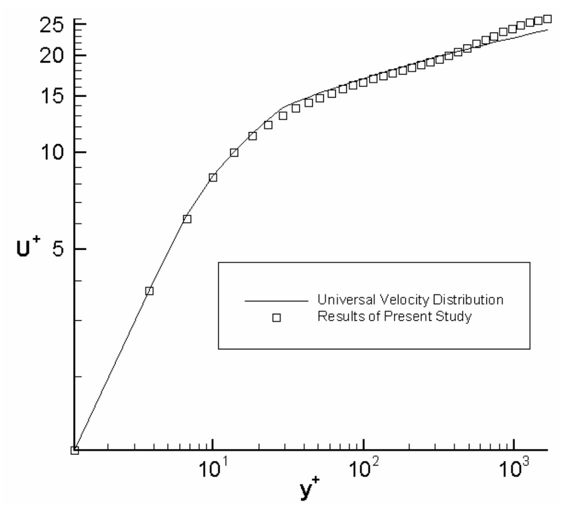


**Figure 4.17:** Comparison of $U^+$ velocity profiles when  distance is 40%

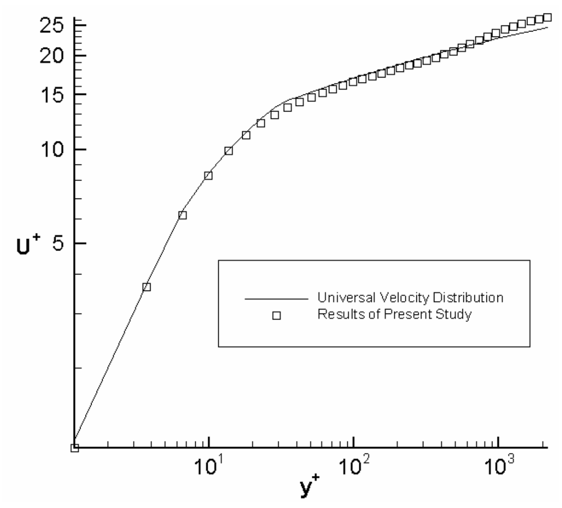


**Figure 4.18:** Comparison of $U^+$ velocity profiles when distance is 64%

Solution is assumed to be converged, when average of absolute changes for all of the conservative variables in the normalized form decreases below $10^{-8}$. Residual history for density is given in Fig. 4.19.

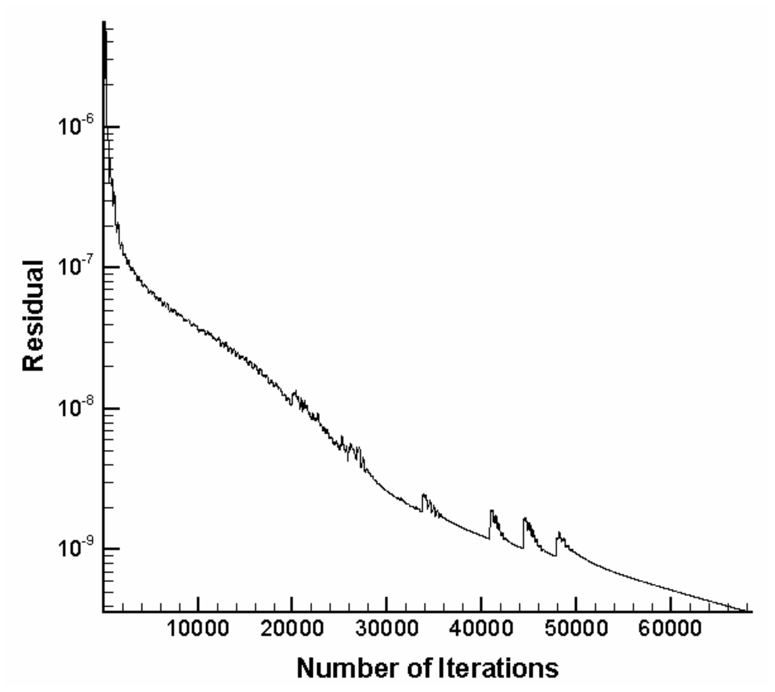


**Figure 4.19:** Residual history for density

## 4.3 Results of Parallel Processing

In this study, two-dimensional solver is parallelized in one dimension which is *y*-direction. Parallel performance is tested for laminar flow by using three mesh alternatives which have 121x81, 121x161 and 241x81 grid size. The first mesh is the same mesh that is used for laminar flat plate problem. The other two mesh alternatives are generated from this mesh. The mesh having size of 121x161 is generated by dividing *y*-direction intervals into two equal parts, and the 241x81 mesh is generated by dividing *x*-direction intervals into two equal parts. Solutions are assumed to be converged, when average of absolute changes for all of the conservative variables in the normalized form decreases below $10^{-6}$. This criteria is tested in every ten time steps in order to decrease communication needs for convergence calculations.

The local skin friction coefficient, $C_f$ distributions on the flat plate together with the Blasius solution is given in Fig. 4.20 to 4.22. These solutions are for the mesh size of 121x81. In these figures, results obtained from the parallel solutions of two, three and four processors cases are compared with the result of the serial code. Parallel results match exactly the results obtained from the serial code.

Run times and speed-up values for the serial code and parallel case of two, three and four processors are given in Table 4.1. Speed-up is a performance measure of parallel processing, and defined as the ratio of serial run time to parallel run time. In ideal case, if a problem is solved by *p* number of processors, then ideal speed-up is *p*. However, speed-up values deviate from the ideal value due to serial parts of the parallel program which are not parallelized, and due to time losses during communication. As seen from Table 4.1, parallel speed-up deviates from the ideal speed-up, as number of processors increases. For two processors case, speed-up is very close to ideal speed-up. This is because of the fact that these two processors are on the same computer, and as a result there is not any restriction on bandwidth of communication like in the cases of Ethernet network communications. For three and four processors case, Ethernet network communication is started to be used and parallel speed-up decreases due to time losses of communication via Ethernet network.

**Table 4.1:** Run times and Speed-up for mesh size of 121x81

| Number of Processors | Run time (min) | Speed-up |
|---|---|---|
| 1 (Serial) | 315 | 1 |
| 2 | 161 | 1.96 |
| 3 | 108 | 2.92 |
| 4 | 135 | 2.33 |

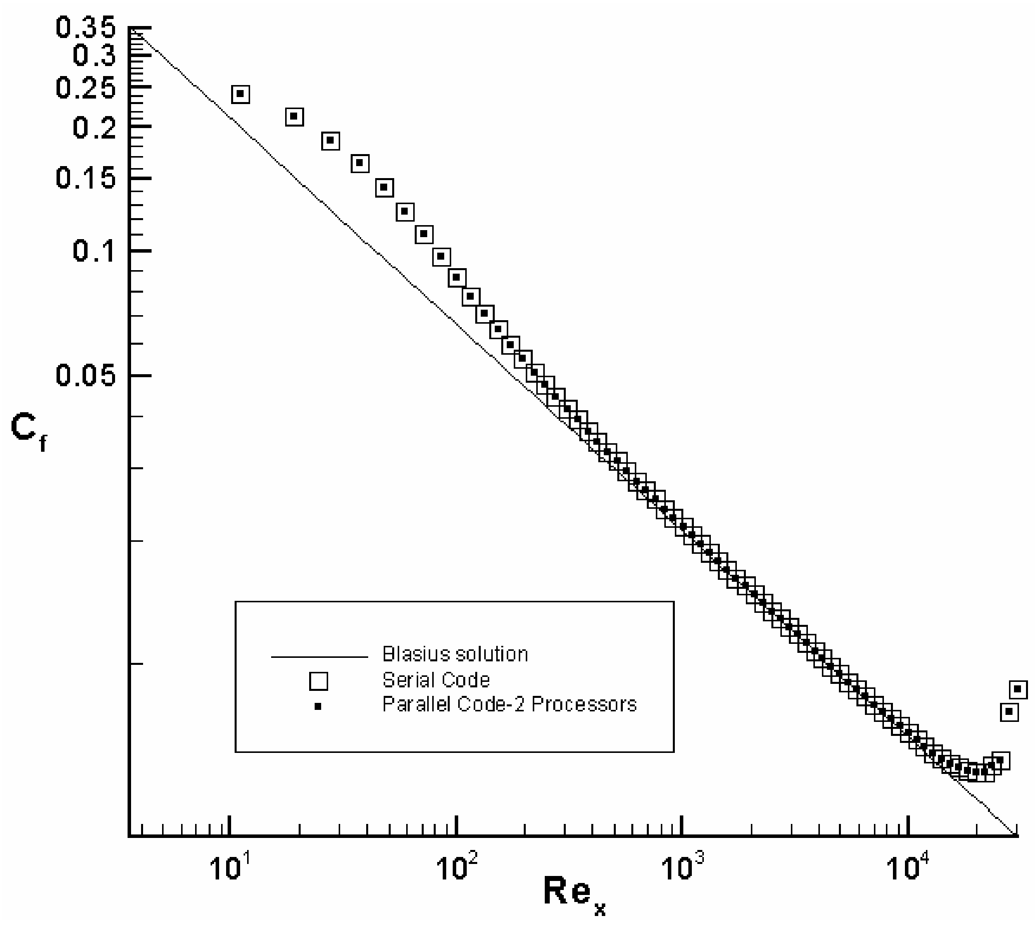


**Figure 4.20:** Local skin friction coefficient values obtained from serial code and parallel code for two processors together with the Blasius solution for mesh size of 121x81

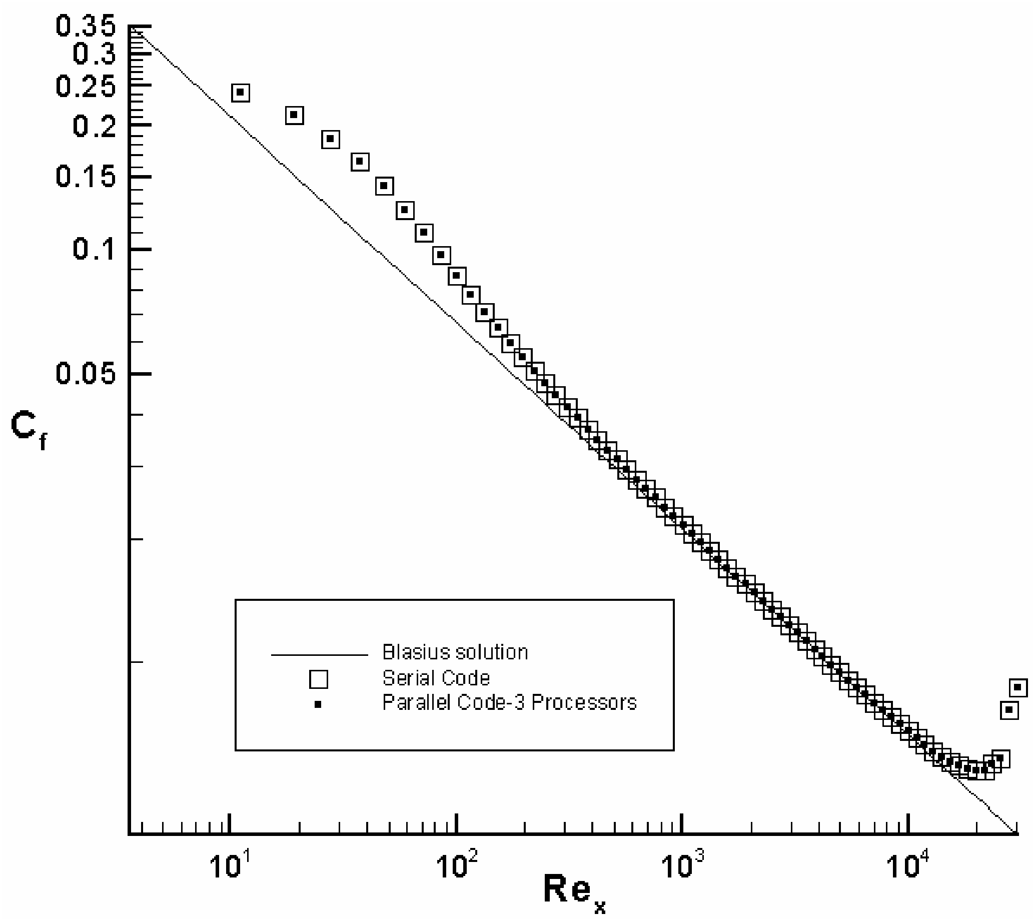


**Figure 4. 21:** Local skin friction coefficient values obtained from serial code and parallel code for three processors together with the Blasius solution for mesh size of 121x81

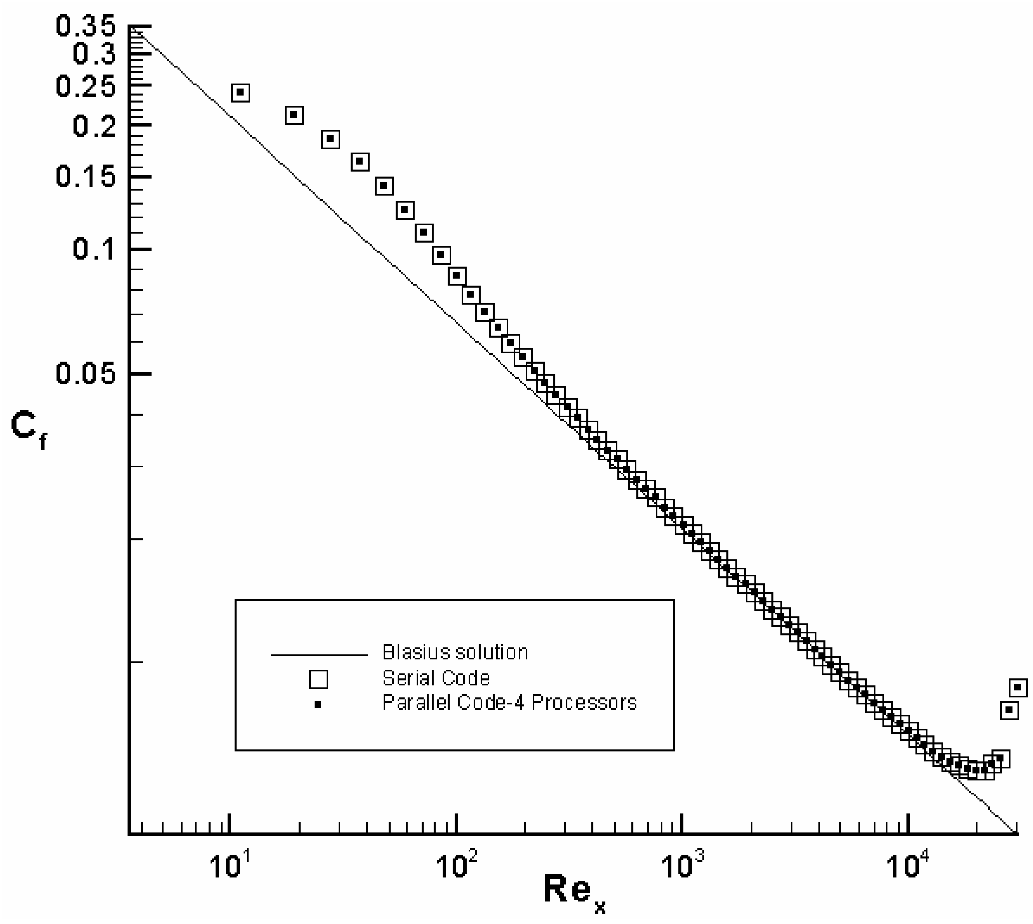


**Figure 4. 22:** Local skin friction coefficient values obtained from serial code and parallel code for four processors together with the Blasius solution for mesh size of 121x81

The local skin friction coefficient, $C_f$ distributions for mesh size of 121x161 are given for two and four processors cases in Fig. 4.23 and Fig. 4.24. Parallel results for these cases also match exactly the results obtained from the serial code.

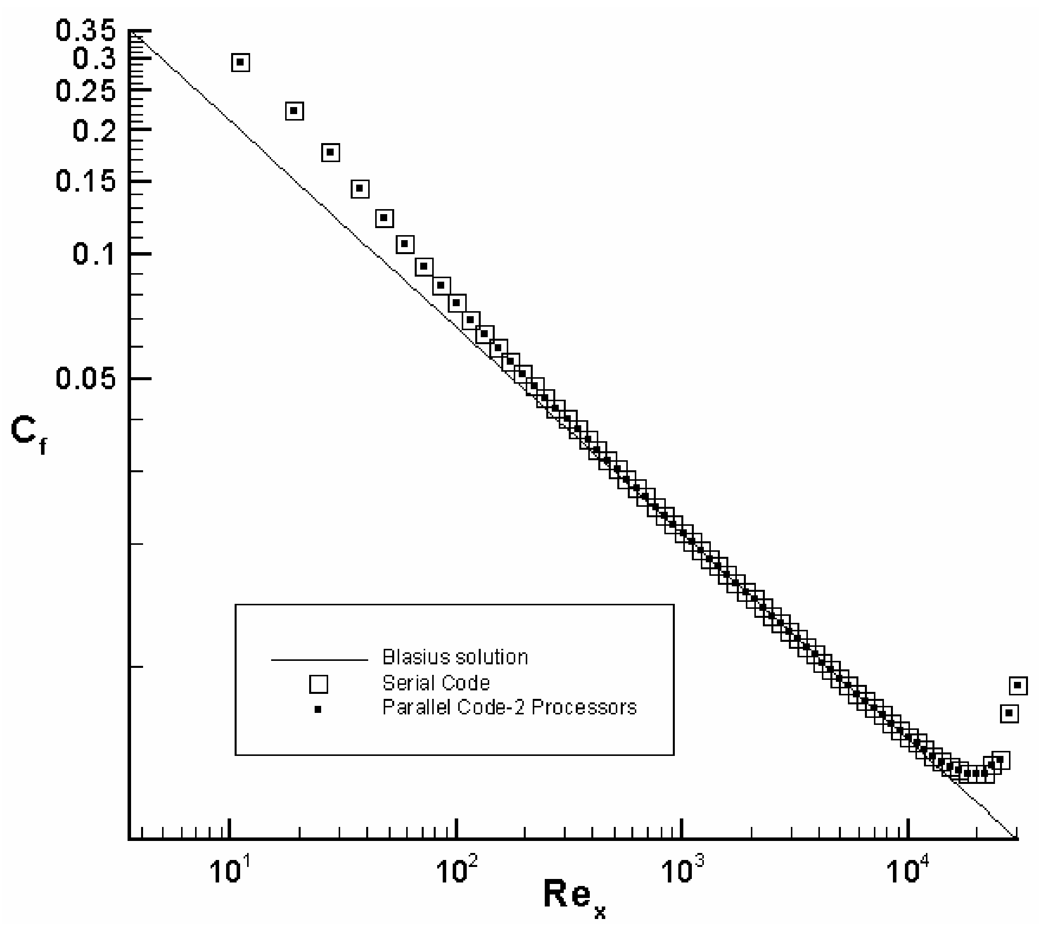


**Figure 4.23:** Local skin friction coefficient values obtained from serial code and parallel code for two processors together with the Blasius solution for mesh size of 121x161

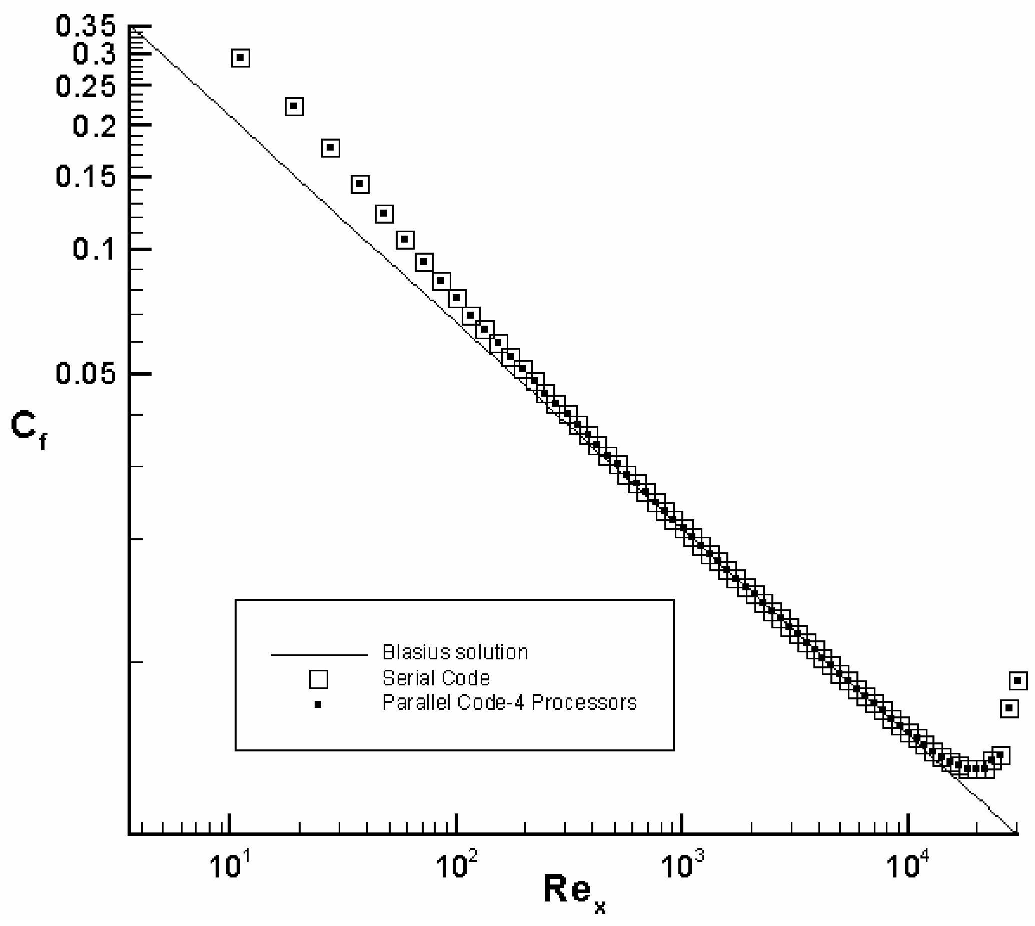


**Figure 4. 24:** Local skin friction coefficient values obtained from serial code and parallel code for four processors together with the Blasius solution for mesh size of 121x161

In mesh size of 121x161, when the domain decomposition is done in *y*-direction, same amount of communication is needed in comparison with mesh size of 121x81; however, amount of cells in operation range of one processor is increased by two times. Run times and speed-up values for the serial code and parallel case of two and four processors are given in Table 4.2. Three processors run is not carried out, since run times are long. It is again worth to mention that in two processors case, two processors on one computer is used, and as a result there is not any communication over Ethernet network

**Table 4.2:** Run times and Speed-up for mesh size of 121x161

| **Number of Processors** | **Run time (min)** | **Speed-up** |
|---|---|---|
| 1 (Serial) | 969 | 1 |
| 2 | 483 | 2.01 |
| 4 | 399 | 2.42 |

The local skin friction coefficient, $C_f$ distributions for mesh size of 241x81 are given for two and four processors cases in Fig. 4.25 and Fig. 4.26. Parallel results for these cases also match exactly the results obtained from the serial code.

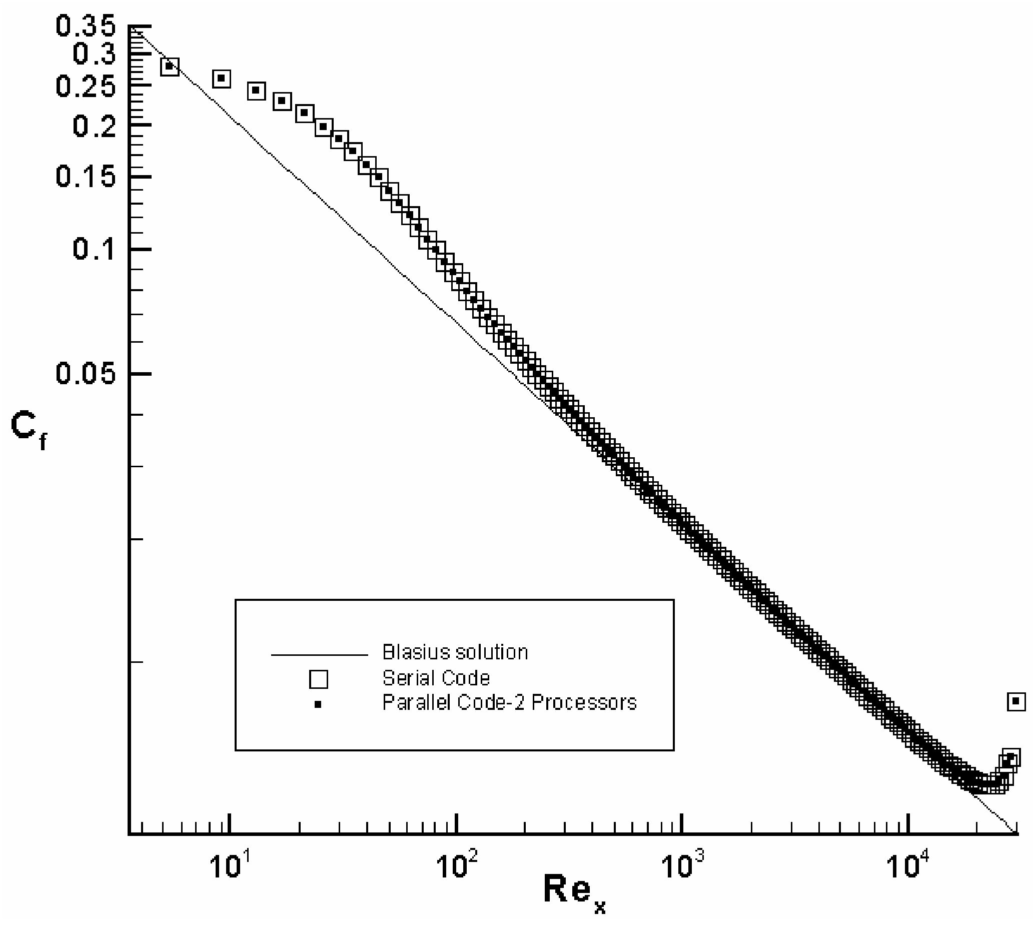


**Figure 4.25:** Local skin friction coefficient values obtained from serial code and parallel code for two processors together with the Blasius solution for mesh size of 241x81

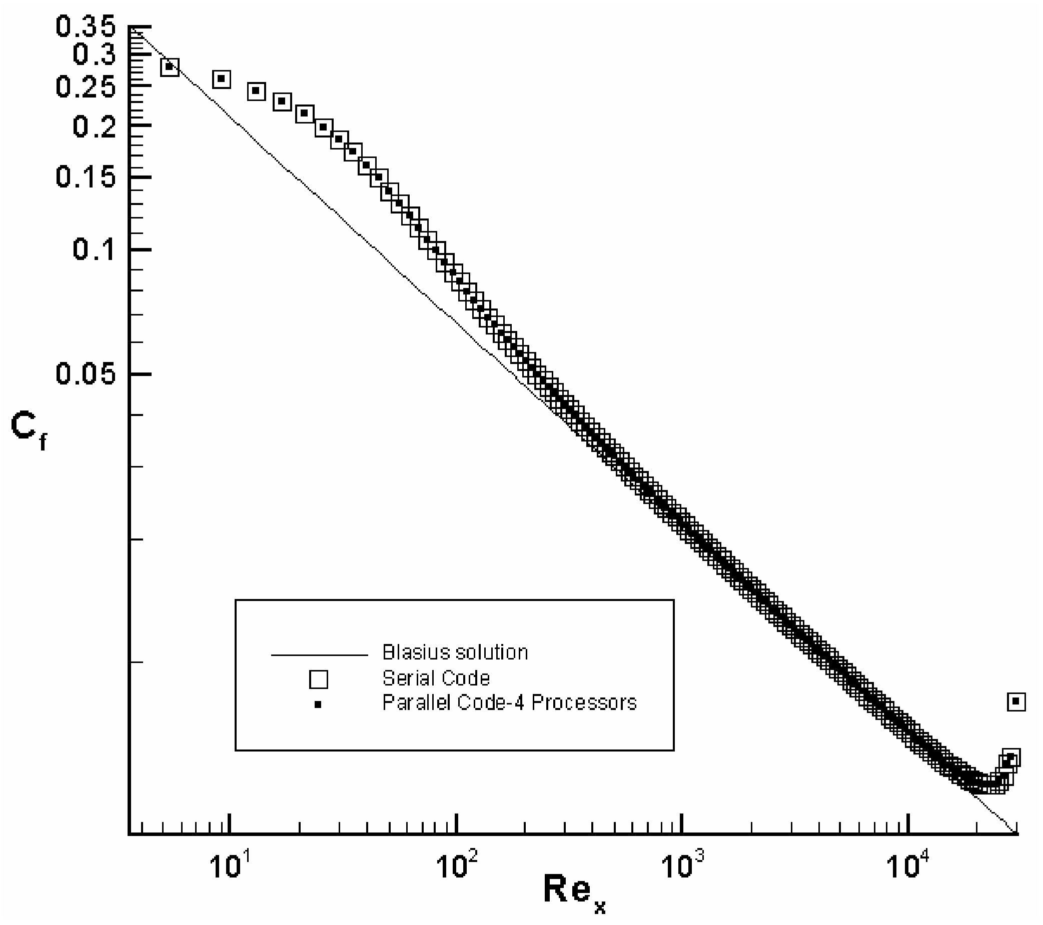
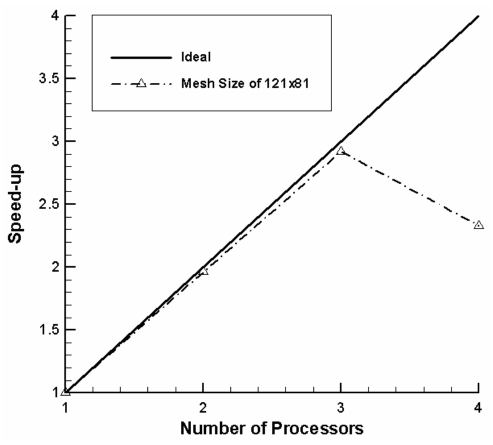


**Figure 4. 26:** Local skin friction coefficient values obtained from serial code and parallel code for four processors together with the Blasius solution for mesh size of 241x81

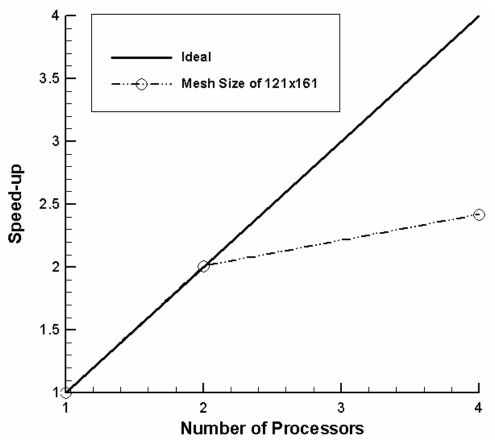
In mesh size of 241x81, amount of communication and amount of cells in operation range of one processor is increased by two times compared to mesh size of 121x81. Run times and speed-up values for the serial code and parallel case of two and four processors are given in Table 4.3. The tabulated results of speed-up values are combined in Fig. 4.27.

**Table 4.3:** Run times and Speed-up for mesh size of 241x81

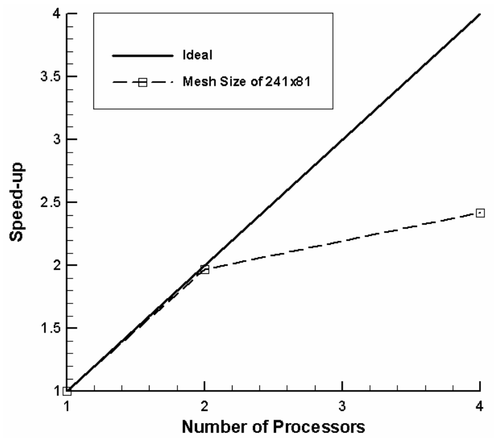
| Number of Processors | Run time (min) | Speed-up |
|---|---|---|
| 1 (Serial) | 656 | 1 |
| 2 | 333 | 1.97 |
| 4 | 271 | 2.42 |



**a)** Mesh size of 121x81



**b)** Mesh size of 121x161



**c)** Mesh size of 241x81

**Figure 4.27:** Speed-up values for various mesh alternatives

When the parallel speed-up values obtained from mesh size of 121x81 is compared with the other two mesh sizes, it can be seen that speed-up increases with the increase in the amount of cells in operation range of one processor. The reason is that, ratio of time needed for communication at each time step to time needed for calculation at each time step decreases. As a result, parallelization becomes more feasible at the expense of time losses due to communication. This becomes more apparent for four processors case where Ethernet network is used for communications between processors in different computers.

An interesting result is obtained for two processors case of mesh size 121x161. Speed-up for this case is slightly over the ideal speed-up. This is most probably due to the decrease in the processor performance that is caused by the heat generation in the processor in 969 minutes for the serial case.

When the parallel speed-up values obtained from mesh sizes of 121x161 and 241x81 are compared, it is expected that the speed-up for the former case is more than the latter case. Since, amount of communication in mesh size of 241x81 is two times the other mesh size. This result is obtained for two processor case. However, for four processor case, speed-up values are the same. This is most probably due to the communication latencies in the Ethernet network that dominate communication times, and using better hardware for communication decreases this latencies. Calculating communication time in ideal case of no communication latencies in Ethernet network makes the effect of communication latencies apparent. The amount of data that is interchanged on an interface boundary for mesh size of 121x81 is calculated as 7.56 KB for one time step. Ethernet connection speed is 100 Mbps which is equal to 12800 KB per second. In ideal case of no communication latency, data transfer at one time step is completed in $(5.9)10^{-4}$ seconds, and this creates a time difference of approximately 24 seconds in 40000 time steps. As a result, increase of communication by two times for such small data amounts does not affect speed-up, if Ethernet network is used for communication.

## 4.4 Inviscid Flow in Three Dimensions

For validation of three-dimensional Euler solver, transonic flow in a channel with a circular arc bump test case is investigated. Results obtained from three-dimensional solver are compared with the results obtained from two-dimensional solver. Computational domain can be seen in Fig. 4.28 with a mesh size of 129x33x5.

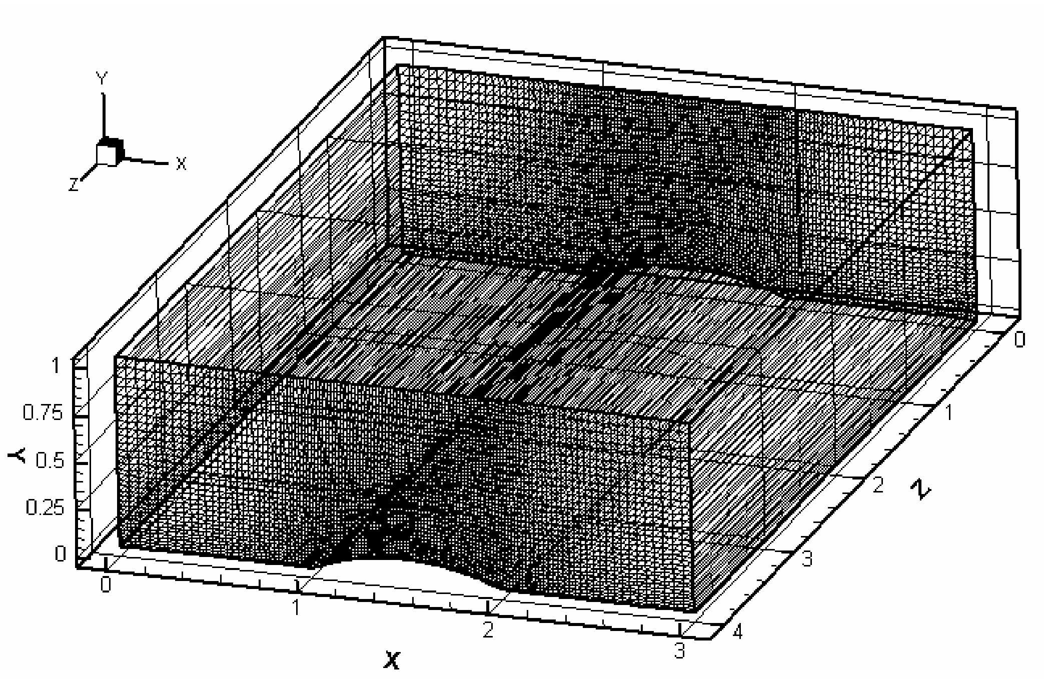


**Figure 4.28:** Mesh for circular bump in three-dimensions (129 x 33 x 5)

There are five nodes in the *z*-direction and these nodes are separated by 1 unit length. Since the flow has a two-dimensional nature, the distance between nodes in *z*-direction does not have an effect on the solution. Convergence is declared when average of absolute changes for *x* direction momentum in the normalized form decreases below $10^{-8}$. In Fig. 4.29, isomach lines in three-dimensions are shown. As it can be seen from the figure, isomach lines do not vary in *z*-direction. This is expected due to the two-dimensional nature of the flow.

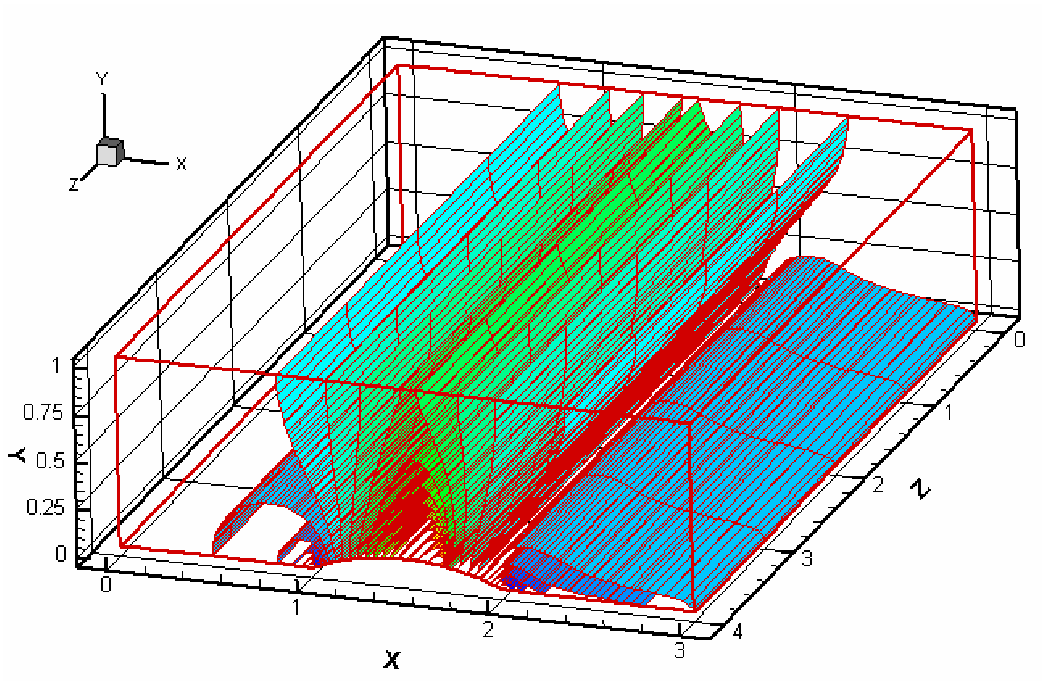


**Figure 4.29:** Isomach lines in three-dimensions at inlet Mach number of 0.675

In Fig. 4.30, isomach lines in *x*-*y* plane is shown for the first node in *z*-direction, while isomach lines obtained from two-dimensional Euler solver is given again in

Fig. 4.31 for comparison purposes. There are small differences in isomach lines especially in the shock region.

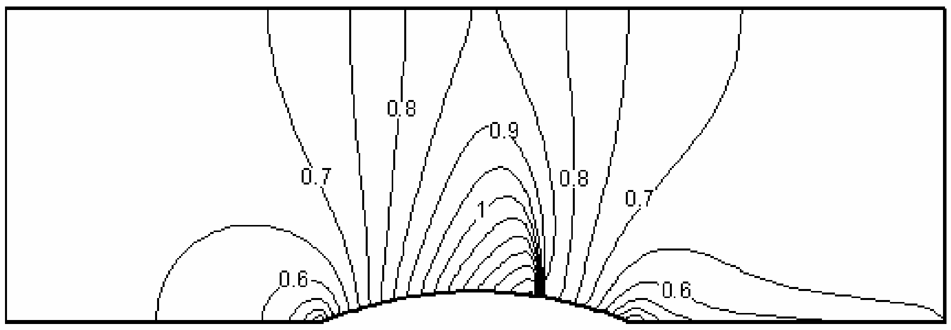


**Figure 4.30:** Isomach lines in *x*-*y* plane for the first node in *z*-direction at inlet Mach number of 0.675

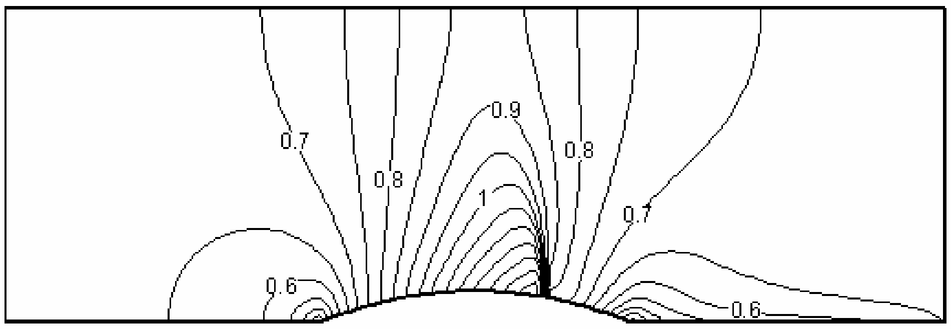


**Figure 4.31:** Isomach lines of the two-dimensional Euler solver at inlet Mach number of 0.675

In Fig. 4.32, Mach number distributions on the top and the bottom surfaces for the first node in *z*-direction are compared with the results obtained from two-dimensional Euler solver. Results obtained from three-dimensional Euler solver match the results obtained from two-dimensional Euler solver except small discrepancy around the shock region. This discrepancy may be due to the extra calculations that are carried out for rotation of local coordinates in three-dimensions.

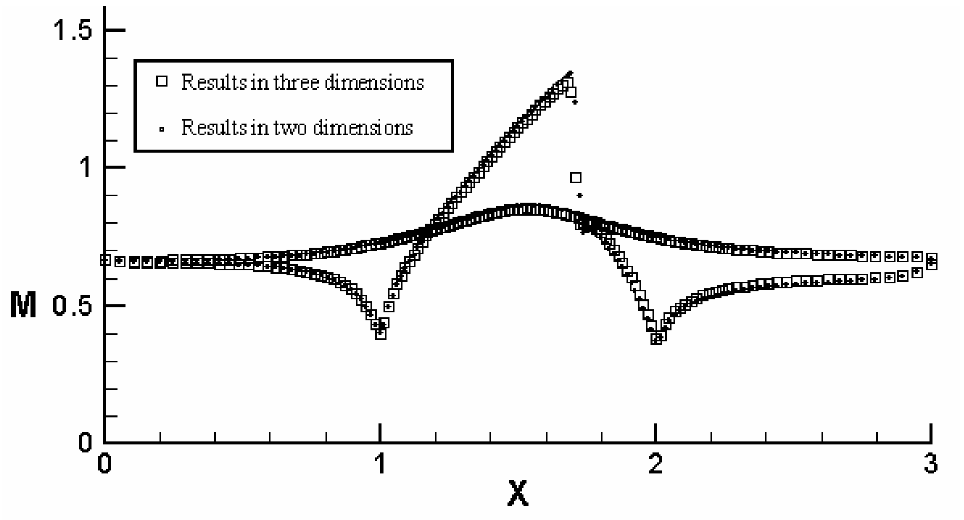


**Figure 4.32:** Comparison of the Mach number distributions on the top and the bottom surfaces at inlet Mach number of 0.675

# CHAPTER 5

# CONCLUSION

This study has two main parts. In the first part, two serial codes are developed. These codes are a Navier-Stokes solver that is capable of solving two-dimensional compressible and turbulent flows, whereas the other one is a three-dimensional compressible Euler solver. In the second part of the study, Navier-Stokes solver is parallelized in one direction.

The study started with the development of a two-dimensional compressible Euler solver. By addition of sub programs involving calculation of viscous fluxes, the code is generalized to a Navier-Stokes solver. Next, Baldwin-Lomax turbulence model was implemented on the two-dimensional Navier-Stokes solver. Finally, two-dimensional Euler solver was generalized to three dimensions. At every step, code validation was done by comparing numerical results with theoretical, experimental or other numerical results, and adequate consistency between these results was obtained.

The parallelized code developed in the second part of the study yielded satisfactory parallel performance and gave an insight to parallel processing after various mesh alternatives were tested.

Future studies based on the codes developed should include a complete generalization to three-dimensions and parallelization of the three-dimensional code. The performance obtained from parallel processing is promising. Parallel processing is an indispensable tool for solving complex problems of engineering applications, and should be used in all of the further computational studies.

As a conclusion, a solid background in computational fluid dynamics is established with this study. The codes developed are good basis for further researches. If the previously mentioned further improvements were accomplished, the resulting code would be a powerful tool for solving complex fluid problems encountered in engineering.

# REFERENCES

[1] Hirsch, C., *Numerical Computation of Internal and External Flows*, Vol. 1&2, John Wiley & Sons, 1988 & 1990

[2] Wesseling P., *Principles of Computational Fluid Dynamics*, Springer, 2000

[3] Aksel, M. H., *Lecture Notes on Computational Fluid Dynamics Using Finite Volume Method*, Department of Mechanical Engineering, METU, 2003

[4] Yalım, M. S., *Development of a Three-Dimensional Object-Oriented Navier-Stokes Solver by Using Total Variation Diminishing (TVD) Method*, M. Sc. Thesis, METU, September 2002

[5] Doğru, K., *Parallel Processing of Two-Dimensional Euler Equations for Compressible Flows*, M. Sc. Thesis, METU, May 2000

[6] McDonald, P. W., *The Computation of Transonic Flow Through Two-Dimensional Gas Turbine Cascades*, ASME Paper 71-GT-89, 1971

[7] MacCormack, R. W., and Paullay, A. J., *Computational Efficiency Achieved by Time Splitting of Finite Difference Operators*, AIAA Paper 72-154, 1972

[8] Lax, P. D., *Hyperbolic Systems of Conservation Laws II*, Comm. Pure and Applied Mathematics, 10, 537-66, 1957

[9] Lax, P. D., and Wendroff, B., *Systems of Conservation Laws*, Comm. Pure and Applied Mathematics, 13, 217-37, 1960

[10] Lax, P. D., and Wendroff, B., *Difference Schemes for Hyperbolic Equations with High Order of Accuracy*, Comm. Pure and Applied Mathematics, 17, 381-98, 1960

[11] MacCormack, R. W., *The Effect of Viscosity in Hypervelocity Impact Cratering*, AIAA Paper 69-354, 1969

[12] Lerat, A., *Implicit Methods of Second Order Accuracy for the Euler Equations*, AIAA Paper 83-1925, AIAA 6th Computational Fluid Dynamics Conference; also AIAA Journal, 23, 33-40, 1983

[13] Briley, W. R., and McDonald, H., *Solution of the Three-Dimensional Navier-Stokes Equations by an Implicit Technique*, Proc. Fourth International Conference on Numerical Methods in Fluid Dynamics, Lecture Notes in Physics, Vol. 35, Berlin: Springer, 1975

[14] Beam, R. M., and Warming, R. F., *An Implicit Finite-Difference Algorithm for Hyperbolic System in Conservation Law Form*, Journal of Computational Physics, 22, 87-109, 1976

[15] Jameson, A., Schmidt, W., and Turkel, E., *Numerical Solution of the Euler Equations by Finite-Volume Methods Using Runge-Kutta Time-Stepping Schemes*, AIAA Paper 81-1259, June 1981

[16] Courant, R., Isaacson, E., and Reeves, M., *On the Solution of Nonlinear Hyperbolic Differential Equations by Finite Differences*, Comm. Pure and Applied Mathematics, 5, 243-55, 1952

[17] Steger, J. L., and Warming, R. F., *Flux Vector Splitting of the Inviscid Gas-Dynamic Equations with Application to Finite Difference Methods*, Journal of Computational Physics, 40, 263-93, 1981

[18] Van Leer, B., *Flux Vector Splitting for the Euler Equations*, Proc. 8th International Conference on Numerical Methods in Fluid Dynamics, Springer Verlag, 1982

[19] Godunov, S. K., *A Difference Scheme for Numerical Computation of Discontinuous Solution of Hydrodynamic Equations*, Math. Sbornik, 47, 271-306, 1959 (in Russian), Translated US Joint Publ. Res. Service, JPRS 7226, 1969

[20] Engquist, B., and Osher, S., *Stable and Entropy Satisfying Approximations for Transonic Flow Calculations*, Mathematics of Computation, 34, 45-75, 1980

[21] Osher, S., *Riemann Solvers, the Entropy Condition and Difference Approximations*, SIAM Journal Numerical Analysis, 21, 217-35, 1984

[22] Roe, P. L., *The Use of the Riemann Problem in Finite Difference Schemes*, Lecture Notes in Physics, Vol. 141, 354-9, Berlin: Springer Verlag, 1981

[23] Roe P. L., *Approximate Riemann Solvers, Parameter Vectors and Difference Schemes*, Journal of Computational Physics, 43, 357-72, 1981

[24] Ferziger, J. H., and Peric, M., *Computational Methods for Fluid Dynamics*, Springer, 2002

[25] Wilcox, D. C., *Turbulence Modeling for CFD*, DCW Industries, December 2002

[26] Cebeci, T., and Smith, A. M. O., *Analysis of Turbulent Boundary Layers*, Ser. in Appl. Math. & Mech., Vol. XV, Academic Press, Orlando, 1974

[27] Baldwin, B. S., and Lomax, H., *Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flows*, AIAA Paper 78-257, 1978

[28] Bradshaw, P., Ferriss, D. H., and Atwell, N. P., *Calculation of Boundary Layer Development Using the Turbulent Energy Equation*, Journal of Fluid Mechanics, Vol. 28, Pt. 3, 593-616, 1967

[29] Spalart, P. R., and Allmaras, S. R., *A One-Equation Turbulence Model for Aerodynamic Flows*, AIAA Paper 92-439, 1992

[30] Launder, B. E., and Spalding, D. B., *Mathematical Models of Turbulence*, Academic Press, London, 1972

[31] Saffman, P. G., *A Model for Inhomogeneous Turbulent Flow*, Proc. R. Soc., Lond., Vol. A317, 417-433, 1970

[32] Sinica Computing Center, *mpi_lecuter.pdf*, http://computing.sinica.edu.tw /service/course/lecture_note/mpi_lecture/mpi_lecture.pdf, Last Accessed: 01 September 2005

[33] Baer, T., *Parallel Programming with MPI*, Ohio Supercomputer Center, April 2005

[34] Aoyama, Y., and Nakano, J., *Practical MPI Programming*, IBM Redbooks, August 1999

[35] De Zeeuw D., and Powell K. G., *An Adaptive Cartesian Mesh Method for the Euler Equations*, Journal of Computational Physics, Vol. 104, 56-68, 1993

[36] Usta, E., *Application of a Symmetric Total Variation Diminishing Scheme to Aerodynamics of Rotors*, PhD. Thesis, Georgia Institute of Technology, August 2002

[37] Tuncer, İ. H., *Lecture Notes on Computational Fluid Dynamics on Unstructured Grids*, Department of Aerospace Engineering, METU, 2003

[38] Aksel, M. H., *Notes on Computational Fluid Dynamics and Euler2D*, Department of Mechanical Engineering, METU, 2004

[39] Ni, R. H., *A Multiple-Grid Scheme for Solving the Euler Equations*, AIAA Journal, Vol. 20, 1565-71, 1982

[40] Ak M. A., *Analysis of Transient Regimes in Solid Rocket Propulsion*, PhD. Thesis, METU, June 2001

[41] Gönç L. O., *Computation of External Flow around Rotating Bodies*, PhD. Thesis, METU, March 2005

# APPENDIX A

# VOLUME CALCULATION OF A HEXAHEDRAL

In Fig. A.1, a hexahedral and three pyramids that compose this hexahedral is shown. In Fig. A.2, the pyramids that compose the hexahedral are given in a separated form. Volume of the hexahedral is calculated by calculating the volumes of the three pyramids.

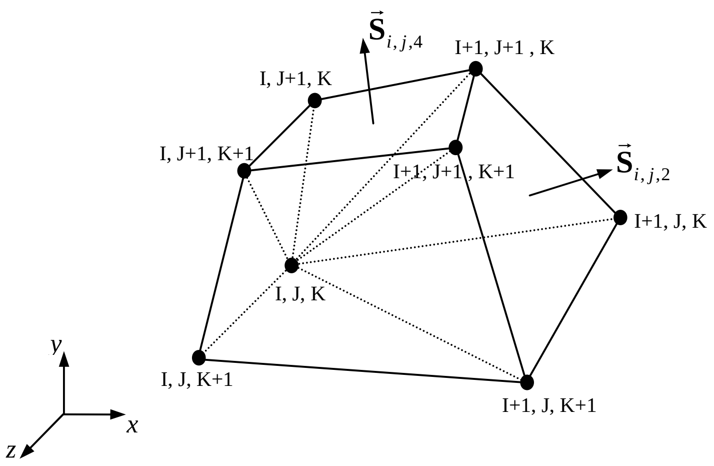


**Figure A.1:** A hexahedral and the three pyramids composing this hexahedral

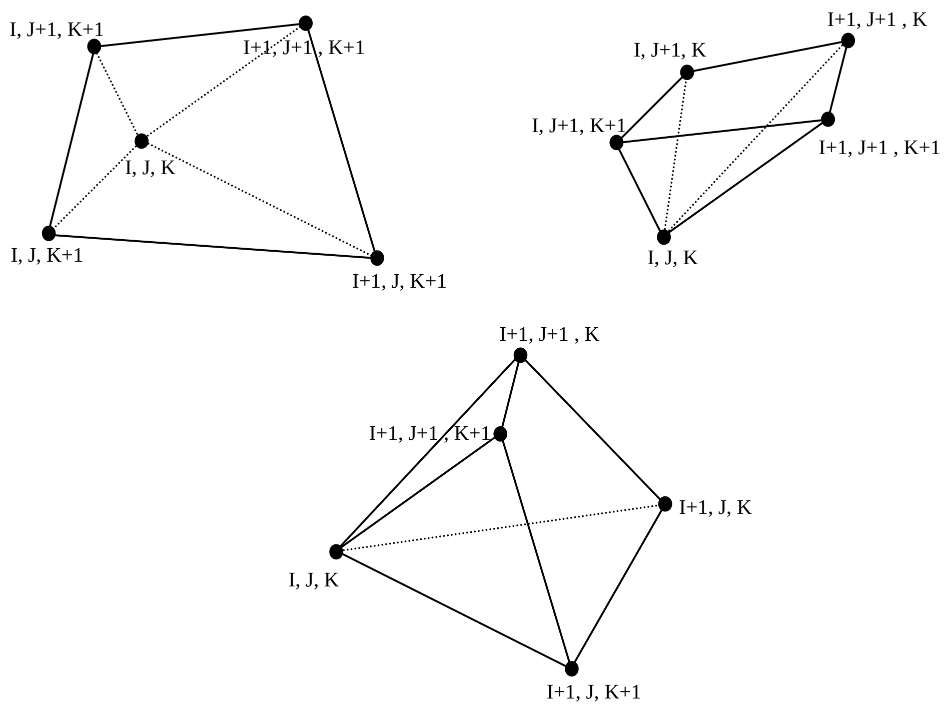


**Figure A.2:** The three pyramids composing the hexahedral

Volume of a pyramid is;

$$V = \frac{A.h}{3} \qquad \text{(A.1)}$$

where *A* is the base area, and *h* is the height of the pyramid.

Height of the pyramid can be found by using the formula of distance between a point and a plane which is given as;

$$d = \frac{\left(\vec{\mathbf{r}}_0 - \vec{\mathbf{r}}_1\right) \cdot \vec{\mathbf{N}}}{\left|\vec{\mathbf{N}}\right|} \qquad \text{(A.2)}$$

where $\vec{\mathbf{r}}_0$ is the position vector of the point, *P₀* for which the distance wanted to be found; $\vec{\mathbf{r}}_1$ is the position vector of an arbitrary point, $P_1$ on the plane, and $\vec{\mathbf{N}}$ is a vector normal to the plane. Direction of the normal vector should be towards to the side of the plane where $P_0$ point is located.

If the three pyramids in the figure are investigated, it can be seen that the three pyramids has (*I*, *J*, *K*) node as a vertex, and (*I*+1, *J*+1, *K*+1) node as a corner of the base. (*I*, *J*, *K*) node is the $P_0$ point, and (*I*+1, *J*+1, *K*+1) node is taken as $P_1$ point for the three pyramids. Normal vector of the base plane is taken as the surface area vector in the opposite sense.

# APPENDIX B

# CALCULATION OF CONVECTIVE FLUXES IN THREE DIMENSIONS

A three-dimensional flow becomes locally two-dimensional, if local coordinates are rotated in such a way that *z*-direction is perpendicular to the plane formed by velocity vectors of left and right cells. This plane is defined with the unit normal vector which is given as;

$$\hat{\mathbf{n}} = \frac{\left(\vec{\mathbf{V}}_{\mathbf{L}} \times \vec{\mathbf{V}}_{\mathbf{R}}\right)}{\left|\vec{\mathbf{V}}_{\mathbf{L}} \times \vec{\mathbf{V}}_{\mathbf{R}}\right|} \tag{B.1}$$

The desired rotation can be obtained by first rotating the local coordinates about *y* axis with the rotation matrix given in Eqn. (B.2). This rotation is followed by the rotation given by Eqn. (B.3) which is about *x* axis.

$$R(\psi) = \begin{pmatrix} Cos\psi & 0 & Sin\psi \\ 0 & 1 & 0 \\ -Sin\psi & 0 & Cos\psi \end{pmatrix}$$
$$Cos\psi = \frac{-n_z}{\sqrt{n_x^2 + n_z^2}} \qquad Sin\psi = \frac{n_x}{\sqrt{n_x^2 + n_z^2}} \tag{B.2}$$

$$R(\theta)=\begin{pmatrix}1 & 0 & 0\\ 0 & Cos\theta & Sin\theta\\ 0 & -Sin\theta & Cos\theta\end{pmatrix}$$

$$Cos\theta=\frac{-n'_z}{\sqrt{\left(n'_y\right)^2+\left(n'_z\right)^2}}\qquad Sin\theta=\frac{n'_y}{\sqrt{\left(n'_y\right)^2+\left(n'_z\right)^2}} \tag{B.3}$$

where the primed vector components are the components of unit normal vector given in Eqn. (B.1) after the rotation matrix $R(\psi)$ is applied.

After the above rotations, convective fluxes are calculated with the same formulation given by two-dimensional case. Flux created by the pressure component in the direction of unit normal vector is added to the fluxes found on the plane.

As the last step, fluxes are rotated to the global coordinates by applying $R(-\theta)$ rotation which is followed by $R(-\psi)$ rotation.